Real-Time Light Field Path Tracing

Markku Mäkitalo[1]*, Erwan Leria[1], Julius Ikkala[1], and Pekka Jääskeläinen[1]

[1]Tampere University, P.O. Box 553, 33014 Tampere, Finland
*Corresponding author, markku.makitalo@tuni.fi

Abstract. Light field rendering and displays are emerging technologies that produce more immersive visual 3D experiences than the conventional stereoscopic 3D technologies, as well as provide a more comfortable virtual or augmented reality (VR/AR) experience by mitigating the vergence–accommodation conflict. Path tracing photorealistic synthetic light fields in real time is extremely challenging, since it involves rendering a large amount of viewpoints for each frame. However, these viewpoints are often spatially very close to each other, especially in light field AR glasses or other near-eye light field displays. In this paper, we propose a practical real-time light field path tracing pipeline and demonstrate it by rendering a 6×6 grid of 720p viewpoints at 18 frames per second on a single GPU, through utilizing denoising filters and spatiotemporal sample reprojection. In addition, we discuss how the pipeline can be scaled to yield higher-quality results if more parallel computing resources are available. We also show that our approach can be used to simultaneously serve multiple clients with varying light field grid sizes, with the quality remaining constant across clients.

Keywords: ray tracing, image-based rendering, virtual / augmented reality, image processing

1 Introduction

Photorealistic rendering has a wide variety of applications ranging from entertainment, telepresence and other remote communications, medical industry, architecture, industrial design, and many other virtual and augmented reality applications. With steadily growing interest towards more interactivity and an ultra-realistic immersive user experience, there are significant challenges in rendering such content in real time with high resolutions and high frame rates. The currently available stereoscopic displays lack in providing all the necessary visual cues for an immersive 3D experience, and often cause visual discomfort due to the vergence–accommodation conflict [6, 7]. A light field can be used to describe a 3D scene from multiple viewing angles simultaneously, thus also providing more accurate visual cues. However, this comes at the cost of a vast increase in computational complexity, since we need to process data for all the different viewpoints.

Specifically, the light field can be characterized with a 5D vector function called the plenoptic function. Further, assuming it is in unobstructed space, it can be reduced to a 4D function, with individual viewpoints being different 2D slices of this 4D function [15]. These 2D slices, or perspective images, typically exhibit significant redundancy, as they are representing the same scene from slightly varying viewpoints. Hence, especially in real-time rendering applications, exploiting this redundancy is crucial in order to reduce the amount of data to be rendered. This can be done for example by reprojecting already rendered samples, both spatially between the viewpoints and temporally between frames.

Path tracing is a ubiquitously used algorithm for photorealistic rendering, which models how the light physically travels and interacts with the world. It is a progressive process, yielding higher-quality results as more samples (paths) per pixel are traced. This is computationally very expensive even for traditional monoscopic applications, and much more so with the challenging goal of photorealistic real-time light field rendering. Given the strict computational constraints of real-time rendering, we can currently afford to trace only a few samples per pixel (spp), even when rendering just a single viewpoint and using hardware-accelerated tracing.

In this paper, we propose and demonstrate a practical pipeline capable of real-time light field path tracing. Its basic principle is to path trace a subset of the pixels, and improve and synthesize the full data through an effective combination of denoising and spatiotemporal sample reprojection. In particular, we consider a light field represented by a grid of regularly spaced virtual cameras with small horizontal and vertical translations between the adjacent viewpoints, producing a grid of 2D perspective images. In our experiments, we focus on a 6×6 grid in order to match a prototype near-eye light field display; it is also close to what other similar devices, such as Huang et al. [8], use. In addition, we demonstrate that our approach can be used to simultaneously serve multiple clients with varying light field grid sizes, at a comparable quality for each client.

Our main contributions in this paper are:

- We propose a general pipeline that is suitable for real-time light field path tracing, and flexible for scaling the performance according to the available parallel resources.
- We demonstrate that a simple viewpoint-based practical implementation of the pipeline can render a light field in real time on a single GPU (a 6×6 grid of 720p viewpoints at 18 frames per second) with acceptable quality, by path tracing and denoising only a single full viewpoint in the middle of the grid, and using it to synthesize the other 35 viewpoints.
- We show that the rendered data can be used to simultaneously serve multiple light field clients, with the quality remaining constant across different grid sizes.

The rest of the paper is structured as follows: Section 2 reviews the related work on sample reprojection, denoising, and view synthesis. Section 3 introduces the proposed pipeline and discusses our specific implementation details. Section 4 describes our experiments, Section 5 presents the runtime and quality results obtained with our implementation, and finally, Section 6 concludes the paper.

2 Related work

In image-based rendering (IBR), a collection of 2D images is used in order to reconstruct the 3D scene or to synthesize novel viewpoints. This reuse of rendered data can be done even from a single 2D image, by warping, or reprojecting, the existing samples from one viewpoint onto another, based on the depth information and knowledge about the camera parameters and transformations between the two viewpoints. For natural images, these need to be estimated, but for synthetic rendering, the information is readily available, simplifying the process.

The IBR problem can also be modelled as a plenoptic sampling problem [18], hence considering the existing viewpoints as sparsely sampled data from the full (i.e., densely sampled) light field. Then, reconstructing the densely sampled light field, also enabling novel view synthesis, is often approached in the epipolar plane image (EPI) domain [23] or in Fourier domain [21]. Due to anisotropy in the light field, samples can also be reused to reconstruct a more dense temporal or indirect light field from a sparse input, thus increasing the effective sampling rate significantly [13, 14]. However, these approaches are generally targeting high-quality reconstruction, and are currently not suitable for real-time view synthesis.

On the other hand, sample reprojection is straightforward and computationally cheap, so it is commonly used even in monoscopic real-time rendering applications in order to utilize the temporal coherence between successive frames. For example, most of the state-of-the-art real-time and interactive path tracing denoising filters, such as [19, 20, 11, 17, 10], incorporate temporal reprojection in their reconstruction pipeline. Out of these denoising filters, we identify BMFR [11] and SVGF [19] as particularly suitable candidates for improving the quality of real-time light field path tracing: it takes only about 1 ms to denoise a single 1280×720 pixel viewpoint with either of the two filters.

Besides temporal reprojection, in stereoscopic rendering the samples can also be reprojected between the two viewpoints, and this approach can be further generalized for multi-view rendering. In particular, Andersson et al. [1] accelerate multi-view ray tracing through multi-dimensional adaptive sampling and edge detection. Fraboni et al. [3] accelerate multi-view path tracing by progressively generating the traced paths in a way that facilitates sharing a significant portion of their contributions between different viewpoints. Nevertheless, neither of these multi-view methods target real-time applications, and are rather costly to compute. A light field display, such as the one by Lanman and Luebke [12] or Huang et al. [8], typically requires dozens of viewpoints to be rendered for a single frame, so the benefit of reprojection quickly becomes very significant, if only a few viewpoints need to be rendered and the rest can be synthesized based on the existing data. This is crucial especially for multi-view applications striving for real-time rendering.

Neural networks have also been employed for multi-view reprojection [2, 5, 25]. However, they consider the reprojection part to be straightforward enough, so that it is left to be done outside of the network; the networks only perform the final blending of the novel viewpoint, given the multiple existing viewpoint inputs. They produce high-quality results, but are computationally too expensive for real-time frame rates or require hours of preprocessing.

Sample reprojection is used by Hansen et al. [4] for accelerating rasterized light field rendering for head-mounted displays. They consider a 15×8 grid of perspective viewpoints, in which they only render the four corner viewpoints, and reproject all the remaining viewpoints. Although they reach an interactive performance of about 5 FPS, filling the missing data in places where the reprojection fails is difficult with their method. In contrast, with path tracing, the missing pixels can easily be traced after the reprojection step.

Mäkitalo et al. [16] present a systematic evaluation of the quality of reprojected path traced samples for the stereoscopic case. They report that spatiotemporal reprojection brings an almost 25-fold quality increase for 1 spp input data in terms of the effective spp, with having to path trace only 2–6 % of the samples in the target frame. We are not aware of such work existing for light fields.

Finally, Xie et al. [24] demonstrate production-quality real-time cluster path tracing rendering that scales linearly with the number of RTX cluster nodes used. However, they focus on single-view rendering and increasing the spp count by adding more nodes, instead of attempting to reduce the computational effort through reusing data.

3 General pipeline and our implementation

3.1 General pipeline

Figure 1 visualizes the proposed general pipeline for real-time light field path tracing. First, a number of independent *source regions* are path traced, where the number of regions can be decided based on the amount of parallel resources available for real-time path tracing; the figure uses two regions for the purposes of visualization. These regions can correspond to the viewpoints in the light field grid, but they can also be smaller or larger regions. They can even be sparse discontinuous sets of pixels, although that will likely present additional challenges for the denoising filters, for optimizing the memory accesses, and for data transfers between different computing nodes.

Next, temporal reprojection is applied for each path traced source region, in order to blend the current data for the region with the corresponding data in the previous frame, thus increasing the effective quality.

Alternative steps in the pipeline are marked with dashed blue and red routes. In particular, the denoising filter can be applied either to the path traced source regions (regions 1 and 2 in the example graph) after temporal blending, or at a later stage after all the missing *target regions* ($3..n$) have been synthesized through spatial reprojection. The former option (shown with the blue route) is particularly useful if the amount of parallel resources is limited, as the denoising improvement can be propagated from the few source regions into all the target regions without expending too much computational effort on the denoising. On the other hand, the latter option (red route) likely produces a higher-quality

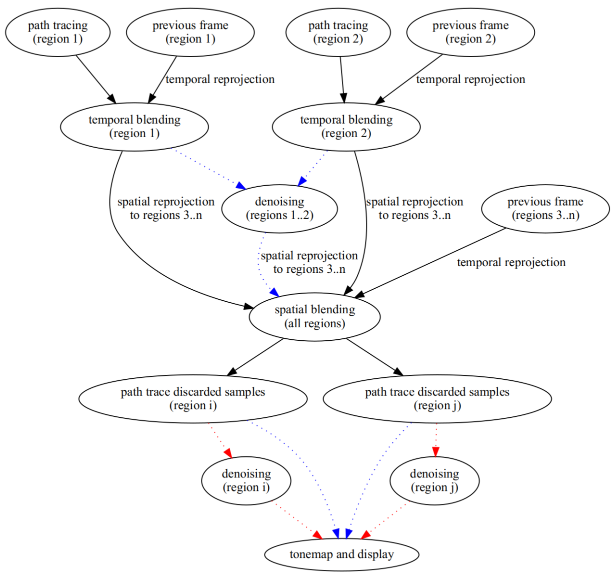

path tracing (region 1)
previous frame (region 1)
path tracing (region 2)
previous frame (region 2)
temporal reprojection
temporal reprojection
temporal blending (region 1)
temporal blending (region 2)
spatial reprojection to regions 3..n
denoising (regions 1..2)
spatial reprojection to regions 3..n
previous frame (regions 3..n)
spatial reprojection to regions 3..n
temporal reprojection
spatial blending (all regions)
path trace discarded samples (region i)
path trace discarded samples (region j)
denoising (region i)
denoising (region j)
tonemap and display

Fig. 1: A high-level task graph of the proposed general pipeline. The blue route can be replaced by the red route, if there are enough resources to denoise all regions in parallel in real time. The choice of two independently path traced regions is only for the visualization; the number is not dictated by the pipeline.

result. However, it requires enough resources to denoise all n regions in parallel in real time, and ensuring that the separately denoised regions do not cause distracting artifacts at the region boundaries; irregularly shaped regions or additional lightweight deblocking filters could be used to alleviate that. In either case, the temporally blended source regions (denoised or not), are used to synthesize the remaining regions $3..n$ through spatial reprojection.

Backward reprojection is commonly used for both temporal and spatial reprojection, starting from the target region, and finding for each pixel, which pixel in the source region maps to it. Not all pixels can be reprojected, due to data being out of the viewing frustum or occluded in one of the regions, for example. Having multiple source regions in different parts of the light field grid brings better coverage for those areas, but at the cost of having to path trace more source regions, which is expensive.

The discarded pixels in the synthesized target regions now need to be filled by path tracing. As this step does not depend on the previously chosen region division, a different set of regions can be used for the remainder of the pipeline (regions i and j in the graph), if it is deemed more optimal in terms of the currently available parallel resources. If there were a lot of discarded pixels, denoising after the missing pixels have been path traced (red route) will reduce the visible differences between the reprojected higher-quality areas and the newly path traced areas. Finally, all the regions will be combined for tonemapping and displaying the result.

3.2 Our implementation

For this paper, we implement a version of the pipeline that targets real-time light field path tracing on a single GPU. More specifically, we assume for simplicity that each region corresponds to a full viewpoint in the light field grid, and that denoising is applied only on the source viewpoint(s) before spatial reprojection, not on the target viewpoints. In other words, we adhere to the blue route shown in Figure 1.

We use backward reprojection in all of our experiments. For each sample, we look at the difference in both world positions and shading normals of the source and target pixel locations, and the instance IDs of the objects. The sample is reprojected to the target location if the instance IDs match and both differences stay below specified thresholds. The thresholds are chosen manually for each scene based on the scale of the scene. If either threshold is exceeded, or if the instance IDs do not match, or if the target pixel is otherwise not visible in the source viewpoint (out-of-frustum data), we discard that pixel. The discarded pixels will then be path traced after spatial reprojection. Note that the reprojection requires us to render the G-buffers for the non-rendered target viewpoints as well. This comes with a modest additional cost, as our runtime experiments will indicate. Furthermore, we specifically opt to use a relatively simple reprojection strategy in order to demonstrate the feasibility of reprojection in general as a part of a real-time light field path tracing pipeline; it utilizes only the world positions, shading normals, and instance IDs, but is very fast to evaluate. However, more accurate reprojection could be achieved by leveraging more information, such as material properties, or anisotropy in the light field.

The reprojection in general is not a 1:1 pixel mapping, so it involves a resampling step in order to pick the closest pixel(s) from the source viewpoint to reproject. For example, using bilinear interpolation between the closest existing samples acts as a simple blurring kernel, which improves the quality of the noisy data. Moreover, when multiple source viewpoints have candidate pixels for the reprojection, their contributions can be combined, blending the final pixel value. In such cases, in order to minimize the computational overhead, we simply pick the source pixel with the smallest depth value, instead of blending multiple values together.

In addition, temporal reprojection usually includes accumulation, which increases temporal stability and the effective spp count. We accumulate the pixel data temporally with a blending factor of 0.2, as is commonly done [19, 11].

Our implementation considers only fully path traced individual viewpoints. However, the proposed pipeline also allows for sparsely and adaptively distributed samples irrespective of the viewpoint boundaries, although that is likely to present additional challenges both for the denoising algorithms and for distributing the computations. Specifically, denoising algorithms typically rely on spatially neighbouring samples to be present, which would not necessarily be the case with sparse sampling. For example BMFR operates in a blockwise fashion, and SVGF estimates the variance in a spatial neighbourhood when only one sample per pixel is available.

4 Experiments

We consider four test scenes: Sponza, Eternal Valley, Warehouse, and Bistro Interior. Each scene consists of 30 frames, each frame having a 6×6 grid of 1280×720 pixel viewpoints. The camera paths range from a simple forward dolly tracking in Sponza to a very rapid nonlinear fly-through in Bistro Interior. The difference between adjacent viewpoints is approximately 7 pixels for Sponza, 9 pixels for Eternal Valley, and 8 pixels for both Warehouse and Bistro Interior. An example 4096 spp reference frame for each scene is shown in Figure 2.

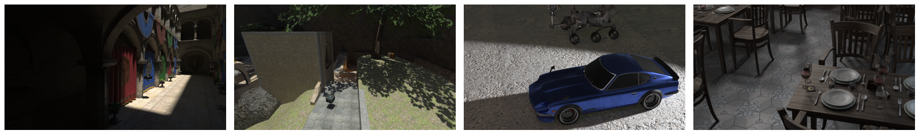


Fig. 2: Example 4096 spp reference viewpoints of the datasets used in the experiments. Each frame has a 6×6 grid of 1280×720 pixel viewpoints. From left to right: Sponza, Eternal Valley, Warehouse, Bistro Interior.

In Section 5.1 (**"Runtime"**), we measure the real-time performance of our implementation described in Section 3.2, both as a whole and by looking at the individual stages of the pipeline. Our implementation is built on top of our own Vulkan-based real-time path tracer, running on a high-end desktop with an NVIDIA RTX 3090 GPU. By default, we path trace 1 spp both for the source viewpoint(s) and when filling the discarded pixels later, using a maximum of 8 bounces, and *next event estimation* (NEE) on.

In order to identify the most suitable source viewpoint configuration for single-GPU rendering, we also look at the runtimes and spatial discard percentages for five different configurations with 1–4 path traced source viewpoints; each additional path traced viewpoint requires a significant amount of additional computations, but can improve the reprojection, as the scene is covered from multiple angles. Specifically, we have two single-camera setups, with the rendered viewpoint either in the middle of the 6×6 grid, or at the top left corner. In our dual-camera setup we render the top left and bottom right viewpoints; in the three-camera setup, the rendered viewpoints form a triangular shape; and in the four-camera setup, all four corner viewpoints are rendered. These locations are depicted with camera icons in Figure 5 (Appendix A).

In Section 5.2 (**"Quality"**), we evaluate the quality of the rendering, focusing on evaluating the most efficient setup identified in the previous experiment, which turns out to be to path trace only the middle viewpoint. That is, we start with a 1 spp path traced input in the middle viewpoint, denoise it, and then spatiotemporally reproject it onto the other 35 viewpoints. For the denoising filter, we use a state-of-the-art real-time filter, specifically either BMFR [11] or SVGF [19]. By denoising before reprojection, we aim to propagate the benefits of the denoising filter onto all viewpoints without incurring the extra computational cost of having to denoise all viewpoints. BMFR and SVGF were chosen for their good balance between speed and quality: it takes only about 1 ms to denoise a single 1280×720 pixel viewpoint with either of them. We measure the average results in terms of root mean squared error (RMSE) and SSIM [22] against the 4096 spp reference, both for spatiotemporal reprojection on its own, and for the combination of denoising and reprojection.

Finally, we consider a scenario where a rendering server is providing data for multiple light field clients, whose display configurations may vary. For example, the clients' display grids may vary between 3×3 and 6×6 viewpoints. However, with the proposed configuration, the path traced middle viewpoint can be used as the basis for all clients, with the amount of reprojected viewpoints around it determined by the client. Depending on the rendering capabilities and data transfer requirements, the server can then perform the reprojection and send the desired subset of viewpoints to a client; or if the client is also able to apply reprojection and do lightweight path tracing for the samples discarded by the reprojection algorithm, the server can only transfer the path traced middle viewpoint. To corroborate the validity of the multi-client approach, we evaluate the quality after denoising and reprojection as in the earlier quality experiments, but now for grid sizes of 3×3, 4×4, 5×5, and 6×6. A representative selection of these results is shown at the end of Section 5.2.

5 Results

This section presents our results for the runtime and quality experiments described in Section 4.

5.1 Runtime

The runtime differences of spatial reprojection for the five viewpoint configurations are fairly minor: Spatial reprojection takes 0.07–0.08 ms per target viewpoint for both single-camera setups, and 0.09–0.14 ms for the multi-camera setups, with the average time being close to the lower end of the range in each case. Hence, spatial reprojection to 35 viewpoints, for example with the path traced middle viewpoint, takes about 2.45 ms per frame in total. Temporal reprojection also takes 0.07–0.08 ms per viewpoint, resulting in about 2.52 ms per frame when reprojecting all 36 viewpoints.

Figure 3 shows the execution time of the different pipeline stages when rendering Sponza. Generating the G-buffers takes 0.28 ms per viewpoint, so about 10 ms in total for the 36 viewpoints. Path tracing the middle viewpoint of a 6×6 grid of 720p viewpoints (1 spp, 8 bounces, next event estimation on) takes 22 ms.

Then, that viewpoint is denoised with BMFR or SVGF, taking about 1 ms. After the temporal and spatial reprojection stages, which take about 2.5 ms each as described above, path tracing is used to fill in the discards in the 35 target viewpoints, in this case 2.08 % of the pixels (see Table 1). This takes 16 ms, yielding a total execution time of 54 ms, or about 18 FPS. With the other viewpoint configurations, the corresponding numbers are about 16 FPS (path traced top left viewpoint), 15 FPS (top left + bottom right), 11 FPS (triangle), and 9 FPS (corners).

Appendix A further illustrates the relationship of the viewpoint-wise average discard percentages and the camera locations in different configurations. In summary, we conclude that in terms of our target of achieving real-time frame rates, rendering a single viewpoint in the middle of the grid offers the best trade-off between the reprojection benefits and the amount of path tracing required. Hence, in the quality experiments, we focus on evaluating the middle-viewpoint single-camera configuration.

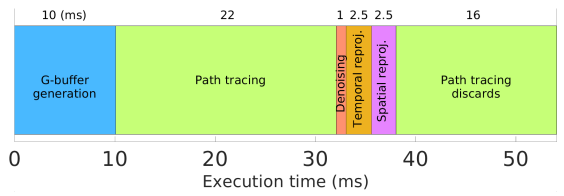


Fig. 3: Execution time breakdown of our implementation, rendering the Sponza light field (6×6 grid of 720p viewpoints) on RTX 3090 in 54 ms.

Overall, path tracing dominates the total execution time. This further validates our choice of minimizing the number of path traced viewpoints, as our implementation aims to achieve real-time frame rates. Specifically, NEE takes up most of the path tracing time. Disabling it reduces the path tracing time per viewpoint from 22 ms to 6 ms, so it would boost the performance from 18 FPS to 36 FPS, albeit yielding a somewhat lower visual quality. Similarly, the number of bounces could be reduced at the cost of sampling complex light transport less accurately: for example, with 3 bounces instead of 8, path tracing a viewpoint takes 7.5 ms instead of 22 ms, yielding 33 FPS (with NEE still on).

5.2 Quality

The RMSE and SSIM results for all four test scenes are presented in Table 1. Each value is an average taken over all 30 frames and over all the viewpoints. First, we see that even when spatiotemporal reprojection is applied on its own without a denoising filter before it, the quality increase compared to the 1 spp input is significant. However, the combination of denoising and reprojection is able to provide much larger overall improvements, especially on the more difficult Eternal Valley, Warehouse and Bistro Interior scenes.

On the other hand, a large area of Sponza is under direct illumination, so the 1 spp input is already of relatively high quality. Interestingly, the RMSE improvements obtained for Sponza with a denoiser included are smaller than those obtained with the spatiotemporal reprojection on its own. We suspect this is due to the denoising filters causing some additional blurring, which RMSE appears to consider unfavourable, as opposed to SSIM. This scene-specific deficiency could be addressed by fine-tuning the temporal accumulation parameters, or by using a denoising filter with more advanced adaptive capabilities, such as A-SVGF [20].

Table 1 also shows the spatiotemporal discard percentages for each scene, averaged over all frames and viewpoints. For Sponza, Eternal Valley and Warehouse, only 2.1–3.7 % of the samples in the 35 synthesized viewpoints need to be filled by path tracing. In Bistro Interior, the discard percentage is much higher at 11.5 %, but this is expected due to the rapid camera motion and a large amount of disocclusions, especially near the chairs, tables and glasses. Hence, the RMSE and SSIM results are also clearly lower than in the other scenes, as the larger disoccluded areas do not benefit from reprojection or temporal accumulation.

Example result frames are presented in Appendix B, where the subjective visual quality and temporal stability are also discussed. Moreover, example results for all scenes are presented in the supplementary video.

Table 1: RMSE and SSIM of the denoised and spatiotemporally reprojected frames compared to the 4096 spp reference, averaged over 30 frames and over all the viewpoints. Similarly averaged spatiotemporal reprojection discard percentages are also presented.

		Discard %	**RMSE**	**SSIM**
Sponza	1 spp input	2.08	0.1014	0.2631
	Spatiotemporally reprojected		0.0393	0.6384
	BMFR + reprojected		0.0512	0.7953
	SVGF + reprojected		0.0425	0.8046
Eternal Valley	1 spp input	3.69	0.2732	0.0704
	Spatiotemporally reprojected		0.1447	0.2960
	BMFR + reprojected		0.0756	0.6002
	SVGF + reprojected		0.0913	0.5728
Warehouse	1 spp input	2.30	0.1546	0.1520
	Spatiotemporally reprojected		0.1196	0.2489
	BMFR + reprojected		0.0582	0.7595
	SVGF + reprojected		0.0671	0.6671
Bistro Interior	1 spp input	11.53	0.2519	0.0435
	Spatiotemporally reprojected		0.2155	0.1073
	BMFR + reprojected		0.1252	0.4905
	SVGF + reprojected		0.1468	0.3904

For the multiple-client setup with grid sizes of 3×3, 4×4, 5×5, and 6×6, the SSIM results after denoising and spatiotemporally reprojecting the 1 spp input are illustrated in Figure 4 for Sponza and Eternal Valley. For both scenes, the quality is fairly constant for all grid sizes. Since the discard percentages for the viewpoints furthest away from the middle are higher in Eternal Valley than in Sponza, there is a slight drop in SSIM for the largest grid sizes, but the effect is still very minor. These results indicate that the described setup can be used to serve different light field displays with comparable quality across all of them.

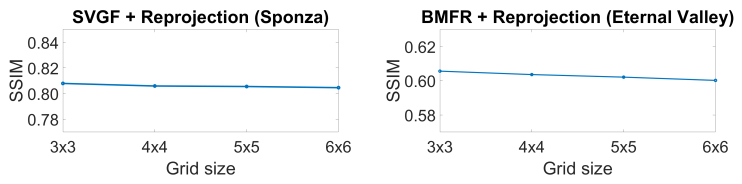


Fig. 4: SSIM of the denoised and spatiotemporally reprojected 1 spp frames for varying grid sizes.

6 Conclusions

We proposed a real-time pipeline for photorealistic light field path tracing, and demonstrated it with a practical implementation that renders a 6×6 grid of 1280×720 viewpoints at 18 frames per second on a single GPU. The real-time performance is achieved by path tracing a viewpoint in the middle of the grid at 1 spp, denoising it, and constructing the 35 other viewpoints around it through spatiotemporal reprojection. We also showed that this pipeline could be used for simultaneously serving multiple light field clients with different display grid sizes, with the quality remaining constant across clients.

Our implementation works best for scenes with slow or moderate movement; rendering fast-paced content proves more challenging, due to the compromises made in order to achieve real-time performance on a single GPU. However, if more distributed resources are available, all viewpoints can be denoised in parallel after reprojection, instead of only denoising the path traced viewpoint(s) before reprojection. With more resources available, path tracing multiple viewpoints also becomes a more feasible option, and is expected to improve the quality for fast-paced content. The additional resources could also be used to improve the input spp, in which case more lightweight denoising may be sufficient.

Acknowledgements

This project is supported by the Academy of Finland under Grant 325530.

References

1. Andersson, M., Johnsson, B., Munkberg, J., Clarberg, P., Hasselgren, J., Akenine-Möller, T.: Efficient multi-view ray tracing using edge detection and shader reuse. The Visual Computer 27(6–8) (2011)
2. Flynn, J., Neulander, I., Philbin, J., Snavely, N.: Deepstereo: Learning to predict new views from the world's imagery. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (2016)
3. Fraboni, B., Iehl, J.C., Nivoliers, V., Bouchard, G.: Adaptive multi-view path tracing. In: Proc. 30th Eurographics Symposium on Rendering (2019)
4. Hansen, A.J., Klein, J., Kraus, M.: Light field rendering for head mounted displays using pixel reprojection. In: VISIGRAPP (1: GRAPP) (2017)

5. Hedman, P., Philip, J., Price, T., Frahm, J.M., Drettakis, G., Brostow, G.: Deep blending for free-viewpoint image-based rendering. ACM Trans. Graph. 37(6) (2018)
6. Hoffman, D.M., Girshick, A.R., Akeley, K., Banks, M.S.: Vergence–accommodation conflicts hinder visual performance and cause visual fatigue. Journal of Vision 8(3) (2008)
7. Hua, H.: Enabling focus cues in head-mounted displays. Proc. IEEE 105(5) (2017)
8. Huang, F.C., Luebke, D., Wetzstein, G.: The Light Field Stereoscope. In: ACM SIGGRAPH 2015 Emerging Technologies. ACM, New York, NY, USA (2015)
9. Huo, Y., Yoon, S.e.: A survey on deep learning-based monte carlo denoising. Computational Visual Media 7(2), 169–185 (2021)
10. Işik, M., Mullia, K., Fisher, M., Eisenmann, J., Gharbi, M.: Interactive Monte Carlo denoising using affinity of neural features. ACM Trans. Graph. 40(4) (2021)
11. Koskela, M., Immonen, K., Mäkitalo, M., Foi, A., Viitanen, T., Jääskeläinen, P., Kultala, H., Takala, J.: Blockwise multi-order feature regression for real-time path-tracing reconstruction. ACM Trans. Graph. 38(5) (2019)
12. Lanman, D., Luebke, D.: Near-eye light field displays. ACM Trans. Graph. 32(6) (2013)
13. Lehtinen, J., Aila, T., Chen, J., Laine, S., Durand, F.: Temporal light field reconstruction for rendering distribution effects. In: ACM SIGGRAPH 2011 papers (2011)
14. Lehtinen, J., Aila, T., Laine, S., Durand, F.: Reconstructing the indirect light field for global illumination. ACM Trans. Graph. 31(4) (2012)
15. Levoy, M., Hanrahan, P.: Light field rendering. In: Proc. 23rd Annual Conference on Computer Graphics and Interactive Techniques (1996)
16. Mäkitalo, M.J., Kivi, P.E., Jääskeläinen, P.O.: Systematic evaluation of the quality benefits of spatiotemporal sample reprojection in real-time stereoscopic path tracing. IEEE Access 8 (2020)
17. Meng, X., Zheng, Q., Varshney, A., Singh, G., Zwicker, M.: Real-time Monte Carlo denoising with the neural bilateral grid. In: EGSR (DL) (2020)
18. Pearson, J., Brookes, M., Dragotti, P.L.: Plenoptic layer-based modeling for image based rendering. IEEE Transactions on Image Processing 22(9) (2013)
19. Schied, C., Kaplanyan, A., Wyman, C., Patney, A., Chaitanya, C.R.A., Burgess, J., Liu, S., Dachsbacher, C., Lefohn, A., Salvi, M.: Spatiotemporal variance-guided filtering: Real-time reconstruction for path-traced global illumination. In: Proc. High Performance Graphics (2017)
20. Schied, C., Peters, C., Dachsbacher, C.: Gradient estimation for real-time adaptive temporal filtering. Proc. ACM on Computer Graphics and Interactive Techniques 1(2) (2018)
21. Shi, L., Hassanieh, H., Davis, A., Katabi, D., Durand, F.: Light field reconstruction using sparsity in the continuous Fourier domain. ACM Trans. Graph. 34(1) (2014)
22. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: From error visibility to structural similarity. IEEE Trans. Image Process. 13(4) (2004)
23. Wu, G., Zhao, M., Wang, L., Dai, Q., Chai, T., Liu, Y.: Light field reconstruction using deep convolutional network on EPI. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (2017)
24. Xie, F., Mishchuk, P., Hunt, W.: Real time cluster path tracing. In: SIGGRAPH Asia 2021 Technical Communications (2021)
25. Xu, Z., Bi, S., Sunkavalli, K., Hadap, S., Su, H., Ramamoorthi, R.: Deep view synthesis from sparse photometric images. ACM Trans. Graph. 38(4) (2019)

Appendix A: Discard percentages

Figure 5 illustrates the average discard percentages per viewpoint after spatial reprojection in Sponza, for the five tested viewpoint configurations. Overall, the average discard percentages confirm the intuition that rendering a middle viewpoint is slightly better in terms of spatially discarded pixels than rendering the top left viewpoint, specifically for the furthest reprojected viewpoints in the corners. Adding a second rendered viewpoint clearly lowers the amount of discarded pixels compared to the single-camera setups. However, path tracing more viewpoints means much more computational effort compared to rendering only a single viewpoint, so Figure 5 indicates that the reduction in the discard percentages is simply not enough to justify path tracing an additional viewpoint when striving for real-time performance.

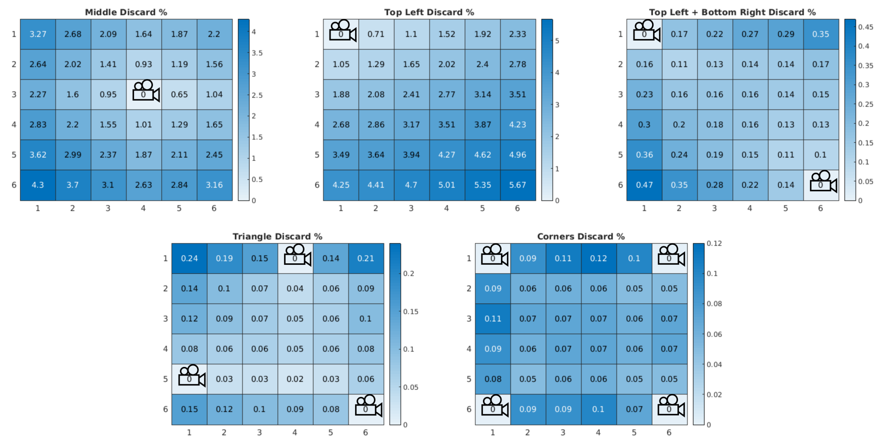


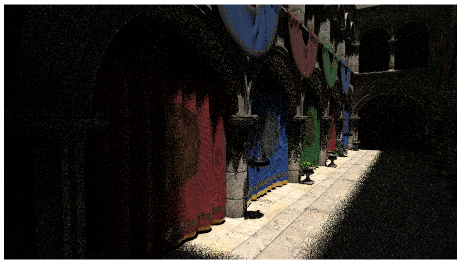
Fig. 5: Viewpoint-wise spatial discard percentage heatmaps for Sponza, for the different tested configurations. The path traced viewpoints are marked with camera icons. Note that each heatmap has a different range.

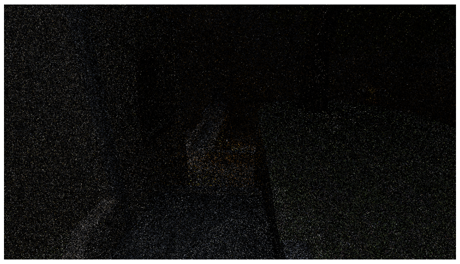
Appendix B: Comparison images

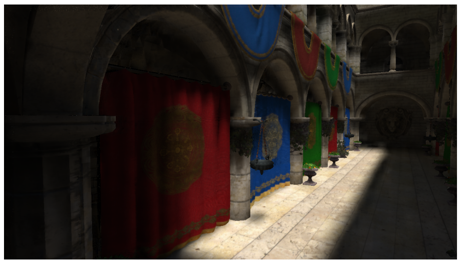
Figures 6–7 illustrate the results for all scenes, showing the original 1 spp input for the middle viewpoint, the denoised + reprojected results for the synthesized top left viewpoint, and the reference 4096 spp top left viewpoint. Overall, the visual quality is high enough to be usable in various real-time applications; the most visible denoising artifacts are the oversmoothing of shadows, and a moderate amount of blur. Moreover, the blockwise nature of BMFR can still be seen in the Eternal Valley result (Figure 6d), as temporal accumulation has not yet had time to fully compensate for it. Similarly, temporal accumulation has not yet removed some of the residual noise in Figures 7c–7f.

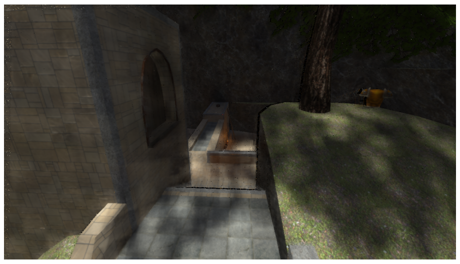
In general, narrow strips along the top and left borders exhibit a higher amount of noise, as they were not visible in the middle viewpoint, and thus did not benefit from the denoising. The effect of discards due to the fast camera motion is also evident in the disoccluded areas of Bistro Interior (Figures 7d and 7f). These are further visualized in Figure 8, which shows a selection of the results in Figures 6–7 before the discarded areas have been filled in, and with the discards shown in red. These artifacts could be mitigated for example by path tracing more samples at the affected locations, or by leveraging the denoise-after-reprojection pipeline option if parallel resources are available. Applying deep learning based temporal rendering techniques (see [9]) can also be an attractive option in that case.

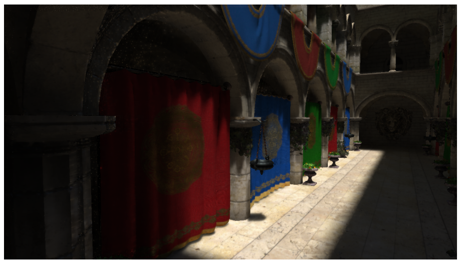
The temporal quality of the results can be seen in the supplementary video, which features all four test scenes. Overall, the temporal behaviour is relatively stable, thanks to the temporal accumulation. However, the accumulation does also involve a tradeoff between introducing ghosting (too much reuse) and not being able to suppress fireflies (too little reuse), as highlighted in the video for Eternal Valley. As for the quality in the disoccluded areas, the single-frame examples discussed above are also representative of the full video sequences.

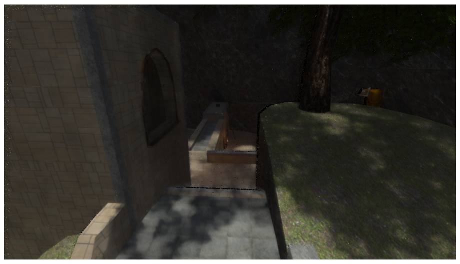

(a) 1 spp input (middle viewpoint)

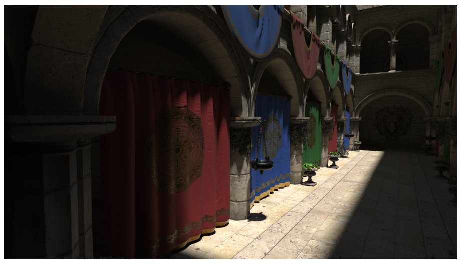

(b) 1 spp input (middle viewpoint)


(c) BMFR + reprojected to top left


(d) BMFR + reprojected to top left


(e) SVGF + reprojected to top left


(f) SVGF + reprojected to top left


(g) 4096 spp reference (top left viewpoint)

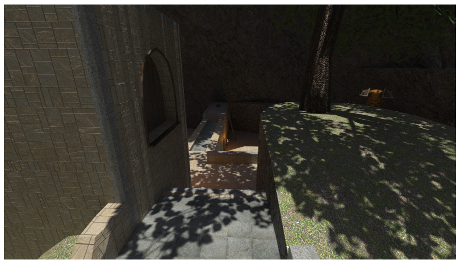
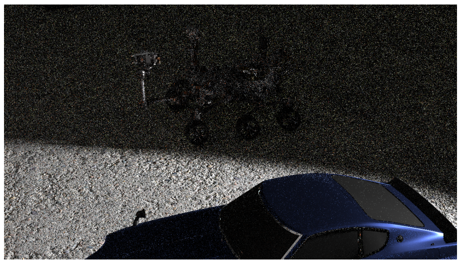
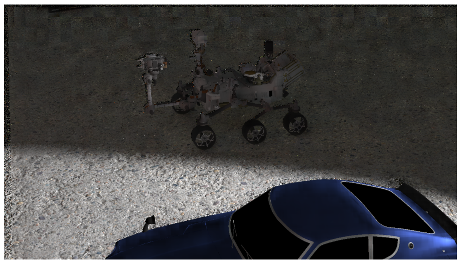
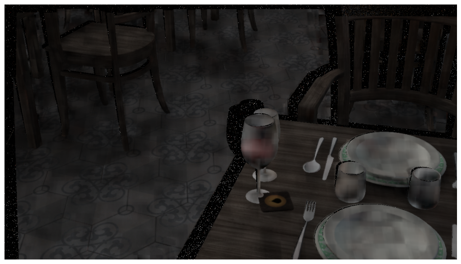

(h) 4096 spp reference (top left viewpoint)

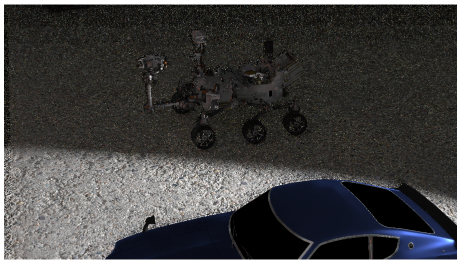
Fig. 6: Results for frame 15 of Sponza (left column) and Eternal Valley (right column) after denoising and reprojecting the middle viewpoint onto the top left viewpoint of the 6×6 grid.

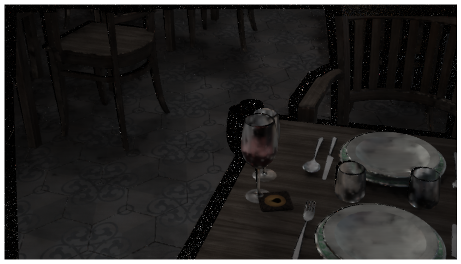

(a) 1 spp input (middle viewpoint)

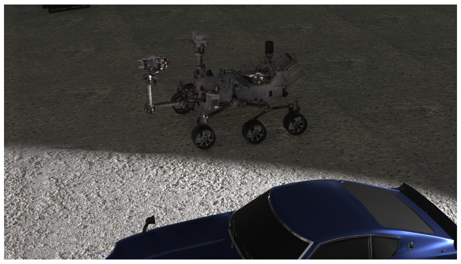
(b) 1 spp input (middle viewpoint)


(c) BMFR + reprojected to top left


(d) BMFR + reprojected to top left


(e) SVGF + reprojected to top left


(f) SVGF + reprojected to top left


(g) 4096 spp reference (top left viewpoint)

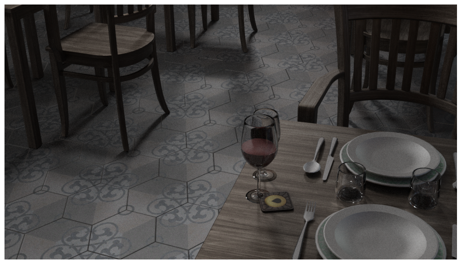
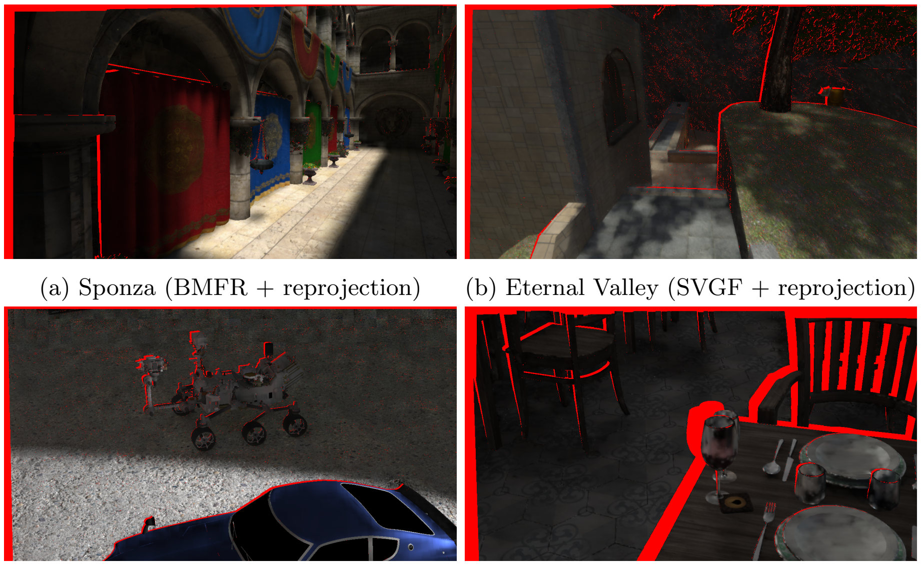

(h) 4096 spp reference (top left viewpoint)

Fig. 7: Results for frame 6 of Warehouse (left column) and frame 8 of Bistro Interior (right column) after denoising and reprojecting the middle viewpoint onto the top left viewpoint of the 6×6 grid.



(a) Sponza (BMFR + reprojection) (b) Eternal Valley (SVGF + reprojection)

(c) Warehouse (BMFR + reprojection) (d) Bistro Interior (SVGF + reprojection)

Fig. 8: The denoising + reprojection results corresponding to Figure 6c, Figure 6f, Figure 7c and Figure 7f, before the discarded reprojection locations (visualized in red) have been filled in with path tracing.