DOI: 10.1111/cgf.15014

EUROGRAPHICS 2024 / A. Bermano and E. Kalogerakis
(Guest Editors)

COMPUTER GRAPHICS *forum*
Volume 43 (2024), Number 2

Real-time Neural Rendering of Dynamic Light Fields

Arno Coomans[1] Edoardo A. Dominici[1] Christian Döring[2] Joerg H. Mueller[3] Jozef Hladky[2] Markus Steinberger[3,4]

Huawei Technologies, [1]Switzerland, [2]Germany, [3]Austria
[4]Graz University of Technology, Austria

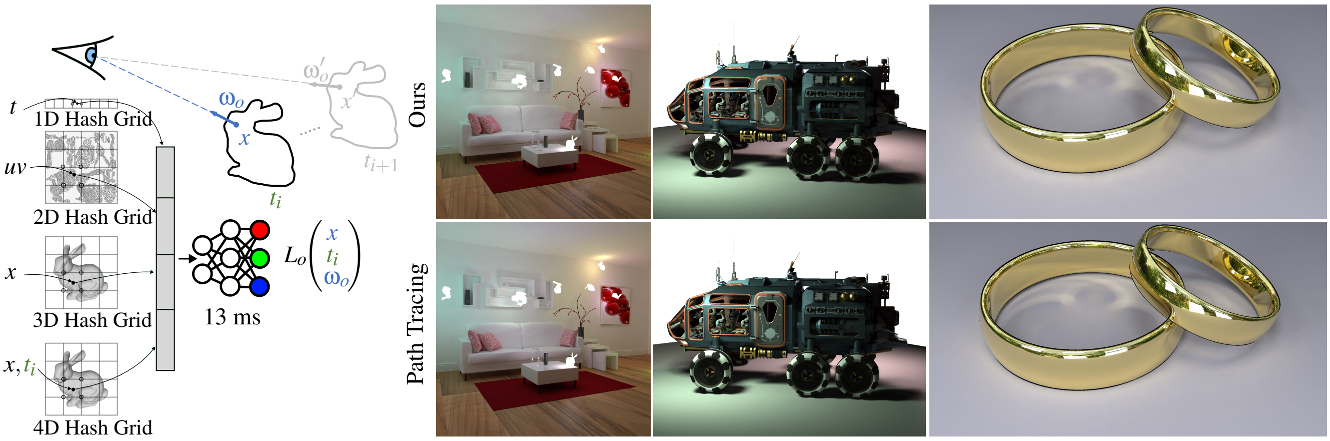


Figure 1: *Our method uses a small neural network to explicitly represent a dynamic light field, acting as a radiance cache for spatio-temporal samples (x, ω_o, t_i). During inference, a simple visibility pass provides (x, ω_o) samples at time t_i. Our novel combination of parameter encodings enables a small neural network to compute highly detailed radiance estimates in real-time. Due to the low computation cost of ∼13ms with all our encodings enabled, we can render a full neural light field of a dynamic scene with more than 60 frames per second in 1920 × 1080 resolution. Additionally, our parameter encodings capture different scene dynamics within a compact memory footprint (40-80 MB). In the dynamic scenes above, our neural dynamic light field representation is $\geq$ 20 times faster than previous works [HCZ21, DPD22].*

Abstract

Synthesising high-quality views of dynamic scenes via path tracing is prohibitively expensive. Although caching offline-quality global illumination in neural networks alleviates this issue, existing neural view synthesis methods are limited to mainly static scenes, have low inference performance or do not integrate well with existing rendering paradigms. We propose a novel neural method that is able to capture a dynamic light field, renders at real-time frame rates at 1920x1080 resolution and integrates seamlessly with Monte Carlo ray tracing frameworks. We demonstrate how a combination of spatial, temporal and a novel surface-space encoding are each effective at capturing different kinds of spatio-temporal signals. Together with a compact fully-fused neural network and architectural improvements, we achieve a twenty-fold increase in network inference speed compared to related methods at equal or better quality. Our approach is suitable for providing offline-quality real-time rendering in a variety of scenarios, such as free-viewpoint video, interactive multi-view rendering, or streaming rendering. Finally, our work can be integrated into other rendering paradigms, e.g., providing a dynamic background for interactive scenarios where the foreground is rendered with traditional methods.

CCS Concepts

- ***Computing methodologies*** → *Ray tracing; Neural networks;*

1. Introduction

Rendering high-quality images with multi-bounce global illumination in real-time has been a long term goal of rendering research. Because high-quality path tracing for non-trivial scenes is still far from real-time, precomputing complex lighting remains an attractive alternative. Ideally, a precomputed cache should capture a spatio-temporal, omni-directional light field, i.e., support radiance queries at any point in a scene, for any direction, at any point in time. Such a cache may be used in many applications. For example, free-viewpoint 3D videos can be rendered directly from it; the backdrop rendering of animated scenes can be served from the cache, while foreground scene interactions are rendered on demand and use the cache as an environment map; material changes can be encoded into the time domain to, e.g., allow product customization like recolouring, and even dynamic lighting changes like a complete day/night cycle can be captured through the dynamic dimension of the cache. Obviously, for real-time applications, querying the cache must be highly efficient. Additionally, a compact memory footprint, would allow streaming the cache, e.g., to enable free-viewpoint video with a lightweight head mounted display.

Having access to a model that represents outgoing radiance has two advantages over sampling-based methods at query time. First, it reduces the complexity of rendering to running primary ray intersection pass and a subsequent cache evaluation, which is significantly faster than multi-bounce path tracing. Second, the image directly rendered from the cache does not suffer from sampling-related variance on the radiance estimate.

Precomputing the radiance transfer for static scenes [SKS02] has been researched for decades. However, computing and caching a light field for dynamic scenes comes with a multitude of challenges. A naive dynamic cache might attempt to discretise across all dimensions to store radiance estimates. While conceptually possible, such an approach suffers from high memory requirements. Neural rendering methods of synthetic [HCZ21, RBRD22, DPD22] and real [MST*20] scenes overcome these problems by compressing and approximating the light field with a neural network [MST*20]. While neural rendering methods are becoming indistinguishable from Monte Carlo simulations, high-quality approaches typically do not support real-time and/or dynamic settings. We use the term *light field* in favour of radiance field to distinguish our work from existing neural radiance fields (NeRFs), which often do not have access to scene parameters during training and inference.

Existing neural rendering methods mainly focus on static scenes, while some partially support dynamic scenes. *Neural Radiance Caching (NRC)* [MRNK21] learns a radiance cache of the indirect illumination on the fly which is combined with direct illumination estimation at runtime. When the scene conditions change the radiance estimate will take some time to adapt, resulting in bias w.r.t. the current time step. This bias and variance due to sampling the light sources, can be sidestepped in the case where the scene dynamics are known beforehand. Namely, we want to learn the full dynamic light field of a fixed animation compared to adapting to a dynamic scene at runtime. *Neural Radiosity* [HCZ21] learns a light field of a static scene, but supports dynamic environments through transfer learning. Online transfer learning as presented in Neural Radiosity would not work in real-time applications due to the still significant training time. *Active Exploration* [DPD22] comes closest to our goals, as it learns the light field for each permutation of selected scene parameters. Unfortunately, active exploration is far from real-time and struggles to capture high frequency effects.

We make two assumptions about the context in which the cached dynamic light field is used. First, to keep the computational cost low the cache is directly used to synthesise images, without deferring to further bounces (unless a fully reflective surface is hit). This implies that the cache needs to capture both the spatial and temporal aspects of the scene well, otherwise any unfiltered error in the radiance estimate will be directly visible in the synthesised image. Second, the cache can be trained as precomputation. Capturing a complex spatio-temporal signal—at the very least—requires exploring the signal and thus requires a compute budget similar to path tracing many frames.

In our work, we address the shortcomings of prior work and propose a real-time neural rendering approach for a dynamic light field. We make the following contributions:

1. We propose a real-time capable spatio-temporal cache to approximate a dynamic light field, relying on small neural networks that can be efficiently evaluated [Mül21] and a combination of multiple spatio-temporal hash grid encodings [MESK22] to optimally capture different scene dynamics.
2. We quantitatively and qualitatively investigate several possible encodings for scene dynamics and present a novel surface-space encoding, significantly improving temporal and spatial stability.
3. We present an improved self-training strategy inspired by reinforcement learning, which shows better sampling distributions and an improved gradient flow compared to previous work [HCZ21].

2. Related Work

Many solutions to multi-bounce global illumination have been proposed over the years by either improving the quality of sampling [Vea98, LW95] or by caching [WRC88, KGPB05] solutions of the rendering equation [Kaj86]. A shared property of state-of-the-art methods is that they are data-driven; they build statistical models over previously seen information, for example by progressively learning a sampling distribution [MMR*19, BWP*20] or constructing a neural model to estimate the outgoing radiance [RWG*13, MRNK21].

Precomputed Radiance Transfer for low-frequency far-field illumination was successfully achieved by Sloan et al. [SKS02], and later extended to enable self-shadowing [KLA04], near-field lighting [KAMJ05] and more dynamic lighting conditions [RWG*13]. While capturing complex global illumination in the presence of dynamic geometry and lighting remains an open challenge, neural representations have shown promise in accelerating light sampling of complex geometric lights [ZBX*21], importance sampling [MMR*19] and representing materials [ZRW*23].

Neural Rendering of Synthetic Data has attracted increasing interest over the last few years. Hadadan et al. [HCZ21] proposed to solve the rendering equation for static scenes with a formulation similar to classical radiosity techniques. Their approach generates random surface samples and minimises the loss between the prediction of outgoing radiance in a sampled direction and a Monte Carlo estimate of the integrand using self-learning. Other methods operate in image space, they can rely on cheap to compute G-buffers [TF17]. Image space approaches can even be extended to dynamic scenes [GRPN20, DPD22, ZHM*23] by augmenting the input domain with scene parameters. However, previous image space methods rely on relatively large neural networks. With our method, we show that a combination of smart parameter encodings allows us to use a smaller neural network, boosting inference time by more than an order of magnitude while still obtaining high-quality radiance estimates. Furthermore, our approach is capable of handling much more complex scene dynamics.

Neural Radiance Fields (NeRFs) [MST*20] typically operate on real world image or video data, which has a number of disadvantages compared to using synthetic data. Nevertheless, NeRFs have been extended to the temporal domain although the signal is much sparser than in the static one [SCL*23]. A large class of methods [PCP-MMN20, FYW*22, GCD*22] learn a deformation to a canonical space, on which they apply volume integration techniques to render the image. These approaches are well-suited for single animated objects. Time can be fed as a latent representation to the neural radiance field, e.g., by simple concatenation [LSZ*22], as 4D spatio-temporal positional encoding [PSJ*23] or by lifting time to higher dimensions using additional networks [YJM*23, FYW*22, PSJ*23]. Additionally, one can further reduce the dimensionality of the latent space using tensor decomposition, notably 2D-2D [SZT*23] and 3D-1D [IRG*23]. To handle long animations, it can be split into multiple segments and interpolated. Techniques that adaptively segment the animation sequence [IRG*23] are orthogonal to our work and can be adapted to our models. Though not leveraging neural networks, Kerbl et al. [KKLD23] introduced a promising approach to render static radiance fields in real-time. In general, capturing full radiance fields from images or videos is a much more difficult problem, as the scene geometry is unknown and only a predetermined number of samples is available for training. In our setting, we consider the scene geometry known and can actively generate samples for training where they are needed most.

Real-time rendering of global illumination in a dynamic scene shares some goals with us, but our target is to obtain offline-quality global illumination (GI) in a compact representation that can be rendered in real-time. Baked GI approaches typically learn per-scene low-frequency diffuse indirect lighting, often stored in light probes, which can be updated dynamically [MGNM19, MMK*21]. Similarly, a neural radiance cache can learn indirect lighting during runtime [MRNK21]. While approaches that rely on online updates have the advantage that they can adjust to unknown scene dynamics, they rely on sampling at inference time (variance) and a time window in which their learnt radiance representation is outdated (bias). With our approach, we capture the full, high-quality light field of a fixed dynamic scene and enable rendering with real-time frame rates directly from a cache.

Object space and texture space shading methods store view-independent shading into an off-screen texture [Bak16, BFM10], which can serve as a cache for simple shading. Storing view-dependent shading information in a texture limits its use to equal or nearby viewpoints. However, temporal and spatial coherence can be exploited to reduce shading efforts [HY16, MNV*21, NMSS22]. Additionally, the temporal coherence enables split rendering pipelines and framerate upsampling [MVD*18, HSS21, HSV*22]. Storing intermediate effect computations in a texture-based cache, allows to share and reuse a large amount of computations [WTS*23]. While the previously mentioned approaches have focused on simple effects, they have demonstrated that an UV-space may provide superior spatio-temporal coherence compared to other parameterizations. In our work, we exploit the UV-space as additional parameter space demonstrating that it can also help capture spatio-temporal coherence for caching complex light fields.

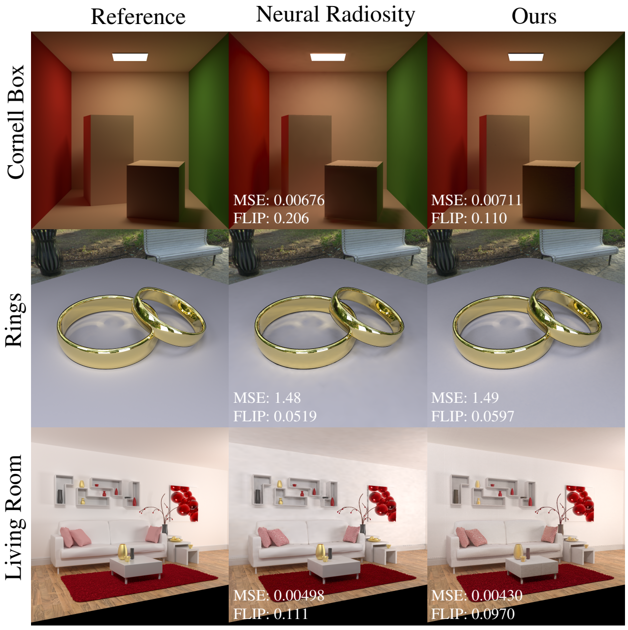


Figure 2: *Equal training time comparison between our method and the static Neural Radiosity [HCZ21]. Even though we support dynamic scenes and are 20× faster during inference, our radiance estimates are similar to theirs.*

3. Dynamic Neural Light Fields

Directly synthesising images using the output of a learned light field, reduces the complexity of rendering to an efficient visibility pass and a network query. We use ray tracing for the visibility, typically stopping at the first bounce to query the cache. In case the first hit surface is reflective, we trace the bounced ray, avoiding the overly complicated task of learning the outgoing radiance of mirror surfaces, which would potentially require learning how the entire scene can be seen from each point of a mirror surface. We want to stress again that this task is different from NeRFs, which also need to learn the scene geometry alongside the radiance and draw multiple samples along each ray. In our case, we query the cache only once for each view ray (as we encounter a non-reflective bounce). This not only simplifies the task of learning, but also allows significantly more efficient inference, as the number of network evaluations is greatly reduced.

With the goal of providing a high-quality, real-time cache, we forgo complex network architectures, as it would be much more challenging to optimise inference time, and rather offload the complexity of the learning to the encodings. We equip a multilayer perceptron (MLP) with a collection of multi-resolution hash grids [MESK22], which not only leads to faster training times but is also easier to adapt to domains other than spatial, does not require any precomputation before training (like an occupancy grid) and exposes a small set of parameters that can be used to tune the number of optimization variables.

3.1. Basic Architecture

Our dynamic light field cache is conceptually similar to Neural Radiosity [HCZ21], but captures a dynamic light field $L(x,\omega,t)$ using a neural dynamic light field $L_\theta(x,\omega,t)$, parameterised by a neural network with parameters θ. In the following, we draw comparisons with Neural Radiosity; to simplify the discussion, we will omit t for brevity and consistency with the formulation in Neural Radiosity. Figure 2 shows the output of our method compared to Neural Radiosity on static scenes.

To train the cache, we collect samples of the residual (Equation 1) of the rendering equation, effectively training the cache on itself through self-training [DK17]. Following the methodology of Neural Radiosity, we take N Monte Carlo samples of the norm of the residual (Equation 2) sampling a set of positions x and directions ω_o. For each of these outgoing radiance pairs, we trace rays towards M points on the hemisphere using multiple importance sampling between the BSDF and the light sources, which are used to estimate the scattered radiance on the surface. Given these samples, we can update the network to minimise r_θ using automatic differentiation and an off-the-shelf optimizer. Network and sampling parameters are listed in Appendix A.

The signal stored in the cache converges to a high-quality estimate of the light field given sufficient network capacity and training time, see Hadadan et al. [HCZ21] for details on the gradients. The following formulas describe the residual (Equation 1) and the estimation of the norm of the residual (Equation 2), where $E(x,\omega)$ denotes the emitted outgoing radiance towards direction ω, f is the bidirectional scattering distribution function, $p(x,\omega)$ is the probability of sampling a point and outgoing direction and $x'(x,\omega)$ is the intersection obtained through tracing a ray from x towards ω against the scene geometry.

$$\begin{aligned} r_\theta(x,\omega_o) &= L_\theta(x,\omega_o) - E(x,\omega_o) \\ &\quad - \int_\Omega f(x,\omega_i,\omega_o)\, L_\theta(x'(x,-\omega_i),-\omega_i)\, d\omega_i^\perp \end{aligned} \tag{1}$$

$$\mathcal{L}(\theta) \approx \frac{1}{N}\sum_{j=1}^{N} \frac{r_\theta(x_j,\omega_{o,j})^2}{p(x_j,\omega_{o,j})} \tag{2}$$

Extending this formulation to a dynamic light field, adds an extra input t (regardless of used encoding) to the network that needs to be sampled repeatedly during training. While it is technically possible to randomly sample this time t for every sample in the batch, we opt to not do this due to the prohibitive cost of keeping thousands of instances of the scene in memory at the same time. Concretely, we use one time step for each batch, effectively biasing the gradient for that batch. However, since across batches this time step is sampled randomly, training is unbiased in the limit [GBC16]. We experimented with mixing several samples of t in a batch but found no significant changes in the quality of the results.

To achieve real-time inference, the neural network representing the dynamic light field needs to be highly efficient. To that end we use tiny fully fused neural networks [Mül21]. While making the network trivially small reduces the computational load, it also reduces the overall representational strength of the network. We suggest several spatio-temporal encodings in the following section that help our dynamic light field's small network to represent the complex spatio-temporal signal.

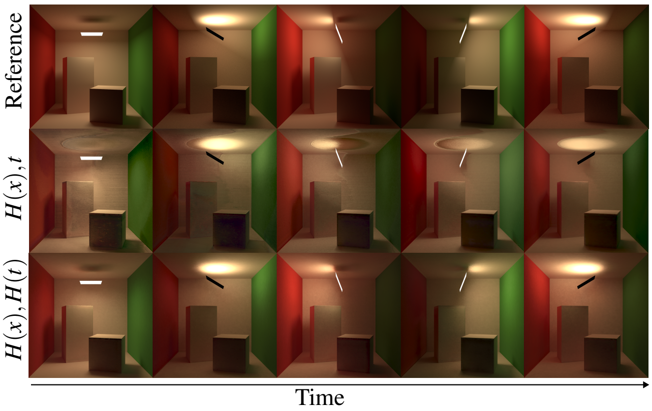


Figure 3: *Using only time as additional input the network is not capable of producing accurate dynamic lighting effects, even in a trivial scene like the Cornell box, with a simple rotating light source.*

3.2. Spatio-temporal Encodings

It is unclear from prior work which temporal encoding is optimal, especially for capturing a dynamic light field of a synthetic scene. We show that combining encodings in different domains, while reducing the number of latent dimensions, is an effective strategy in capturing a variety of dynamic light fields. Specifically, we focus on the following encodings:

- A 4D spatio-temporal encoding $H(x,t)$ that captures correlations between the spatial and temporal signal in the light field to better represent radiance flowing through space coherently.
- A 3D spatial encoding $H(x)$ which extracts the static features of the scene for a more stable reconstruction.
- A novel 2D UV-space encoding $H(uv)$ which improves robustness for fast moving objects by informing the radiance network of quantities that are position or rotation-invariant.
- A 1D temporal encoding $H(t)$ that yields a higher dimensional representation for time to capture smoother changes in light intensity.

We found that the combination of these encodings is necessary to achieve high-quality radiance estimates for challenging dynamic scenes under a constraint memory and computation budget. It is worth noting that we also supply basic parameters as additional variables to the network, including 3D position, time, UV-coordinates, outgoing direction and surface properties—see Appendix A.

Starting from existing static neural light fields methods [HCZ21], a naive extension would be to simply pass an extra float t to the network. As can be seen in Figure 3, using the time t directly as input to the neural network MLP$(H(x),t)$ is not robust enough, as

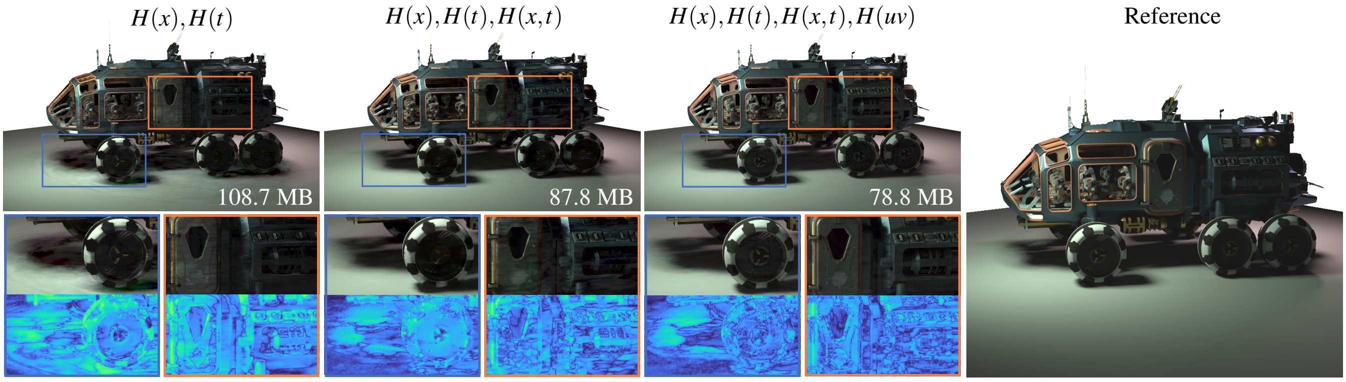


Figure 4: *In the Rover Temporal scene the rover is lit by multiple light sources and rotates quickly leading to challenging shadows on the floor as well as significant lighting changes on the rover. We incrementally add parameter encodings while reducing the number of optimizable parameters. The bottom row shows the FLIP error visualised with the viridis colour map: (left) A separate spatial and temporal encoding $H(x),H(t)$ is unable to capture high frequency details. (centre) Adding a spatio-temporal encoding $H(x,t)$ better captures the dynamic shadows on the floor. (right) Only a surface-level encoding $H(uv)$ is capable of reconstructing fine details on the rover.*

even with slowly changing low frequency illumination, the network is not capable of learning smooth transitions over time. Therefore, it is necessary to lift time to a higher dimensional latent representation. The addition of a simple temporal encoding $H(x),H(t)$ is sufficient for the simple Cornell Box scene, but becomes impractical for more complex scenes with varying animation. It would require a substantial growth in network size which would be negative to our inference times. In the following experiments we evaluate the quality of different temporal encodings for different animation types.

Fast Moving Objects Figure 4 shows a complex rotating object, which casts a shadow on a static ground plane. Due to moving geometry, the state of spatial grid cells fluctuates between a complete occlusion and bright illumination (see supplementary video). There is a strong correlation in the illumination as the shadows move across the ground plane, which can be exploited to better represent the light field. Instead of making the network significantly deeper and wider, we opt to leverage a 4D $H(x,t)$ grid to encode this correlation. This approach is particularly effective at representing the shadows on the ground (centre).

The high-dimensional 4D grid creates noticeable artefacts on regions of the Rover that traverse more spatial grid cells (door and engine compared to the roof of the vehicle). While the outgoing radiance for these points is fairly similar in a small window of time, due to the constant rotation, the network needs to learn this offset as well. Even more, the difference in tangential speed across the mesh results in different spatial and temporal frequencies, further increasing the complexity of the learning task.

A surface-based encoding can help the network, as it directly targets changes in illumination *on the surface*. Our novel 2D surface-level encoding $H(uv)$ is inspired by object-space lighting approaches that have been shown to improve spatio-temporal coherence [MNV*21]. We add a single multi-level grid on top of a *UV atlas* for the entire scene, which combines the UV maps of each object proportionally to their surface area. This new encoding shows considerable quality improvements (Figure 4 and 6) . Figure 5 shows the UV map of a simple scene that uses a hash grid over the UV domain and the visualisation of the outgoing radiance in the normal direction in the bottom row. Shading effects that are translation and/or rotation invariant are captured well by the encoding. Relying on a UV-encoding allows us to capture spatio-temporally stable shading for moving objects, which would otherwise show up as dynamic high frequency effects in 3D space. Appendix B provides details on the creation of the UV atlas.

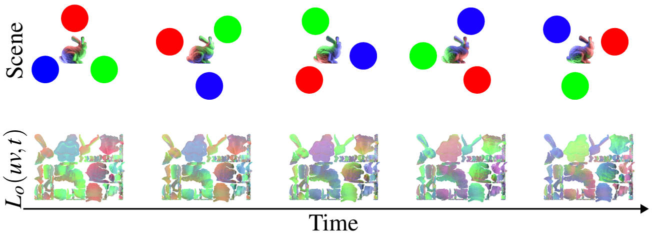


Figure 5: *Visualising the predicted radiance from a network trained with a 2D grid encoding over the UV domain: For each texel in texture space we evaluate the network with the surface normal as outgoing direction and plot the colour for various frames. We show that the network is able to learn texture space shading.*

Static Geometry with High-Frequency Lighting Another dynamic scene is presented in Figure 6, the *Dining Temporal* scene, which is lit mostly indirectly by a directional light that rotates and a table that is translating. The transformations of the objects are non-linear. The table, hanging lamp and blinds cast moving hard shadows on the wall. Even though the scene geometry is mostly static, the $H(x),H(t)$ encoding struggles to keep up with the lamp moving and loses the hard shadows towards the end of the animation. Similarly to the Rover scene (Figure 4), capturing the correlation of the signal as it moves across the left wall is important, but in this case the lighting is much higher frequency and the 4D spatio-temporal grid is unable to preserve the sharp boundaries during the animation. To solve both issues simultaneously we concatenate the hash grids into $H(x),H(t),H(x,t)$. Due to the mostly static nature of the scene,

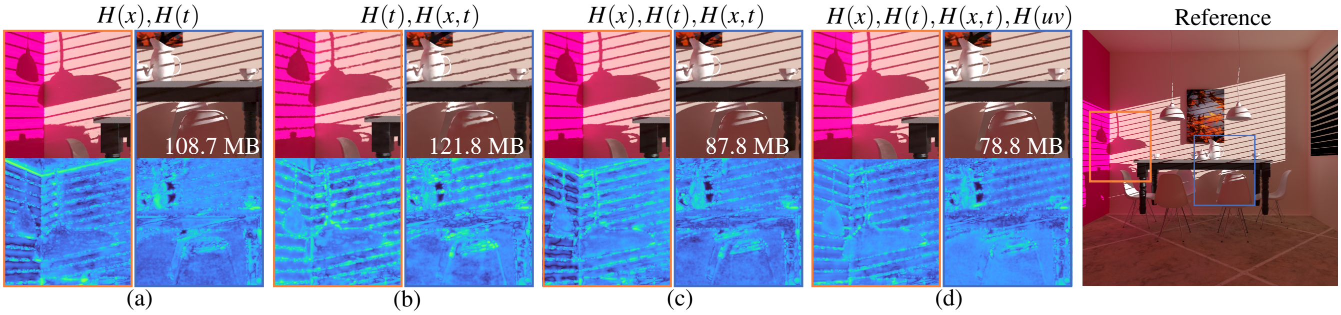
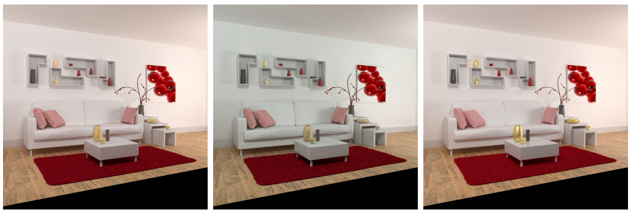


Figure 6: *We show the quality of different encodings in the Dining Temporal scene. The bottom row shows the FLIP error visualised with the viridis colour map. The advantage of (b) the dynamic encoding compared to (a) a separate spatial and temporal encoding is that it can better track the moving hard shadows over time. Unfortunately, it comes with a disadvantage as the shadows now have fuzzy boundaries. (c) The union of the static and dynamic encoding combines the benefits of both by being able to capture the sharp boundaries more accurately for the entire length of the animation. (d) Although the scene geometry is mostly static, adding the surface encoding does not deteriorate the quality.*

Table 1: *Inference time of the neural representation in milliseconds for different encodings and their combinations. As expected, the overhead of each encoding is proportional to its dimensionality. Reducing the resolution of the intermediate level of the grid can result in performance improvements. Note that the columns are not cumulative as they contain the network inference time. α is the albedo of the surface. All inference information is obtained with an RTX 4090 and i9-13900K.*

Model	$\vec{x},\vec{n},\vec{w},\alpha$, uv, t	$H(x)$	$H(t)$	$H(x,t)$	$H(uv)$	Σ
Full	2.76	2.66	1.84	3.68	2.09	12.65
Low res.	2.73	1.85	1.54	2.25	1.61	7.85



(a) *Stop Gradient* **(b)** *No Stop Gradient* **(c)** *Reference*

Figure 7: *Comparison of the learned light field after 40,000 training iterations. Not stopping the gradient darkens the Living Room and the colour bleed across the scene is not captured well. While this scene is one of the more extreme examples, we notice similar qualitative differences on all scenes. Blocking the gradient's flow never decreases the quality.*

adding $H(uv)$ should not improve quality. Interestingly, the overall error is slightly removed still, highlighting that the UV-encoding in general does not negatively impact quality.

3.3. Practical Considerations

Stop gradient. Since we use self-learning, the neural network appears twice in the gradient computation of the norm of the residual, once for the outgoing radiance and N times for the sampled incoming radiance. In contrast to the Neural Radiosity method, we found that stopping the gradient flow of the N incoming radiance estimates performs significantly better. Namely, the cost of training goes down due to the reduced size of the computational graph used in automatic differentiation. Qualitatively, we also gain up to an order of magnitude of improvements in some scenes, see Figure 8. Equation 3 shows the modified formulation of the residual, sg stops the gradient flow of its parameters.

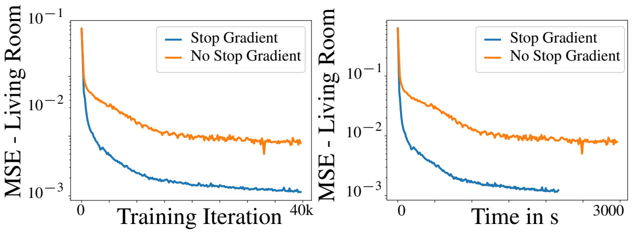


Figure 8: *Difference in image quality due to the use of a stop gradient function, blocking the gradient flow greatly improves the convergence of the network. While subtle, the reduced computational cost of a smaller computation graph increases training steps per second.*

$$\begin{aligned} r_\theta(x,\omega_o) = {} & L_\theta(x,\omega_o) - E(x,\omega_o) \\ & - \int_\Omega f(x,\omega_i,\omega_o)\, sg(L_\theta(x'(x,-\omega_i),-\omega_i))\, d\omega_i^\perp \end{aligned} \tag{3}$$

We notice a strong similarity with deep reinforcement learning methods where stopping the gradient flow towards the target network [MKS*15, SB18] is common practice. Recent graphics publications [DK17] hinted at the similarity of reinforcement learning to the rendering equation and provided similar experimental results by stopping the gradient flow [HLN*23].

Improved sampling. To estimate the norm of the residual (Equation 2) pairs of (x,ω_o) need to be sampled from $p(x,\omega_o)$. In comparison to Neural Radiosity, we use area importance sampling of x by building a cumulative distribution function (CDF) over the objects'

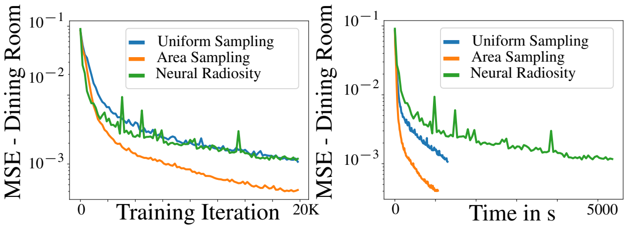


Figure 9: *Impact of positional sampling of our implementation (blue and orange) and the open source implementation of Neural Radiosity [HCZ21] (green) on the quality of the synthesised images of the Dining Room. Note that uniform sampling refers to uniformly selecting an object irrespective of its surface area, not uniformly distributing points in the scene.*

surface areas, while still uniformly sampling the hemisphere. The benefits of positional sampling are mainly noticeable in scenes with strongly varying surface area between objects. For example, using uniform object sampling in the *Dining Room* would allocate the same amount of samples to one of the tea cups as to all the walls in the room. Overall, importance sampling the area of the objects improves performance significantly, see Figure 9 for a comparison between our implementation and the open source version of Neural Radiosity. Additionally, in Appendix C, we show a heat map of the sample density of a batch of positions.

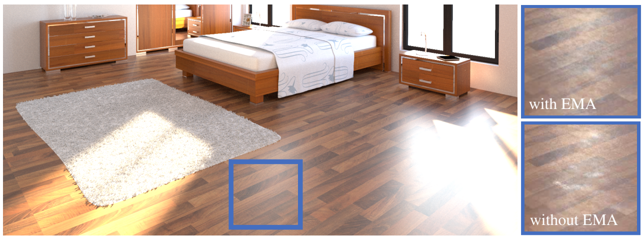


Figure 10: *Improving temporal stability by computing the exponential moving average of the temporal inputs $H(t), H(x,t)$ and t before feeding them to the dynamic neural light field.*

Temporal Stability. Due to the high frequency hash grid encoding, high frequency patterns might pop in and out of neighbouring temporal frames of a non-converged dynamic light field. Existing methods that adaptively train a neural network to capture the light field [MRNK21] suggest computing an exponential moving average (EMA) of the network weights of neighbouring time steps to improve temporal stability. Since our network weights are the same for every point in time, we choose to compute an EMA on the temporal subset of the network inputs: $H(t), H(x,t)$ and t. While an EMA of the entire input vector works even better for a static camera and scene, it adds ghosting artefacts when either the camera or the geometry move. See Figure 10 and the supplementary material for a video on the effects of EMA.

4. Results

We focus on a variety of scenes which exhibit different temporal behaviour. We are interested in varying frequencies of animation, soft and hard shadows, static and dynamic geometry of different complexity as well as lighting changing in intensity and tint. For our tests, we built on top of the standard dataset of Bitterli [Bit16]: The scenes and their animations are depicted in Figure 11 as well as the supplementary video and they each present different challenges: The *Rover* is a complex mesh with diffuse, dielectric and rough conductor surface BSDFs rotating around itself. The *Slow* version performs a 180 degree rotation, while the *Fast* performs a full 360 degree rotation. It is lit exclusively by two area lights that cast soft shadows on the floor and results in high frequency changes in lighting on the surface of the vehicle. The light in the *Cornell Box* is rotating by the specified amount of degrees. As the light is one sided and close to the ceiling, it creates a strong highlight with a quickly moving boundary between dark and light. The *Dynamic Rings* scene contains complex caustics being cast on the floor by the ring geometry through 6 different area light sources, which make a full rotation around the scene resulting in rotating caustics; *Dining temporal* is made dynamic by the table moving up (*Slow*) and down again (*Fast*) as well as a directional moving light cast through the blinds. This results in spatially high frequency hard shadows cast on the wall which move in a non-linear way. The *Bunnies* scene is a long animated sequence where multiple rotating bunnies are scattered in the scene with high velocity and collide elastically with the geometry. As the bunnies emit differently coloured light, tinted highlights appear and disappear at various points in time and space; The *Bedroom* scene is lit by a rotating environment map. This creates a strong highlight that can be seen moving across a textured floor with a rough plastic BSDF; The *Chair* and *Mirror Chair* contain the same animation of a tilting chair, which is completely reflective in the mirrored version. The main challenge is the hard shadow that is cast on the floor.

Note that we allocate an equal amount of memory to all variants of our methods, especially when varying the encodings where we ensure that our full model always stays below 80MB. This is similar to the budget used by Hadadan et al. [HCZ21]. We do not fine-tune the parameters in a scene dependent way, but rather use a robust set of parameters (Appendix A) that work well on a wide range of scenes.

We use Mitsuba [JSR*22] as the basic framework for training and rendering. Since Mitsuba does not explicitly support dynamic scenes, we treat them as a sequence of static scenes, resulting in a significant overhead for reoccurring scene and data structure setup. Thus, we ignore these costs for rendering time measurements for all tested methods and only focus on the inference speed of the neural representation.

4.1. Static Scenes

While not the focus of this paper, we briefly discuss our performance on static scenes. Due to our network architecture, encoding and training setup (see section 3.3), we significantly outperform Neural Radiosity [HCZ21], running at ≥ 60 frames per second in full HD compared to two frames per second, see Figure 2. Comparing only the respective speed of the dynamic light field representation—without Mitsuba—we are around 23 times faster at runtime, while the quality of our renders is similar. As the network does not simply learn outgoing radiance towards the camera—it learns the full light

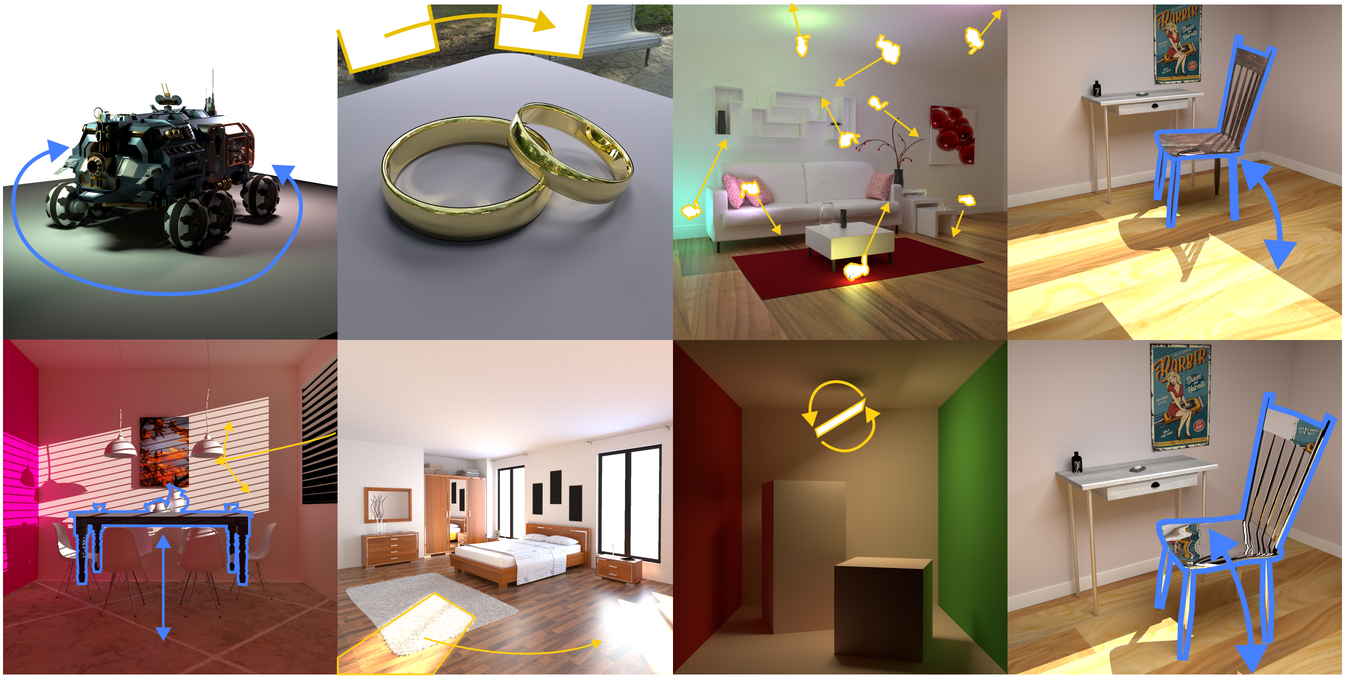


Figure 11: *Overview of the animations used in Table 3. From left to right and top to bottom: Rover, Rings Dynamic, Bunnies, Chair Tilting, Dining Slow/Fast, Bedroom, Cornell 180/360/720, Chair mirror tilting.*

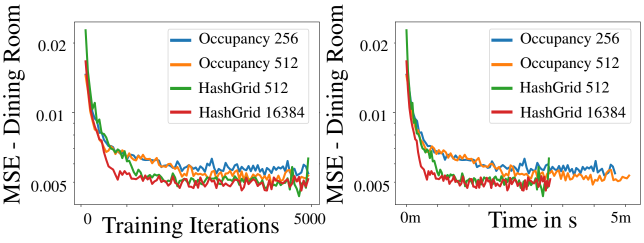


Figure 12: *A hash grid encoding [MESK22] is able to allocate budget dynamically and in our scenes always performs better or equivalent to sparse feature grids [HCZ21].*

field—the scene can be freely explored from all possible camera angles at runtime.

We compare the use of hash grids with the static sparse feature grids from Neural Radiosity [HCZ21] in Figure 12, concluding that while qualitatively they behave similarly, the hash grids are more modular and have a higher performance. On top of this, extending hash grids to other types of encodings such as time and UV coordinates is straightforward and the hash grids do not require slow precomputation, e.g., 1 hour in the *Rover* scene.

4.2. Performance

Key runtime performance factors are the number of inputs to the neural network and the coherence of the memory access of the hash grids. To assess these two factors, we decrease the budget of the grids by reducing the number of intermediate levels while keeping the dimensions of the features and the minimum and maximum resolution fixed. We find that this provides a single practical control to memory and performance and show the results of such experiment in Table 3. This lower resolution model uses roughly 50% of the memory while only slightly reducing the visual quality, indicating that tuning the network/encoding capacity on a scene by scene basis can further improve inference speed. A compact memory representation is crucial for network streaming and low performance hardware support. In our experiments, reducing the size of the network even further showed to be more harmful to quality than reducing the dimensionality of the grids. The inference time of this low resolution model is about 35% lower compared to the full resolution model as shown in Table 1.

Resolution	Active Exploration	Neural Radiosity	Ours
1280x720	128.53	134.58	**6.62**
1920x1080	279.96	293.76	**12.65**

Table 2: *Comparing the inference time of the neural network of [DPD22] and [HCZ21] as provided by their open source implementations. The time is reported in milliseconds for various resolutions. Our reported inference time uses a small network (128x128x4, tiny-cuda-nn [Mül21]) and all our hash grid encodings (section 3).*

4.3. Comparisons

Our main contribution is a way to render dynamic light fields in real-time. Our method is unique in the sense that it represents the

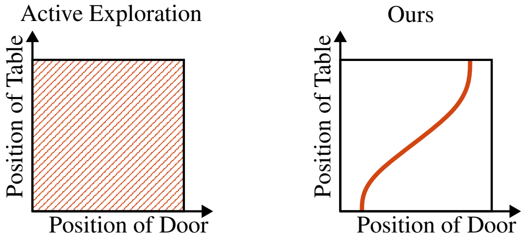


Figure 13: *The learning task of Active Exploration would be more complex as it learns all the permutations of the parameter space, we only learn one manifold in the parameter space.*

dynamic light field in an explicit spatio-temporal cache. We compare it to the two most closely related neural rendering methods, Active Exploration [DPD22] and a NeRF-style rendering method TiNeuVox [FYW*22]. Both methods are designed with dynamic scenes in mind, making them ideal for comparison.

4.3.1. Active Exploration

Since Active Exploration does not explicitly encode time, we reduce its input space to a manifold through all the possible parameter permutations for the objects in the scene. In the example in Figure 13, Active Exploration would learn the outgoing radiance for every combination of door and table position. Since we assume the scene dynamics to be known beforehand, we modify Active Exploration to simply learn one path through this permutation space to setup a fair comparison. Without this modification, the signal Active Exploration learns would be significantly more difficult than our setting. Concretely, we expose a single parameter t to Active Exploration, which determines the current transformation of scene objects.

Inference. Both Active Exploration and our method use a neural network to represent outgoing radiance and use Mitsuba [JSR*22] to create the required input buffers. While Active Exploration generates G-buffers, we use ray tracing for a potential first bounce on reflective surfaces. Note that our method would also work using only a G-buffer input, but at the cost of not being able to capture reflections. The results of the experiment below use ray tracing over the G-buffer option since it is the most robust setting. A G-Buffer pass in modern rendering systems takes less than 1 ms. Running our method on the scenes used in Active Exploration, network inference is about 20 times faster with our method, as seen in Table 2.

Table 3: *We report static, perceptual and video error metrics for the full resolution model as well as a model using half of the parameters. The low resolution grid can often achieve similar quality at ½ memory and ⅔ computation cost, in particular if the scene is simple.*

	Full Model			Low resolution Model		
Scene	MSE ↓	LPIPS ↓	VMAF ↑	MSE ↓	LPIPS ↓	VMAF ↑
Bunnies	9.9e-4	0.0840	82.44	1.1e-3	0.0909	80.71
Dyn. Rings	1.6e-3	0.0294	77.68	1.6e-3	0.0291	78.16
Rover	3.5e-4	0.0254	73.90	3.7e-4	0.0262	71.97
Cornell 180	9.5e-5	0.0168	94.74	9.6e-5	0.0172	94.81
Cornell 360	8.0e-5	0.0174	97.83	1.0e-4	0.0172	97.58
Cornell 720	1.0e-4	0.0179	99.59	1.1e-4	0.0176	99.54
Tilting Chair	5.7e-4	0.0318	84.23	6.1e-4	0.0338	82.73
Mirror Chair	1.2e-3	0.0556	80.97	1.4e-3	0.0558	79.92
Dining Slow	1.6e-3	0.0366	81.46	1.9e-3	0.0386	76.27
Dining Fast	2.9e-3	0.0521	70.25	3.1e-3	0.0539	65.03

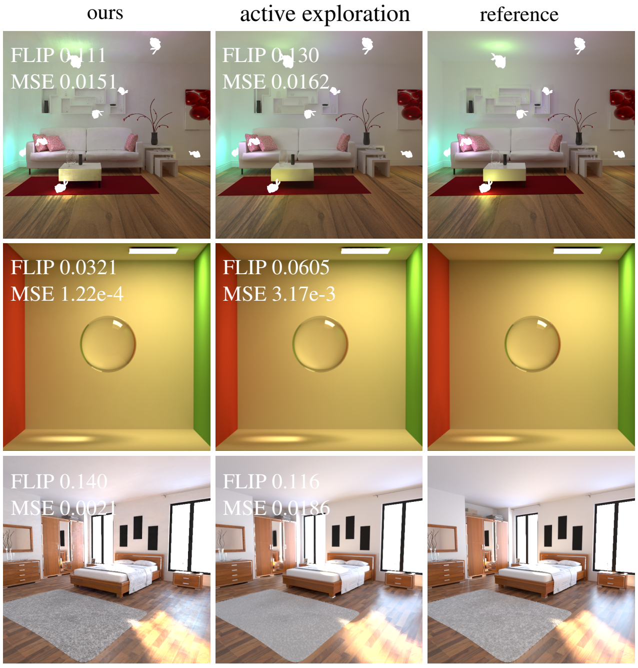


Figure 14: *Comparison between Active Exploration [DPD22] and our method on the Bunnies and their Sphere Caustic and Bedroom scene (3 hours of training). While Active Exploration tends to blur the radiance estimate, our method can exhibit higher frequency noise especially in the Bedroom scene due to the glossy floor. For both methods the respective artefacts reduce with a larger training budget.*

Quality. To compare the quality of our method and Active Exploration we train both of them for 3 hours. The results are found in Figure 14. Overall, the error metrics are similar, with the major difference that we are about 20 times faster during inference. Additional quality metrics for our approach can be found in Table 3. One interesting difference is the type of error on the radiance estimate is different for both methods. While our method tends to create high frequency artefacts, Active Exploration tends to blur the signal, which they also mention as a limitation. While quantitatively the error is similar, we can apply a filter or jitter the sampling of the hash grid to further reduce artefacts, as we show by the application of an exponential moving average on the temporal encodings in subsection 3.3. We urge the reader to look at the supplementary video to see the quality of our method. Table 3 shows very consistent low error rates for all tested scenes.

4.3.2. Dynamic NeRF methods

Recent work on reconstruction using neural radiance fields, have extended the training domain to incorporate a temporal dimension [PCPMMN20, FYW*22] and similarly to this work enhance

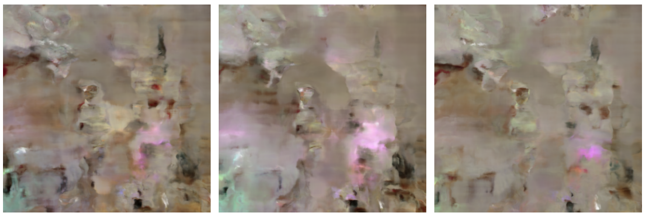


Figure 15: *TiNeuVox trained for 6 hours on the Bunnies scene, similar to the view in Figure 14.*

the neural representation with novel representations and encodings. Since neural radiance field methods need to reconstruct the scene geometry as well, we hypothesise that they will not work well on such a high frequency dynamic scene. Figure 15 shows the result of training on 150 frames of the *Bunnies* Animation. More recent works like K-Planes [FKMW*23] and HexPlane [CJ23] report similar quantitative results on synthetic dynamic data. HexPlane and K-Planes obtain similar results to TiNeuVox on the *Bunnies* scene with a PSNR around 10-15 dB.

5. Discussion and Conclusion

In this paper we introduced neural dynamic light fields, an efficient approach for spatio-temporal caching of radiance. Our approach to image synthesis is fast due to the use of a cache at the first (non-reflective) bounce and the use of a small neural network. Still, we achieve high quality due to the combination of several spatio-temporal encodings. Using a novel surface-space encoding, we improve the neural network's ability to learn shading that remains similar under rotation and translation. As such, our approach is an order of magnitude faster than previous work, opening the door for such caching to be used in real world applications.

Limitations We rely on a reasonable, bounded, non-overlapping UV mapping for our encoding. While most assets used in practice use textures of some form and thus provide at least a good starting point for the UV-map, the UV-maps may show overlaps or may rely on repeating texture coordinates. We currently use a simple texture map generation approach to generate novel UV mappings, as discussed in Appendix B. In general, we consider the generation of a low-distortion UV mapping a separate problem that could still increase our quality. Similarly, non-rigid objects and transformations may not offer a good UV-mapping. A more efficient UV-map generation and packing approach would reduce the memory budget of our UV hash grids [LVS18].

Using a small neural network for inference, comes with the downside that all information that is not contained in the encoding now has to be represented with less capacity. This can cause our method to show high frequency artefacts at lower training iterations, especially parts of the light field that are highly dependent on directions such as glossy materials. Supporting shading effects that are highly dependent on the direction—e.g. by encoding the direction vectors as well—without the need for a larger neural representation is an interesting research direction.

Obviously, our approach is limited to synthetic scenes that provide surfaces to sample on. Volumetric elements, such as fog or participating media—which can be represented with a neural radiance field—are not directly supported by our current method. However, we could also draw multiple samples inside of a participating media and learn from these volumetric scattering events. Alternatively, at the boundaries of volumes to reduce slot allocations in the hash grid.

All dynamics in our scenes are currently one-dimensional, i.e., one parameter completely controls all dynamics in the scene. An obvious choice of this parameter is time, which allows to represent arbitrary complex dynamics in a scene to, e.g., generate a high-quality 3D free-viewpoint video. As mentioned before other choices are capturing one specific animation or changing on material parameters. Although we did not go into details in the paper, initial experiments suggest that our main contribution—a novel combination of encodings to enable real-time dynamic light field rendering—still applies to higher dimensional input domains. Ideally, we would like to extend our method not only to a higher dimensional parameter space, but to novel objects. Currently, introducing a new object within the dynamic light field that was not observed during training would result in incorrect shading. Handling unseen geometry within our framework—for example, to render a dynamic avatar of a user—could be achieved using a separate shader pass that samples the dynamic light field.

Future work To extend our approach to multi-dimensional scene dynamics, we could follow an approach similar in spirit to how NeRFPlayer [SCL*23] categorises different types of temporal observations (static, dynamic, new observations). We could predict which encoding could be most useful to capture the dynamics for different parts of a complete scene. For example, non-moving objects may not benefit from a UV-encoding, areas that exhibit very homogeneous lighting conditions throughout all parameter variations can be encoded with only position. Having direct access to the scene description, may allow to make such decisions with high accuracy.

We believe an approach based on smarter sample allocation proportional to where neural networks struggles to capture the signal like in Diolatzis et al. [DPD22] could be a promising research direction. Tensor decomposition [SZT*23] techniques could also be applied to reduce the dimensionality of the hash grid.

Finally, we are looking forward to seeing whether UV-space encodings and explicitly representing the dynamic light field with a spatio-temporal cache, will be as transformative for other domains as they are for image synthesis. Our work is a step forward in caching complex global illumination in dynamic scenes, which enables many real-time applications, like architectural visualisation and free viewpoint movie experiences.

Acknowledgements

We thank Franz Scherr, Juan Gabriel Kostelec and Hector Garcia Rodriguez for their insightful machine learning related remarks. We are grateful to Blendswap user Wig42 for both the Living and Breakfast Room which we modified, Blendswap user vajrablue for the rover and Blendswap user barcin for the Chair scene which was modified.

References

[Bak16] BAKER D.: Object space lighting. GDC, 2016. 3

[BFM10] BURNS C. A., FATAHALIAN K., MARK W. R.: A lazy object-space shading architecture with decoupled sampling. In *Proc. High Performance Graphics* (2010), HPG '10, pp. 19–28. 3

[Bit16] BITTERLI B.: Rendering resources, 2016. https://benedikt-bitterli.me/resources/. 7, 13

[BWP*20] BITTERLI B., WYMAN C., PHARR M., SHIRLEY P., LEFOHN A., JAROSZ W.: Spatiotemporal reservoir resampling for real-time ray tracing with dynamic direct lighting. *ACM Transactions on Graphics (TOG) 39*, 4 (2020), 148–1. 2

[CJ23] CAO A., JOHNSON J.: Hexplane: A fast representation for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2023), pp. 130–141. 10

[Com18] COMMUNITY B. O.: *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. URL: http://www.blender.org. 13

[DK17] DAHM K., KELLER A.: Learning light transport the reinforced way. In *ACM SIGGRAPH 2017 Talks*. 2017, pp. 1–2. 4, 6

[DPD22] DIOLATZIS S., PHILIP J., DRETTAKIS G.: Active exploration for neural global illumination of variable scenes. *ACM Transactions on Graphics* (2022). 1, 2, 8, 9, 10, 13

[FKMW*23] FRIDOVICH-KEIL S., MEANTI G., WARBURG F. R., RECHT B., KANAZAWA A.: K-planes: Explicit radiance fields in space, time, and appearance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2023), pp. 12479–12488. 10

[FYW*22] FANG J., YI T., WANG X., XIE L., ZHANG X., LIU W., NIESSNER M., TIAN Q.: Fast dynamic radiance fields with time-aware neural voxels. In *SIGGRAPH Asia 2022 Conference Papers* (2022). 3, 9

[GBC16] GOODFELLOW I., BENGIO Y., COURVILLE A.: *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org. 4

[GCD*22] GUO X., CHEN G., DAI Y., YE X., SUN J., TAN X., DING E.: Neural deformable voxel grid for fast optimization of dynamic view synthesis. In *Proceedings of the Asian Conference on Computer Vision (ACCV)* (2022). 3

[GRPN20] GRANSKOG J., ROUSSELLE F., PAPAS M., NOVÁK J.: Compositional neural scene representations for shading inference. *ACM Transactions on Graphics (Proceedings of SIGGRAPH) 39*, 4 (July 2020). 2

[HCZ21] HADADAN S., CHEN S., ZWICKER M.: Neural radiosity. *ACM Transactions on Graphics (TOG) 40*, 6 (2021), 1–11. 1, 2, 3, 4, 7, 8

[HLN*23] HADADAN S., LIN G., NOVÁK J., ROUSSELLE F., ZWICKER M.: Inverse global illumination using a neural radiometric prior. In *ACM SIGGRAPH 2023 Conference Proceedings* (New York, NY, USA, 2023), SIGGRAPH '23, Association for Computing Machinery. doi:10.1145/3588432.3591553. 6

[HSS21] HLADKY J., SEIDEL H.-P., STEINBERGER M.: SnakeBinning: Efficient Temporally Coherent Triangle Packing for Shading Streaming. *Computer Graphics Forum* (2021). doi:10.1111/cgf.142648. 3

[HSV*22] HLADKY J., STENGEL M., VINING N., KERBL B., SEIDEL H.-P., STEINBERGER M.: Quadstream: A quad-based scene streaming architecture for novel viewpoint reconstruction. *ACM Trans. Graph. 41*, 6 (dec 2022). URL: http://doi.acm.org/10.1145/3550454.3555524, doi:10.1145/3550454.3555524. 3

[Hub64] HUBER P. J.: Robust Estimation of a Location Parameter. *The Annals of Mathematical Statistics 35*, 1 (1964), 73 – 101. URL: https://doi.org/10.1214/aoms/1177703732, doi:10.1214/aoms/1177703732. 12

[HY16] HILLESLAND K. E., YANG J. C.: Texel Shading. In *EG 2016 - Short Papers* (2016), Bashford-Rogers T., Santos L. P., (Eds.), The Eurographics Association. 3

[IRG*23] IŞIK M., RÜNZ M., GEORGOPOULOS M., KHAKHULIN T., STARCK J., AGAPITO L., NIESSNER M.: Humanrf: High-fidelity neural radiance fields for humans in motion. *ACM Transactions on Graphics (TOG) 42*, 4 (2023), 1–12. URL: https://doi.org/10.1145/3592415, doi:10.1145/3592415. 3

[JSR*22] JAKOB W., SPEIERER S., ROUSSEL N., NIMIER-DAVID M., VICINI D., ZELTNER T., NICOLET B., CRESPO M., LEROY V., ZHANG Z.: Mitsuba 3 renderer, 2022. https://mitsuba-renderer.org. 7, 9

[Kaj86] KAJIYA J. T.: The rendering equation. In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques* (1986), pp. 143–150. 2

[KAMJ05] KRISTENSEN A. W., AKENINE-MÖLLER T., JENSEN H. W.: Precomputed local radiance transfer for real-time lighting design. *ACM Trans. Graph. 24*, 3 (jul 2005). URL: https://doi.org/10.1145/1073204.1073334, doi:10.1145/1073204.1073334. 2

[KGPB05] KRIVÁNEK J., GAUTRON P., PATTANAIK S., BOUATOUCH K.: Radiance caching for efficient global illumination computation. *IEEE Transactions on Visualization and Computer Graphics 11*, 5 (2005), 550–561. 2

[KKLD23] KERBL B., KOPANAS G., LEIMKÜHLER T., DRETTAKIS G.: 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics 42*, 4 (July 2023). URL: https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/. 3

[KLA04] KAUTZ J., LEHTINEN J., AILA T.: Hemispherical Rasterization for Self-Shadowing of Dynamic Objects. In *Eurographics Workshop on Rendering* (2004), Keller A., Jensen H. W., (Eds.), The Eurographics Association. doi:10.2312/EGWR/EGSR04/179-184. 2

[LSZ*22] LI T., SLAVCHEVA M., ZOLLHOEFER M., GREEN S., LASSNER C., KIM C., SCHMIDT T., LOVEGROVE S., GOESELE M., NEWCOMBE R., ET AL.: Neural 3d video synthesis from multi-view video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2022), pp. 5521–5531. 3

[LVS18] LIMPER M., VINING N., SHEFFER A.: Boxcutter: Atlas refinement for efficient packing via void elimination. *ACM Transaction on Graphics 37*, 4 (2018). doi:10.1145/3197517.3201328. 10

[LW95] LAFORTUNE E. P., WILLEMS Y. D.: A 5d tree to reduce the variance of monte carlo ray tracing. In *Rendering Techniques' 95: Proceedings of the Eurographics Workshop in Dublin, Ireland, June 12–14, 1995 6* (1995), Springer, pp. 11–20. 2

[MESK22] MÜLLER T., EVANS A., SCHIED C., KELLER A.: Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph. 41*, 4 (July 2022), 102:1–102:15. URL: https://doi.org/10.1145/3528223.3530127, doi:10.1145/3528223.3530127. 2, 3, 8, 12

[MGNM19] MAJERCIK Z., GUERTIN J.-P., NOWROUZEZAHRAI D., MCGUIRE M.: Dynamic diffuse global illumination with ray-traced irradiance fields. *Journal of Computer Graphics Techniques (JCGT) 8*, 2 (June 2019), 1–30. URL: http://jcgt.org/published/0008/02/01/. 3

[MKS*15] MNIH V., KAVUKCUOGLU K., SILVER D., RUSU A. A., VENESS J., BELLEMARE M. G., GRAVES A., RIEDMILLER M., FIDJELAND A. K., OSTROVSKI G., ET AL.: Human-level control through deep reinforcement learning. *nature 518*, 7540 (2015), 529–533. 6

[MMK*21] MAJERCIK Z., MUELLER T., KELLER A., NOWROUZEZAHRAI D., MCGUIRE M.: Dynamic diffuse global illumination resampling. In *ACM SIGGRAPH 2021 Talks* (New York, NY, USA, 2021), SIGGRAPH '21, Association for Computing Machinery. URL: https://doi.org/10.1145/3450623.3464635, doi:10.1145/3450623.3464635. 3, 13

[MMR*19] MÜLLER T., MCWILLIAMS B., ROUSSELLE F., GROSS M., NOVÁK J.: Neural importance sampling. *ACM Trans. Graph. 38*, 5 (oct 2019). URL: https://doi.org/10.1145/3341156, doi:10.1145/3341156. 2

[MNV*21] MUELLER J. H., NEFF T., VOGLREITER P., STEINBERGER

M., Schmalstieg D.: Temporally adaptive shading reuse for real-time rendering and virtual reality. *ACM Trans. Graph. 40*, 2 (apr 2021). URL: https://doi.org/10.1145/3446790, doi:10.1145/3446790. 3, 5

[MRNK21] Müller T., Rousselle F., Novák J., Keller A.: Real-time neural radiance caching for path tracing. *ACM Trans. Graph. 40*, 4 (jul 2021). URL: https://doi.org/10.1145/3450626.3459812, doi:10.1145/3450626.3459812. 2, 3, 7, 12

[MST*20] Mildenhall B., Srinivasan P. P., Tancik M., Barron J. T., Ramamoorthi R., Ng R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV* (2020). 2, 3

[Mül21] Müller T.: tiny-cuda-nn, 4 2021. URL: https://github.com/NVlabs/tiny-cuda-nn. 2, 4, 8

[MVD*18] Mueller J. H., Voglreiter P., Dokter M., Neff T., Makar M., Steinberger M., Schmalstieg D.: Shading atlas streaming. *ACM Trans. Graph. 37*, 6 (Dec. 2018), 199:1–199:16. URL: http://doi.acm.org/10.1145/3272127.3275087, doi:10.1145/3272127.3275087. 3

[NMSS22] Neff T., Mueller J. H., Steinberger M., Schmalstieg D.: Meshlets and how to shade them: A study on texture-space shading. *Computer Graphics Forum 41*, 2 (2022), 277–287. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.14474, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.14474, doi:https://doi.org/10.1111/cgf.14474. 3

[PCPMMN20] Pumarola A., Corona E., Pons-Moll G., Moreno-Noguer F.: D-NeRF: Neural Radiance Fields for Dynamic Scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2020). 3, 9

[PGM*19] Paszke A., Gross S., Massa F., Lerer A., Bradbury J., Chanan G., Killeen T., Lin Z., Gimelshein N., Antiga L., Desmaison A., Kopf A., Yang E., DeVito Z., Raison M., Tejani A., Chilamkurthy S., Steiner B., Fang L., Bai J., Chintala S.: Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035. 12

[PSJ*23] Park S., Son M., Jang S., Ahn Y. C., Kim J.-Y., Kang N.: Temporal interpolation is all you need for dynamic neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2023), pp. 4212–4221. 3

[RBRD22] Rainer G., Bousseau A., Ritschel T., Drettakis G.: Neural precomputed radiance transfer. In *Computer Graphics Forum* (2022), vol. 41, Wiley Online Library, pp. 365–378. 2

[RWG*13] Ren P., Wang J., Gong M., Lin S., Tong X., Guo B.: Global illumination with radiance regression functions. *ACM Trans. Graph. 32*, 4 (jul 2013). URL: https://doi.org/10.1145/2461912.2462009, doi:10.1145/2461912.2462009. 2

[SB18] Sutton R. S., Barto A. G.: *Reinforcement learning: An introduction*. MIT press, 2018. 6

[SCL*23] Song L., Chen A., Li Z., Chen Z., Chen L., Yuan J., Xu Y., Geiger A.: Nerfplayer: A streamable dynamic scene representation with decomposed neural radiance fields. *IEEE Transactions on Visualization and Computer Graphics 29*, 5 (2023), 2732–2742. doi:10.1109/TVCG.2023.3247082. 3, 10

[SKS02] Sloan P.-P., Kautz J., Snyder J.: Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. *ACM Trans. Graph. 21*, 3 (jul 2002), 527–536. URL: https://doi.org/10.1145/566654.566612, doi:10.1145/566654.566612. 2

[SZT*23] Shao R., Zheng Z., Tu H., Liu B., Zhang H., Liu Y.: Tensor4d: Efficient neural 4d decomposition for high-fidelity dynamic reconstruction and rendering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2023). 3, 10

[TF17] Thomas M. M., Forbes A. G.: Deep illumination: Approximating dynamic global illumination with generative adversarial network. *arXiv preprint arXiv:1710.09834* (2017). 2

[Vea98] Veach E.: *Robust Monte Carlo methods for light transport simulation*. Stanford University, 1998. 2

[WRC88] Ward G. J., Rubinstein F. M., Clear R. D.: A ray tracing solution for diffuse interreflection. In *Proceedings of the 15th annual conference on Computer graphics and interactive techniques* (1988), pp. 85–92. 2

[WTS*23] Weinrauch A., Tatzgern W., Stadlbauer P., Crickx A., Hladky J., Coomans A., Winter M., Mueller J. H., Steinberger M.: Effect-based multi-viewer caching for cloud-native rendering. *ACM Trans. Graph. 42*, 4 (jul 2023). URL: https://doi.org/10.1145/3592431, doi:10.1145/3592431. 3

[YJM*23] Yu H., Julin J., Milacski Z. A., Niinuma K., Jeni L. A.: Dylin: Making light field networks dynamic. *arXiv preprint arXiv:2303.14243* (2023). 3

[ZBX*21] Zhu J., Bai Y., Xu Z., Bako S., Velázquez-Armendáriz E., Wang L., Sen P., Hašan M., Yan L.-Q.: Neural complex luminaires: Representation and rendering. *ACM Trans. Graph. 40*, 4 (jul 2021). URL: https://doi.org/10.1145/3450626.3459798, doi:10.1145/3450626.3459798. 2

[ZHM*23] Zheng C., Huo Y., Mo S., Zhong Z., Wu Z., Hua W., Wang R., Bao H.: Nelt: Object-oriented neural light transfer. *ACM Trans. Graph. 42*, 5 (aug 2023). URL: https://doi.org/10.1145/3596491, doi:10.1145/3596491. 2

[ZRW*23] Zeltner T., Rousselle F., Weidlich A., Clarberg P., Novák J., Bitterli B., Evans A., Davidovič T., Kallweit S., Lefohn A.: Real-time neural appearance models, 2023. arXiv:2305.02678. 2

Appendix A: Parameters

Network We use a fully fused neural network with 4 hidden layers of 128 neurons and ReLU as activation function, except for the output layer which has no activation function. The PyTorch [PGM*19] network is trained without bias vectors so that the weights can be quantized to 16-bit and transferred to the fast fully fused implementation of Müller et al. [MRNK21]. Together with the encoding we concatenate a mix of auxiliary parameters which further help the network distinguish possible surface properties and hash collisions. The parameters are listed in Table 4.

Variable	Dimension	Domain	Description
$\vec{x}$	$\mathbb{R}^3$	$[0,1]$	Position
$\vec{n}$	$\mathbb{R}^3$	$[-1,1]$	Normal
$\vec{w}$	$\mathbb{R}^3$	$[-1,1]$	Outgoing direction
α	$\mathbb{R}^3$	$[0,1]$	Diffuse reflectance
t	$\mathbb{R}$	$[0,1]$	Time of the animation
uv	$\mathbb{R}^2$	$[0,1]$	UV coordinates

Table 4: *Auxiliary inputs passed to the neural network in our implementation.*

The model is optimized with the Adam optimizer included with Pytorch and a learning rate of $5e-4$. We use a Huber Loss [Hub64] with $\delta = 1.0$.

Encoding For the multi resolution hash grid encodings we follow the recommended parameters of Müller et al. [MESK22] and their notation. We use a base resolution $N_{min} = 16$, a number of levels

$L = 16$ and a hashmap allocation budget of $T = 2^{19}$ with each level in the grid increasing in scale by 2 for the high resolution model and 4 for the low resolution model. The dimensionality of each latent vector F in the multi resolution hash grid is reported in Table 5. We note that for time, it is often not necessary to have such a high number of levels but given enough training time the high frequency artefacts disappear.

Encoding	Dimensions
H(x), t	$F_{H(x)} = 8$
H(x), H(t)	$F_{H(t)} = F_{H(x)} = 8$
H(t), H(x, t)	$F_{H(t)} = 2, F_{H(x,t)} = 8$
H(x), H(t), H(x, t)	$F_{H(x)} = F_{H(t)} = 2, F_{H(x,t)} = 4$
H(t), H(x), H(x, t), H(uv)	$F_{H(t)} = F_{H(x)} = F_{H(x,t)} = F_{H(uv)} = 2$

Table 5: *Number of feature dimensions per entry in the hash grids shown throughout the paper.*

Implementation Details To estimate the gradient of the residual (Section 3) we use $N = 2^{14}$ and $M = 32$ samples for all experiments. While varying these values can improve performance on certain scenes/animations we found these values to be sufficiently robust. Each one of the M hemispherical samples are drawn proportional to the BSDF and combined with M different light source samples through multiple importance sampling. One interesting direction of research would be to drive the sample selection with the neural representation, for example through resampling based on the incoming radiance estimate [MMK*21]. In practice we store our training samples in a buffer so that we can reuse them in later training iterations without the need of tracing rays, amortizing the cost of sampling the residual. For all the experiments in our work sample reuse was turned off.

Appendix B: Details of UV encoding

To generate the UV maps, we automatically process the standard dataset of Bitterli [Bit16] and unwrap the meshes with the "Smart UV Project" operator in Blender [Com18] with maximum angle limit set to 66%. The mapping is not guaranteed to be injective (free of overlaps), but upon manual inspection we found only a few meshes to have overlaps and that in general the majority of the UV maps had a good coverage of the domain. As described in appendix A, to further discriminate from possible collisions we input the global UV coordinates to the dynamic light field network. To pack the individual object UVs in a UV atlas, we pack each UV map along a grid proportional to their shape's surface area.

Appendix C: Area Sampling

In Figure 16 we show a scatter plot of the sample distribution in the *Bedroom* scene. Scenes that contain objects with strongly varying surface areas—like Figure 16—will benefit the most from area importance sampling. Interestingly, when the model struggles to capture highly glossy materials—which are often small in our scenes—uniformly sampling objects has a slight advantage in terms of convergence time due to the increase in sampling density where to model struggles to capture the signal. For example, in Figure 2 the quality of the golden vase is significantly lower due to the decreased training sample density. Taking the material complexity, or the training loss itself [DPD22], into account when placing samples would further reduce the training error. While the longer training of our method diminishes the artefacts on glossy materials, we consider efficiently representing glossy materials with a tiny neural network an open research problem.

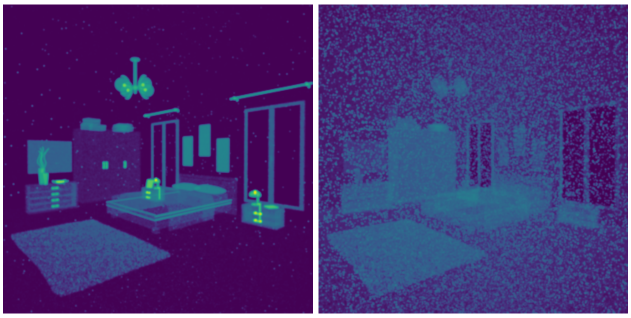

Uniform *Area*

Figure 16: *Heatmap of the difference of uniform sampling of objects and sampling based on their surface area. The gradient of the batch of samples will be significantly less biased with respect to the actual radiance distribution when using Area sampling.*