

# Real-time Global Illumination in the CAVE

Jesper Mortensen<sup>1</sup>

Pankaj Khanna<sup>1</sup>

Insu Yu<sup>1</sup>

Mel Slater<sup>1,2</sup>

<sup>1</sup>Department of Computer Science, University College London, London, UK

<sup>2</sup>ICREA-Universitat Politècnica de Catalunya, Department de LSI, Barcelona, Spain

{j.mortensen | p.khanna | i.yu | m.slater}@cs.ucl.ac.uk

## Abstract

Global illumination in VR applications remains an elusive goal. While it potentially has a positive impact on presence, the significant real-time computation and integration complexities involved have been stumbling blocks. In this paper we present recent and ongoing work in the Virtual Light Field paradigm for global illumination as a solution to this problem. We discuss its suitability for real-time VR applications and detail recent work in integrating it with the XVR system for real-time GPU-based rendering in a CAVE<sup>TM</sup>. This rendering method achieves real-time rendering of  $L(S|D)*E$  solutions in time independent of illumination complexity and largely independent of geometric complexity.

**CR Categories:** I.3.7 [Three-Dimensional Graphics and Realism]: Virtual reality; Color, shading, shadowing, and texture

**Keywords:** virtual reality, global illumination, light fields

## 1 Introduction

Presence in virtual environments has many influencing factors. Global illumination of the environments plays a key role in the degree of visual realism that can be achieved, and this realism has been considered to be an important contributing factor for presence (though see Section 2). While a static globally illuminated scene is well within the grasp of current methods, it places limits on the range of environments and tasks that can be effectively represented. Globally illuminated scenes with dynamic shadows, reflections and objects that can represent a wide range of surface materials, could greatly enhance realism and allow new applications of VR. The problems faced however are twofold: Firstly, overcoming the computational complexity of such a real-time rendering system while achieving an acceptable frame-rate is daunting. Secondly, we are faced with a choice of how this is to be achieved – one option is to construct a full system with support for related tasks such as tracking, display management and synchronisation. While this can be achieved on an ad-hoc basis, a general solution requires integration of the global illumination method within a more general rendering system (such as Performer<sup>TM</sup>-CAVELib<sup>TM</sup>, DIVE or XVR) which can again be a complex task.

In this paper, we present the Virtual Light Field (VLF) paradigm as a solution to this problem. We discuss its potential and advantages

in such an application. Finally, we present details of our integration of the VLF rendering method within XVR [Carrozzino et al. 2005] to provide a real-time global illumination solution, including specular surfaces.

## 2 Background

Several studies have looked at the impact of visual realism on presence. While [Hendrix and Barfield 1996] reported an increase in presence due to visual realism, [Zimmons and Panter 2003] found no impact. Other groups have also not found any relationship between visual realism and presence e.g. [Cho et al. 2003]. However, with regards to dynamics in visually realistic environments, such as changing shadows and reflections, the effect on presence has been found to be positive and significant [Slater et al. 1995; Khanna et al. 2006].

Apart from several other factors, the key elements for visual realism are a high-polygon count, realistic models and their global illumination. Ray-tracing [Whitted 1980] and radiosity [Goral et al. 1984] provide a partial global-illumination solution, and have both been considered for VR rendering. While ray-tracing extends easily to real-time dynamics including shadows and reflections, performance is limited by the complexity of the scene and its elements. Global illumination techniques such as path-tracing and photon-mapping amongst others offer a more complete illumination solution, supporting a larger range of materials and light interactions, but do not easily extend to real-time rendering. In VR applications the frame-rate must be real-time and constant, even temporary drops in frame-rate can cause the subject to lose orientation, and even cause motion sickness. Lack of stable frame-rates is a weakness of many caching algorithms where a sudden change in viewpoint can produce a view that is not fully represented in the cache, causing a temporary drop in fidelity or frame-rate. Similarly, dynamic techniques such as ray tracing for global illumination can also exhibit variable frame-rates when the viewpoint changes from a complex region to a less complex region in terms of illumination.

Pre-computed Radiance Transfer [Sloan et al. 2002] offers an approximation to global illumination for static scenes and has been applied to VR rendering in a CAVE in [Dmitriev et al. 2004]. The pre-processed static scene is illuminated by dynamic environment maps for realistic rendering. The Light field [Gortler et al. 1996; Levoy and Hanrahan 1996] presents an image-based approach for representing and rendering radiance information from real or virtual scenes. The advantage of such a representation is that rendering is independent of scene complexity – both in number of polygons and surface materials. Khanna et. al. [Khanna et al. 2004] utilises a DPP light field data structure [Camahort et al. 1998] storing visibility for accelerating ray tracing and subsequently Huang et. al. [Huang et al. 2006] employed a surface light field for that purpose supporting rigid dynamics. Similarly, Ren et. al. [Ren et al. 2005] pre-compute and store visibility for fast global illumination computation of low-frequency lighting at interactive frame rates. The Virtual Light Field [Slater et al. 2004] uses a 5D DPP light field to propagate and represent global illumination in a scene for real-time rendering. Unlike many current techniques in VR and

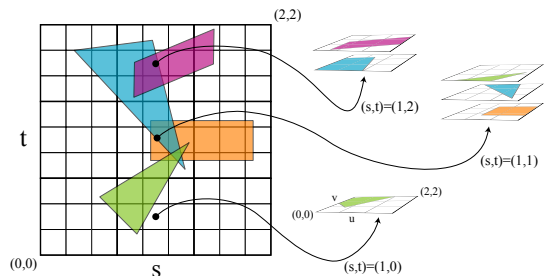
augmented/mixed-reality applications that approximate physically based rendering, the VLF provides a true solution, representing all  $L(S|D)*E$  light paths. After propagation, this radiance information is available for rendering and re-lighting. We believe such a representation has significant potential for allowing realistically illuminated virtual environments, though the rendering method can equally be used to represent a ‘real-world’ light field for VR applications at high, stable frame-rates.

### 3 A brief overview of the Virtual Light Field

#### 3.1 Data-structures

Before presenting our GPU based rendering approach we briefly recap the data structure with some essential notation of the original VLF method [Slater et al. 2004]. Given a scene in world coordinates (WC) we first apply a transformation that translates and uniformly scales the scene such that it is enclosed by the unit sphere centred at the origin; we refer to this as the VLF coordinate system. If  $l$  points with spherical coordinates  $\omega_i = (\theta_i, \phi_i)$  are chosen on the unit sphere, each serve as a normal for a bounded plane orthogonal to  $\omega_i$  large enough to enclose the projection of the unit sphere. Each such unique plane  $i$  is discretised into a regular grid of  $N \times N$  cells, each of which is the origin of a ray parallel to  $\omega_i$ , we refer to such a set of  $N \times N$  parallel rays as a *parallel subfield* (PSF). Each  $PSF_i$  has an associated rotation that aligns the PSF coordinate frame with  $(X, Y, Z)$  such that  $\omega_i$  coincides with  $Y$ ; this is the *canonical* PSF representation. In this representation the ray origin is given by the tuple  $(x, z)$  describing a unique ray parallel to the  $Y$ -axis and  $y$  gives a point along this ray. For efficiency we pre-compute and store a number of transformation matrices. One such matrix pair is  $M_{WC \rightarrow PSF_i}$  and its inverse  $M_{WC \leftarrow PSF_i}$  which transform points between  $WC$  and  $PSF_i$ .

Now consider any ray in the canonical PSF. This will intersect a number of surfaces in the scene. If we parameterise the ray in the form  $r(t) = r_0 + kt, t \geq 0$ , and  $r_0$  is the ray origin, then the intersection points can be characterised as an array of monotonically increasing parametric values  $[t_1, t_2, \dots, t_n]$ . With each one of these intersection points additional information could be stored: the identifier of the surface at that intersection, and eventually the outgoing radiance from the surface at that intersection point. Although this approach is possible, no use would have been made of the great coherence between neighbouring rays, and the memory costs would be substantial. Instead, the grid of cells in a PSF is subdivided into tiles, each at a resolution  $m \times m$  cells, where  $1 \leq m \leq N$  and  $\frac{N}{m}$  is integral. Each tile maintains a sequence of surface identifiers with associated radiance maps for faces intersected by any ray which has its origin within the tile. This is illustrated in Figure 1.



**Figure 1:** Examples of tile lists for four polygons projected to a PSF where  $n = \frac{N}{m} = 3$  and  $m = 3$ . Bold lines mark the tile boundaries.

Thus a radiance value can be retrieved from this parametrisation using the notation:  $L_s(\omega, s, t, u, v, p)$ , where  $\omega$  indicates  $PSF_\omega$ ,  $(s, t)$  is a tile for face  $p$ , and  $(u, v)$  is a cell within this tile. Our approach uses texture atlases for both radiance and irradiance maps for improved efficiency and compact representation.

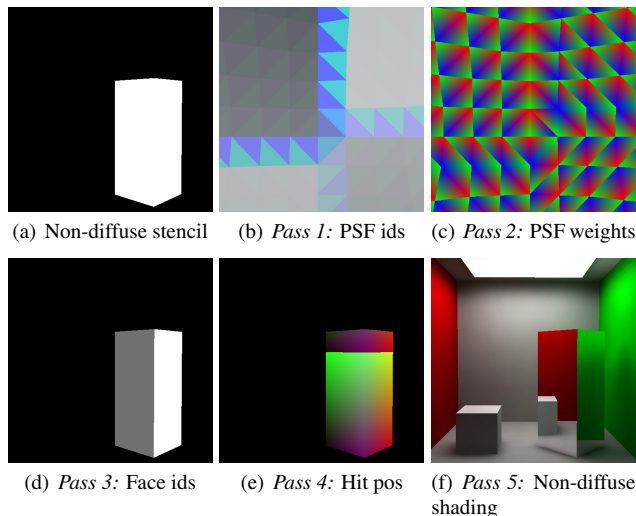
#### 3.2 Propagation

Once the VLF data structure is built, propagation is in principle a straightforward Neumann expansion of the rendering equation. Radiance is emitted from light sources following the paths provided by fixed bundles of parallel rays in the PSFs, which are used as approximations for true ray directions. Coherence is exploited by following parallel bundles of rays rather than dealing with individual rays. This method maps well to the GPU providing a very efficient light transport step. The method can provide solutions with tens of thousands of polygons with millions of ir/radiance elements in minutes.

### 4 Rendering the VLF in the CAVE

When the VLF propagation step has converged, the GPU can render novel views from the data structure by interpolating between samples stored in the diffuse textures and non-diffuse view-dependent radiance tiles. Diffuse surfaces can be rendered directly using texturing with the diffuse textures available in the irradiance texture atlases. The GPU performs interpolation efficiently in this case.

Flat specular faces can be rendered with ray-tracing by recursively following a view ray reflected in the specular face until it strikes a diffuse face where the visible radiance can be collected. A similar idea, often used in real-time VR applications, is to use the stencil buffer to render a reflected view of the scene as seen through the specular face and then paste this onto the face with texturing [Kilgard 2002]. These methods are only efficient if few specular surfaces are present in the scene and do not apply to, for example, glossy BRDFs.



**Figure 2:** Rendering passes.

A more general method is to resample images from the directionally dependent radiance stored in the non-diffuse radiance tiles. As described in Section 3.1, the data structure can be formalised as  $L_s(\omega, s, t, u, v, p)$ . This effectively references a radiance value in direction  $\omega$ , from a point on  $p$  described by the intersection of the

canonical ray  $(s, t, u, v)$  with  $p$ . Due to the discrete representation a PSF matching *exactly* the direction  $\omega$  is rarely available. The three PSFs  $(\omega_i, \omega_j, \omega_k)$  at the vertices of the spherical triangle in which  $\omega$  falls are used with barycentric weights  $(\alpha_i, \alpha_j, \alpha_k)$  for an interpolated value:

$$L_s(\omega, s, t, u, v, p) = \alpha_i * L_s(\omega_i, s, t, u, v, p) + \alpha_j * L_s(\omega_j, s, t, u, v, p) + \alpha_k * L_s(\omega_k, s, t, u, v, p) \quad (1)$$

In order to compute the values necessary to index into Equation 1, four off-screen passes are rendered (see Figure 2). A fifth and final pass performs the final shading producing the globally lit image. In order to identify non-diffuse pixels in the image plane an optional stencil image can be produced by rendering the non-diffuse polygons to an off-screen target (see Figure 2(a)). This can serve to limit the computation performed in each subsequent pass to only non-diffuse pixels.

In *pass 1* the camera is placed at the centre of the unit sphere and the spherical triangles are rendered in false colour to a texture. This produces the indices of the three nearest PSFs  $(\omega_i, \omega_j, \omega_k)$  for each pixel (see Figure 2(b)). This is repeated in *pass 2* this time setting vertex colours for each spherical triangle to  $(1, 0, 0)$ ,  $(0, 1, 0)$  and  $(0, 0, 1)$ . The GPU interpolates this over each triangle, resulting in a texture with three barycentric weights for each pixel (see Figure 2(c)). *Pass 3* serves to determine  $p$ , this time rendering the scene geometry in false colour, yielding a texture with a face identifier for each visible non-diffuse pixel (see Figure 2(d)). *Pass 4* renders the scene geometry again where each vertex is coloured with its world coordinate position, interpolation across the geometry produces a texture with the world coordinate position of the intersection of the viewing ray for that pixel with the face  $p$  (see Figure 2(e)). Note that ray casting could easily replace these last two passes. A fifth and final pass renders the final radiances to the image. For each pixel this is achieved by mapping the hit position to each of the three PSFs by applying the respective  $M_{WC \rightarrow PSF}$  matrix (see Section 3.1) to the hit position, producing an  $(x, y, z)$  value in canonical PSF coordinates where  $(x, z)$  trivially maps to a tile/cell pair  $(s, t, u, v)$ . The tiled data structure is then looked up and a radiance value for each PSF is weighted by its corresponding barycentric weight and written to the image. This is illustrated in Figure 2(f).

Performance is dependent on the time taken to resolve visibility (*pass 3*), the remaining passes and radiance retrieval is small constant time per pixel. Either ray tracing or rasterisation can be used to resolve the visibility, here we use the latter. One of the main points of the VLF approach is that global illumination values can be retrieved directly from the data structure, no further shadow rays or sampling is necessary. This results in stable, predictable frame-rates, which is of great utility in VR applications.

#### 4.1 Implementation details and setup

The rendering method has been integrated into XVR, which is a clustered rendering system capable of driving immersive VR systems such as HMDs, CAVE<sup>TM</sup> systems and multi-screen projection walls (see Figure 3). In our CAVE setup four rendering clients drive the front, left, right and floor projections respectively. OpenGL commands are distributed by a dedicated master node to the rendering clients on a dedicate gigabit LAN. XVR is flexible and allows unconventional rendering to be performed from a plug-in written in C++ and OpenGL.

In order to avoid severe penalties for texture state changes a texture atlas was employed for diffuse textures; packing many textures

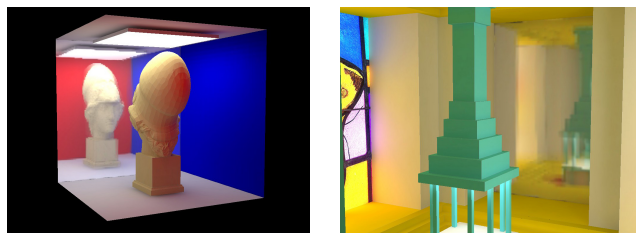


**Figure 3:** User in a VLF rendered virtual bar-room in a CAVE. Note the mirror on the wall.

in a few high resolution texture maps. Similarly, directional radiance tiles were also densely packed into large textures. Further, the  $M_{WC \rightarrow PSF_i}$  matrices were stored for the global set of directions on the GPU in a texture for easy access in *pass 5*. Due to the distributed nature of XVR care was taken to minimise the amount OpenGL commands and data to be sent across the network by exploiting display lists and storing indexing structures and matrices in textures on each client GPU.

## 5 Results

Timings were obtained on an Intel Core 2 Quad (QX6700) 2.66GHz processor with a GeForce 8800 GTX with 768MB graphics memory and 4GB of host memory. Resolution was fixed at  $1024 \times 768$ , with a 2K bi-directional directional discretisation and  $256^2$  radiance maps. Both scenes have a relatively large non-diffuse area, and we are using a perfectly specular BRDF, which is the *worst case* scenario for a light field with discrete directions. Other less directionally dependent BRDFs would require fewer directions and would render with much less visible artifacts. No stencil buffer was used such that the interpolation computation is always performed for *all* pixels in the view yielding a *worst-case* but stable frame-rate.



(a) Atenea scene, 120fps mono (60fps stereo) (b) Grotto scene, 121fps mono (60.5fps stereo)

**Figure 4:** Rendering performance.

The Atenea scene in Figure 4(a) is composed of 9410 polygons with a single emitter. The wall opposite the statuette is specular with a slight bluish diffuse component. The scene contains  $\sim 30.6$ M non-diffuse elements and  $\sim 2.4$ M diffuse elements and propagated in  $\sim 16.3$  minutes with six iterations. The data-structures consume 22.3MB compressed on disk or 142MB texture memory on the GPU. The Grotto scene in Figure 4(b) is composed of 318 polygons with three emitters of which two are textured. The scene contains  $\sim 9.2$ M non-diffuse elements and  $\sim 1$ M diffuse elements and

propagated in  $\sim 2.5$  minutes with six iterations. The data-structures consume 8.1MB on disk or 31MB texture memory on the GPU.

The frame-rates quoted in Figure 4 were sustained from all viewpoints, even when the non-diffuse polygons filled the entire image.

## 6 Conclusion

We have presented a GPU light field rendering method capable of rendering full global illumination for VR applications at real-time frame-rates. Rendering performance is independent of illumination complexity and geometric complexity assuming that visibility can be resolved in real-time. The global illumination shading adds only a small constant time operation per pixel accessing the DPP light field stored on the GPU.

This opens up numerous possibilities for realistically lit VR applications. So far these have been limited to using either traditional light maps (computed with radiosity) or environment mapping with pre-computed radiance transfer (PRT). The former only solves global illumination partly neglecting many important transfer paths and does not apply to dynamic objects, and the latter requires incoming globally lit radiance values typically provided by environment maps and are limited to rigid objects. Environment maps work on the assumption that the object being illuminated is far from the illuminating environment. For many interior VR scenarios with for example avatars this cannot be assumed. The VLF method described in this paper can easily provide correct incoming radiances from a static environment that can then be used with PRT for rigid dynamic objects, integrating dynamic objects into the global illumination solution.

Work almost completed at the time of writing includes adding support for dynamic objects by integrating PRT with the VLF method. Also, support for soft shadows caused by dynamic objects is being added, allowing the integration of dynamic objects into the initial global illumination solution provided by the VLF method. The aim is to enable researchers to create photorealistic VR scenarios with dynamic content such as avatars.

## Acknowledgements

This research was funded by EPSRC grant EP/C511824/1. Thanks to Franco Tecchia, Giuseppe Marino and Marcello Carrozzino from VRMedia for support using the XVR system.

## References

- CAMAHORT, E., LERIOS, A., AND FUSSELL, D. 1998. Uniformly sampled light fields. University of Texas at Austin, Austin, TX, USA.
- CARROZZINO, M., TECCHIA, F., BACINELLI, S., CAPPELLETTI, C., AND BERGAMASCO, M. 2005. Lowering the development time of multimodal interactive application: the real-life experience of the xvr project. In *ACE '05: Proceedings of the 2005 ACM SIGCHI International Conference on Advances in computer entertainment technology*, 270–273.
- CHO, D., PARK, J., KIM, G. J., HONG, S., HAN, S., AND LEE, S. 2003. The dichotomy of presence elements: The where and what. In *VR '03: Proceedings of the IEEE Virtual Reality 2003*, 273.
- DMITRIEV, K., ANNEN, T., KRAWCZYK, G., MYSZKOWSKI, K., AND SEIDEL, H.-P. 2004. A cave system for interactive modeling of global illumination in car interior. In *VRST '04: Proceedings of the ACM symposium on Virtual reality software and technology*, 137–145.
- GORAL, C. M., TORRANCE, K. E., GREENBERG, D. P., AND BATTAILE, B. 1984. Modeling the interaction of light between diffuse surfaces. In *SIGGRAPH '84: Proceedings of the 11th annual conference on Computer graphics and interactive techniques*, 213–222.
- GORTLER, S. J., GRZESZCZUK, R., SZELISKI, R., AND COHEN, M. F. 1996. The lumigraph. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, 43–54.
- HENDRIX, C. M., AND BARFIELD, W. 1996. Presence within virtual environments as a function of visual display parameters. *Presence* 5, 3, 274–289.
- HUANG, P., WANG, W., YANG, G., AND WU, E. 2006. Traversal fields for ray tracing dynamic scenes. In *VRST '06: Proceedings of the ACM symposium on Virtual reality software and technology*, 65–74.
- KHANNA, P., MORTENSEN, J., YU, I., AND SLATER, M. 2004. Fast ray tracing of scenes with unstructured motion. In *SIGGRAPH '04: ACM SIGGRAPH 2004 Posters*, ACM Press, New York, NY, USA, 35.
- KHANNA, P., YU, I., MORTENSEN, J., AND SLATER, M. 2006. Presence in response to dynamic visual realism: a preliminary report of an experiment study. In *VRST '06: Proceedings of the ACM symposium on Virtual reality software and technology*, 364–367.
- KILGARD, M. J. 2002. Improving shadows and reflections via the stencil buffer. White paper, Nvidia Corporation.
- LEVOY, M., AND HANRAHAN, P. 1996. Light field rendering. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, 31–42.
- REN, Z., HUA, W., CHEN, L., AND BAO, H. 2005. Intersection fields for interactive global illumination. *The Visual Computer* 21, 8–10 (September), 569–578.
- SLATER, M., USOH, M., AND CHRYSANTHOU, Y. 1995. The influence of dynamic shadows on presence in immersive virtual environments. In *VE '95: Selected papers of the Eurographics workshops on Virtual environments '95*, 8–21.
- SLATER, M., MORTENSEN, J., KHANNA, P., AND YU, I. 2004. A virtual light field approach to global illumination. In *CGI '04: Proceedings of the Computer Graphics International (CGI'04)*, 102–109.
- SLOAN, P.-P., KAUTZ, J., AND SNYDER, J. 2002. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, 527–536.
- WHITTED, T. 1980. An improved illumination model for shaded display. vol. 23, 343–349.
- ZIMMONS, P., AND PANTER, A. 2003. The influence of rendering quality on presence and task performance in a virtual environment. In *VR '03: Proceedings of the IEEE Virtual Reality 2003*, 293.