A CAVE System for Interactive Modeling of Global Illumination in Car Interior

Kirill Dmitriev, Thomas Annen, Grzegorz Krawczyk, Karol Myszkowski, and Hans-Peter Seidel
MPI Informatik, Saarbrücken, Germany

ABSTRACT

Global illumination dramatically improves realistic appearance of rendered scenes, but usually it is neglected in VR systems due to its high costs. In this work we present an efficient global illumination solution specifically tailored for those CAVE applications, which require an immediate response for dynamic light changes and allow for free motion of the observer, but involve scenes with static geometry. As an application example we choose the car interior modeling under free driving conditions. We illuminate the car using dynamically changing High Dynamic Range (HDR) environment maps and use the Precomputed Radiance Transfer (PRT) method for the global illumination computation. We leverage the PRT method to handle scenes with non-trivial topology represented by complex meshes. Also, we propose a hybrid of PRT and final gathering approach for high-quality rendering of objects with complex Bi-directional Reflectance Distribution Function (BRDF). We use this method for predictive rendering of the navigation LCD panel based on its measured BRDF. Since the global illumination computation leads to HDR images we propose a tone mapping algorithm tailored specifically for the CAVE. We employ head tracking to identify the observed screen region and derive for it proper luminance adaptation conditions, which are then used for tone mapping on all walls in the CAVE. We distribute our global illumination and tone mapping computation on all CPUs and GPUs available in the CAVE, which enables us to achieve interactive performance even for the costly final gathering approach.

Categories and Subject Descriptors

I.3.7 [**Three-Dimensional Graphics and Realism**]: [Virtual reality][Raytracing]

General Terms

Measurement, Design, Algorithms, Performance

Keywords

CAVE, Virtual Reality, LCD panel, BRDF

1. INTRODUCTION

Synthesis of realistic images which predicts the appearance of the real world is a long standing goal in many important virtual reality (VR) applications such as interior design, illumination engineering, environmental assessment, ergonomic studies, along with many others. Predictive rendering should guarantee that a correct image is obtained when valid input data for such rendering is provided. The most critical issue towards such predictivity is a physically correct solution of the global illumination problem, which is extremely challenging in particular in interactive settings due to high computational costs. To reduce those costs some simplifying assumptions about the underlying light transport model are often made and constraints on interaction with the virtual world are imposed.

This research is motivated by an application in automotive industry in which the impact of quickly changing lighting conditions on the visibility of information displayed on LCD panels (commonly mounted on the dashboard in modern cars) is investigated. This requires global illumination solution responding interactively to lighting changes, which result from different car orientations in respect to distant lighting stored as dynamically changing environment maps. Our application scenario is similar to the simulation of free driving in an environment in which buildings, trees, and other occluders change the amount lighting penetrating the car interior. In such a scenario the response for lighting changes should be immediate for an arbitrary observer (virtual camera) position, but we can safely assume that the geometry of car interior is static, which greatly simplifies our global illumination solution. To improve the immersion experience we use the CAVE environment for displaying the car interior. We also employ a head tracking system to monitor the current observer position, which is important to properly model light reflection in the LCD panel. In this paper we briefly survey existing global illumination solutions suitable for interactive applications and analyze their suitability for our VR system. Then we propose a hybrid approach relying on the Precomputed Radiance Transfer (PRT) technique [28] with GPU-based rendering of the car interior and a CPU-based stochastic Monte Carlo integration for the LCD panel rendering. During the integration we use the lighting distribution computed using the PRT technique, which makes this very precise yet costly procedure very efficient.

Another important issue in predictive rendering is the problem of displaying high dynamic range (HDR) images resulting from the global illumination computation on devices with limited dynamic range. The compression of HDR luminance values for accommodating the display range limitations is called tone mapping (refer to a recent survey on tone mapping algorithms [7]). Simple tone mapping algorithms, which do not analyze local image content but instead apply the same tone reproduction curve globally for all pixels, can easily be performed in real-time on modern CPUs [17, 9]. Even more advanced algorithms involving different processing, which might depend on local image content, can be executed at video rates using modern graphics hardware [14]. For sequences with rapid changes in scene intensity, the temporal response of the *human visual system* (HVS) should be modeled. Models of dark and light adaptation [12] have already been incorporated into global tone mapping algorithms [21, 10]. An important parameter required to perform temporally dependent tone mapping is the visual adaptation state, which depends on luminance values in the scene that are determined through the global illumination computation. The scene luminance captured in HDR images may change strongly both in spatial and temporal domains along with displayed image contents. The problem of establishing a proper adaptation level is even more difficult for large size displays, in which case head tracking is required to identify a display region that is currently within the observer field of view. In this research, we propose a solution for the computation of adaptation luminance for the CAVE environment, in which case fragments of two or even three display walls can be seen simultaneously and affect the adaptation state.

The remainder of the paper is organized as follows. In Section 2 we discuss major requirements and constraints imposed on the design of our VR system. In Section 3 we briefly overview existing interactive global illumination algorithms and evaluate their suitability for our application. We describe our hybrid global illumination computation and rendering in the context of the CAVE architecture in Section 4. Then we present our tone mapping solution for multi-display environments in Section 5. Finally, we discuss the obtained results and conclude this paper.

2. DESIGN CONSTRAINTS

The choice of global illumination (GI) solution in our application is strongly constrained by the hardware configuration of the CAVE system. Each of the five walls in this system is powered by two consumer-level dual-processor PCs needed for the stereo projection effect (i.e., 5×2 frames must be simultaneously rendered). Each PC is equipped with a high-end graphic card. Our design goal is to select such GI algorithms that fully exploit the computational power of available CPUs and GPUs and can produce images of high (full screen) resolution at interactive rates.

Our choice of GI algorithm is also influenced by major characteristics of rendered scenes: The car interior. The geometry is quite complex and it is modeled using roughly a half million of mesh triangles. Materials used in the car interior design have usually strongly diffuse reflectance characteristics to avoid false reflections and highlights, which could be harmful for the driver. Such surfaces have strong low-pass filtering properties for reflected lighting [24]. The daylight that illuminates the car interior can be considered as a distant, large area luminaire which is relatively costly to model for sampling-based methods [2].

In terms of user's interaction with the VR system, the car interior can be considered as a static scene. On the other hand, the daylight captured (or simulated) as an HDR environment map can be dynamically changing. Also, the car orientation in respect to the environment map changes during driving simulation. Full freedom of camera motion is allowed to inspect harmful reflections in the LCD panel for any head position of the driver and front-seat passenger.

3. PREVIOUS WORK

Interactive global illumination (GI) algorithms (refer to the recent survey paper [6] on this topic for more details) can be categorized as those that exploit temporal coherence to avoid redundant computations and brute force algorithms that compute each frame from scratch. The algorithms from the first category usually require less computational power at the expense of algorithmic complexity which is required to handle all possible cases of user–scene interaction. The brute force algorithms require huge computational power and usually run on computer clusters (CPU-based) or exploit the programmability of modern graphics hardware (GPU-based).

Temporal coherence in the GI computation can be considered at various levels ranging from ready to display shaded pixels to simple visibility samples shared between frames. Making use of the coherence at a higher level is generally the approach chosen where the response speed is a crucial factor. For example in the Render Cache technique [33] shaded pixels are reprojected from the previous frames to the current frame, while in the Shading Cache technique [30] working in the object space illumination samples are reused for mesh vertices. By considering temporal coherence at lower levels, e.g., at the level of single photon paths, usually more flexibility in sharing information for many frames at once can be achieved. Dmitriev et al. [8] reuse photon hit points in their selective photon tracing technique which enables to identify and update invalid photon paths. Graphics hardware is used to compute the direct illumination with shadows while the photon-based indirect illumination requires scene meshing and works only for Lambertian surfaces.

Wald et al. [31] propose the Instant Global Illumination algorithm which is based on the real-time ray tracing technology [32]. The algorithm efficiently renders specular effects and caustics, but for the indirect lighting computation the best accuracy is achieved for Lambertian surfaces due to exploiting the virtual lights concept. Each frame is essentially rendered from scratch and to achieve interactive performance a cluster of CPUs is required. Network communication between CPUs may limit the frame resolutions.

Recently, Sloan et al. [28] introduced the Precomputed Radiance Transfer (PRT) technique, which permits the illumination of static objects with low-frequency incident lighting represented in spherical harmonics [11]. The object can either be diffuse [28] or glossy [19, 27, 20]. Rendering can be performed in real-time, but requires precomputing the transfer for self-shadowing and other GI effects.

3.1 Discussion

Taking into account our system constraints presented in Section 2 we reject the GI algorithms strongly relying on the temporal coherence because lighting in the car interior may change quite drastically from frame to frame. GI algorithms relying completely on the real-time ray tracing technology are limited in handling full-screen resolution images, especially that we need 5 × 2 of such images in the CAVE. We choose PRT techniques which require very costly preprocessing, but then enable real-time rendering that is particularly efficient for environments with predominantly diffuse reflectance properties. Originally, the PRT techniques were used to render mostly simple, isolated objects. In this work, we leverage those techniques to handle huge meshed models with non-trivial visibility relationships.

The accuracy of light reflection modeling in the LCD panel is the most critical requirement in our VR system. Since the LCD panel reflectance involves also glossy component, the low-frequency lighting assumption that is imposed by view-independent PRT techniques might not hold so well. To improve the spatial resolution of lighting details for a given camera view we use the so-called *final gathering* [25, 29, 5]. In our approach the GI computation is not explicitly performed, but rather the results obtained using the PRT techniques are stochastically integrated for selected sample points in the car interior and environment map [2]. To reduce the variance of such integration the BRDF of LCD panel is used as an importance criterion for choosing sample directions.

As a result of our design we obtain a hybrid GI approach in which the PRT lighting computation and rendering are performed on GPUs for all five CAVE walls in stereo. Our final gathering also uses the results of PRT lighting, but it is performed in parallel on all idle CPUs in our system (each PC is equipped with a dual processor). This way the final gathering reputed as a computationally heavy off-line technique can be performed with interactive performance in our system. Note that the number of pixels that must be computed using final gathering amounts usually to less than 400 × 300, since the LCD panel occupies a relatively small portion of the front screen in the CAVE.

4. SYSTEM ARCHITECTURE

Over the years a standard design pipeline was elaborated in automotive industry. This pipeline includes many stages on the way from designer ideas to a ready car prototype. Because of huge scale and great conservativity existing in automotive industry making revolutionary changes to the pipeline would be infeasible. Therefore, we have limited our changes to developing few new modules and inserting them into the existing pipeline. The architecture of resulting system is sketched in Figure 1.

Typical mesh obtained from the standard CAD/CAM Modeling and Tessellation steps (refer to Figure 1) is not suitable for the PRT computation and rendering. We introduce the Geometry Preprocessor for the mesh optimization and cleaning accordingly to the requirements specific for PRT techniques. The resulting mesh is used by the Predictive Renderer and therefore it is stored in the Database. Also, the mesh is submitted to the PRT Preprocessor in order to compute Radiance Transfer Vectors used for interactive rendering. We describe the algorithms embedded in the PRT and Geometry Preprocessors in Section 4.1.

At the interactive visualization stage we use our Predictive Renderer, which is embedded in the Lightning VR toolkit controlling displays and head tracking in the CAVE. Predictive Renderer exploits precomputed Radiance Transfer Vectors for GPU-based rendering of the car interior. In Section 4.2 we describe our implementation of the PRT rendering in the CAVE. As we discussed in Section 3.1 a final gathering approach is used for high-quality rendering of the LCD panel. In Section 4.3 we present our solution for the LCD panel rendering on a PC cluster, which is incorporated into the Predictive Renderer.

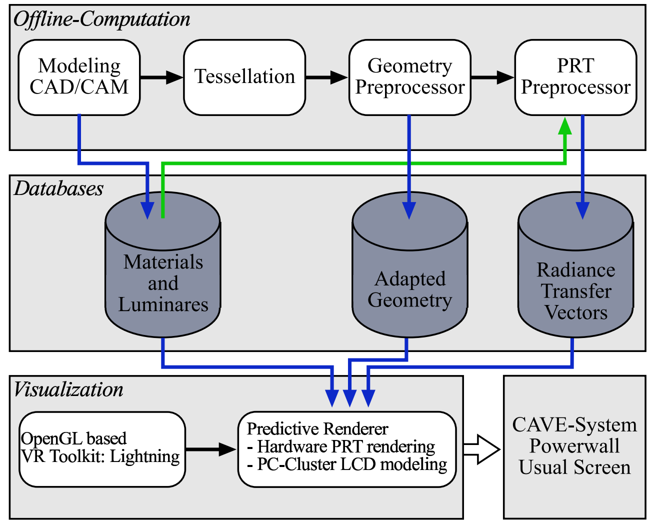


Figure 1: Part of automotive production pipeline devoted to design and visualization and our changes to it.

4.1 Radiance Transfer Precomputation

As discussed in Section 3.1 we select the Precomputed Radiance Transfer (PRT) technique [28] for the GI computation in the car interior, which is decomposed into costly off-line evaluation of light interreflections and very efficient lighting reconstruction performed in realtime during rendering. In this section we briefly outline the main idea of PRT for objects with the Lambertian reflectance function (refer to [28, 18] for a detailed discussion of PRT for materials with arbitrary BRDFs).

The radiance leaving a diffuse surface point $\mathbf{x}$ is determined by the following integral equation:

$$L_{\mathbf{x}} = \frac{\rho_{\mathbf{x}}}{\pi} \int_{\Omega} I_{\mathbf{x}}(\omega) V'_{\mathbf{x}}(\omega) d\omega, \tag{1}$$

$$V'_{\mathbf{x}}(\omega) = V_{\mathbf{x}}(\omega)(\mathbf{n}_{\mathbf{x}} \cdot \omega) \tag{2}$$

where $I_{\mathbf{x}}$ denotes incident radiance at point $\mathbf{x}$, $\rho_{\mathbf{x}}$ is the surface albedo, $V_{\mathbf{x}}$ is the visibility function, and $V'_{\mathbf{x}}$ is the visibility weighted by the cosine between the normal $\mathbf{n}_{\mathbf{x}}$ at $\mathbf{x}$ and direction ω. The incident radiance $I_{\mathbf{x}}$ and the visibility $V'_{\mathbf{x}}$ are projected into a low-order spherical harmonic (SH) basis by integrating against the SH basis functions y_i:

$$\mathbf{I}_{\mathbf{x}} = \int_{\mathbf{\Omega}} y_i(\omega) I_{\mathbf{x}}(\omega) d\omega \tag{3}$$

$$\mathbf{T}_{\mathbf{x}} = \frac{\rho_{\mathbf{x}}}{\pi} \int_{\mathbf{\Omega}} y_i(\omega) V'_{\mathbf{x}}(\omega) d\omega. \tag{4}$$

During rendering the shading integral is then estimated by evaluating the dot-product between the incident lighting vector $\mathbf{I}_{\mathbf{x}}$ and the radiance transfer vector $\mathbf{T}_{\mathbf{x}}$:

$$\tilde{L}_{\mathbf{x}} = \mathbf{I}_{\mathbf{x}} \cdot \mathbf{T}_{\mathbf{x}}. \tag{5}$$

The projection of incident lighting in Equation (3) is inexpensive and it is performed for each frame. Since the SH projected incident lighting $\mathbf{I_x}$ is factored out in Equation (5) lighting can freely change for each frame as it is required in our application. To model dynamically changing lighting we interpolate between captured HDR environment maps, or use an analytic daylight model developed by Preetham et al. [22] with scripted parameters controlling the dynamic sky appearance.

The computation of transfer vectors $\mathbf{T_x}$ is expensive and it is performed off-line. The precomputed vectors $\mathbf{T_x}$ remain valid as long as the scene geometry and reflectance properties are static. Monte Carlo simulation is commonly applied for estimating $\mathbf{T_x}$ for each mesh vertex $\mathbf{x}$. Rays s_d uniformly distributed over a sphere are traced from $\mathbf{x}$. The transfer vector element $T^i_{\mathbf{x}}$ (corresponding to the SH basis function y_i) that describes the bounce b of light is then an average of each ray contribution:

$$T^i_{\mathbf{x}}(b) = \frac{\rho_{\mathbf{x}}}{\pi N_{rays}} \sum_{d=0}^{N_{rays}} y_i(s_d, b) V'_{\mathbf{x}}(s_d, b) \qquad (6)$$

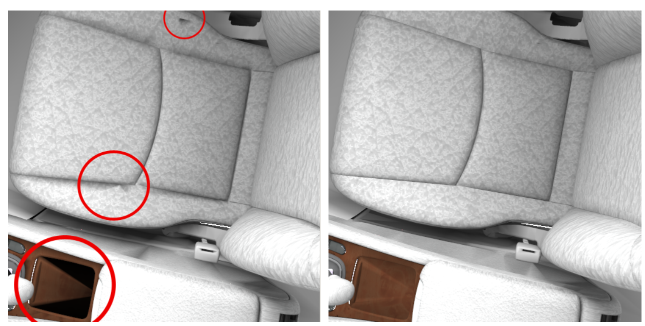
where N_{rays} is the number of traced rays. The meaning of $V'_{\mathbf{x}}(s_d, b)$ and $y_i(s_d, b)$ changes as a function of b. For the very first bounce $V_{\mathbf{x}}(s_d, b)$ (refer to Equation 2) is equal to 1 if ray s_d reaches the environment map and 0 otherwise (i.e., when it is blocked by the scene objects). $y_i(s_d, b)$ equals to the value of SH basis function i in the direction of ray s_d. For all the subsequent bounces $V_{\mathbf{x}}(s_d, b)$ is equal to 0 if ray s_d reaches the environment map and 1 otherwise. $y_i(s_d, b)$ equals to $T^i_q(b-1)$, where q is the surface point intersected by s_d (in practice $T^i_q(b-1)$ is linearly interpolated based on the radiance transfer vectors at vertices of the mesh element that contains q). To improve the accuracy of $\mathbf{T_x}$ estimation for a given number of traced rays we use the Sobol low discrepancy sequence for the generation of ray directions. This sequence is known for its good properties and has been applied before in computer graphics [31].

In existing PRT solutions [28, 19, 27, 20] the radiance transfer vectors $\mathbf{T_x}$ are computed and stored at mesh vertices. In such a case the only way to increase the local accuracy of reconstructed lighting is to tessellate the mesh more densely. Since this leads to superfluous vertex transformations, we experimented with storing the PRT data in textures. This requires finding the bijective mapping (called the parametrization) between the mesh and corresponding textures. Finding an appropriate parameterization proved to be difficult because existing algorithms are designed for meshes that are topologically equivalent to disk [13]. Applying them to an industrial car model would require manual cutting of this huge model into smaller disk-like parts and correcting all non-manifold problems. The most serious problem we faced was securing smooth texture interpolation for such pairs of mesh elements that are separated in the texture space while being adjacent in the model. In such a case smooth appearance during bilinear interpolation is typically achieved by applying a dilation procedure, which replicates texels belonging to one element along the corresponding edge of another element. Since many elements in our model are so tiny that they occupy less than one texel in the texture space, the dilation fails. Increasing the texture resolution was not an option because of texture memory limitations (refer to Section 6 for the GPU specification used in this research). Therefore, we decided to abandon the texture-based approach and we used mesh vertices to store the PRT data. In the following section we discuss a number of problems that arise with meshes used in industrial practice within the PRT framework.

4.1.1 PRT for Arbitrary Meshes

PRT as well as all other mesh based global illumination solutions, impose multiple restrictions on the quality of mesh [3]. We encountered many serious problems with our mesh model such as duplicated triangles, cracks between patches resulting from the NURBS tessellation, overlapping triangles, and sometimes quite irregular meshing. Such a poor quality mesh is perfectly acceptable for simple OpenGL rendering and shading, but leads to unacceptable artifacts in the PRT algorithm.

While it is relatively easy to make mesh better shaped by re-tessellating sliver-like triangles as well as remove duplicated triangles and cracks that are thinner than a user-defined threshold [3], overlapping triangles are a really big problem for PRT. A vertex that is just slightly covered by the neighboring triangle usually causes a well-visible shadow leak. To overcome this problem, instead of shooting rays from the vertex [28], we shoot them from stochastically chosen sample points within the triangle. The contribution of each sample is distributed to the triangle vertices by taking into account the sample point distance to each vertex. This reduces shadow leaks significantly, especially for a very common case when a vertex is only slightly covered by another triangle. Figure 2 illustrates the improvement of shading quality using this technique.



a)   b)

Figure 2: Rendering with PRT. a) Spherical harmonics were computed only at vertices. Artifacts are marked with red circles. b) Spherical harmonics were computed at random surface points and then redistributed to closest vertices with appropriate weights.

Many elements of the car interior are modeled as non-solid objects with non-consistent normal vector orientations. This makes it difficult to identify a side of mesh element that represents the object interior and therefore its shading is not needed because it is always invisible. The safest approach for such objects is to compute two different sets of PRT vectors for each side of a corresponding mesh element, and then use two-pass rendering with backface culling for the visualization of currently visible side. This increases the memory consumption twice and significantly reduces the rendering speed. We propose a solution which is well tailored for the car interior visualization in the CAVE. In this case the only illumination to be considered must pass through the car windows (refer to Figure 3). Therefore, we emit rays from a given triangle over the whole sphere of possible directions but only those rays that pass through the window are considered. As a result, the PRT vectors assigned, for example, to the roof of the car, account only for illumination coming from the inside, but not outside. Thus if the car is viewed from the inside, proper transfer vectors are used, even though shape closeness information is not available.

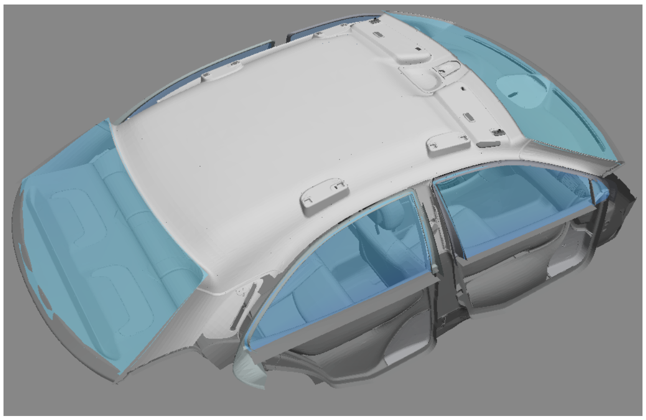


Figure 3: External view of the car model. Only the rays going through the windows contribute to the car illumination.

4.2 The PRT Rendering

Our rendering algorithm is tailored to the existing CAVE setup, which runs at a resolution of 1440×1440 and therefore requires a frame buffer of size 2880×2880 using standard $2\times$ FSAA. This allows for fast rendering when expensive tone mapping is done only on visible pixels. Our algorithm consists of three steps.

First, we evaluate the adaptive tone mapping parameters (refer to Section 5) by rendering a low resolution image of the scene into an HDR off-screen buffer. For all presented results this resolution was 128×128. During low resolution rendering we use a simple pixel shader which converts RGB colors to luminance values. These values are read back to the host memory and are used for estimation of luminance affecting the adaptation level.

During the second pass we render the scene geometry only into the depth buffer. This is not expensive because modern GPUs are optimized for z-only rendering. Finally, we render the scene with lighting enabled and set the depth test to *equals*. This way, pixels where z-test fails are removed early and tone mapping is only performed for pixels that are actually displayed.

4.3 Reflections in LCD Panel

The numerical simulation of light reflection and emission for the LCD panel is difficult due to its complex layered structure. In a typical LCD panel the layers of backlight device, brightness enhancement film, liquid crystal sandwich, light control film, and antiglare / antireflection film can be distinguished (here we list the layers in the order form the most internal to the topmost one), and wave effects such as light polarization must be considered to properly model light transport between the layers [23]. Specialized software such as DIMOS (autronic-MELCHERS GmbH) or SPECTER (Integra, Inc.) can be used for such a modeling task. In this work, we use only measured BRDFs for existing LCD panels, but obviously simulation results from DIMOS or SPECTER could be easily used as well. Checking the ergonomics of LCD panels designed for the car cockpit [1] before their actual manufacturing is an important application of our system.

The LCD panel reflectance properties are strongly influenced by the topmost antiglare / antireflection layer, whose the main task is to absorb as much of incoming light as possible and to attenuate strong specular highlights. A typical measured BRDF for an LCD panel is shown in Figure 4. The measured data resolution is up to one degree and we use the bilinear interpolation to derive intermediate values. Even though it is possible to simulate arbitrary BRDF with the PRT, too many spherical harmonics coefficients might be required to represent highly specular BRDFs as the one shown in Figure 4 with high accuracy. Therefore, we apply a path tracing method [16], which does not impose any limitations on the reflectance functions.

Technically, the LCD panel surface is represented as a texture and path tracing integration of reflected lighting is performed for each texel. The observer position, which is provided by the tracking device, is used to create a ray from the eye to a given texel. Then a number of reflected rays are spawned using the measured BRDF as a criterion for importance sampling [4]. In practice, this means that a majority of reflected rays is concentrated around the direction of ideal specular reflection. In a traditional path tracing the subsequent bounces of light scattering must be considered. However, in our approach we have luminance values provided by PRT for every point in the car interior and we can limit tracing of rays reflected form the LCD panel surface just to one bounce. If a reflected ray hits the car window we query luminance directly from the HDR environment map. Otherwise, the luminance value is linearly interpolated based on the PRT computation for vertices of a mesh element intersected by the reflected ray. Effectively, our display rendering algorithm reduces to final gathering [25, 29, 5], which significantly improves the performance in respect to the full-fledged path tracing.

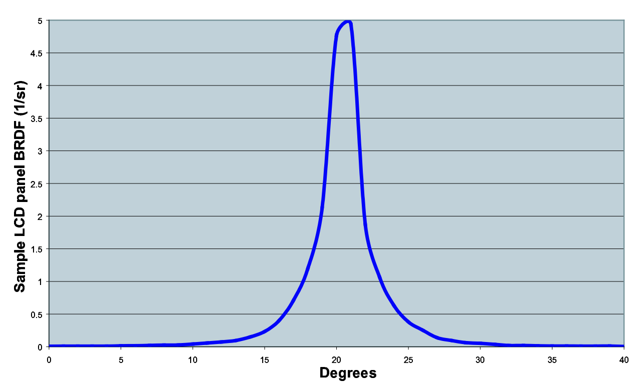


Figure 4: Measured BRDF of a sample LCD panel. The BRDF is visualized for a fixed incoming light direction ($\theta_i = 21^\circ$) and varying θ angle of the observation direction.

Even with only one bounce of path tracing, the LCD panel texture is computed at lower pace than the car interior can be visualized. Therefore, we run those computations as asynchronous threads. The thread responsible for updating the LCD panel texture executes a loop, which queries the current observer position and computes the texture according to this position. The car visualization thread, in its rendering routine, updates the texture image in graphics card memory in case a newer version is available. In this way GPU rendering is not blocked until the LCD panel texture is computed completely. Also, the threads do not need synchronization because the data is accessed for writing only by one thread.

5. HDR DISPLAY IN CAVE

Rendering using precomputed radiance transfer with high dynamic range environment map results in images that contain broad range of luminance values. It is not straightforward to display these results using the CAVE projection system with limited capabilities in terms of displayed luminance. The solution here is to implement an additional post-processing stage realizing tone mapping [7], during which the simulated real-world values of luminance are mapped to displayable RGB values. Many tone mapping algorithms are available. Out of possible choices, the algorithm by Drago et al. [9] appears to be the most appropriate. The algorithm is based on the logarithmic compression of luminance values imitating human response to light (Weber-Fechner law) and its implementation on GPU operates at high frame rates. In the following sections we first discuss the aspect of visual adaptation, which is necessary for correct tone reproduction in multi-display systems like the CAVE. Then we give implementation details of our method for the display of HDR data in the CAVE.

5.1 Tone Mapping for Multi-display System

Although the algorithms of tone mapping, mainly designed for single CRT displays, can be directly adopted to multi-display systems like the CAVE, the aspect of coherent results on all screens needs to be taken into account. Tone mapping in the CAVE system is in fact a parallel process running on 5×2 autonomous projection systems (displays). A naïve approach of executing the tone mapping on each projection system independently would lead to inconsistent results especially visible at the edges and corners of the CAVE room. This can happen, because the characteristics of the HDR frames rendered on individual displays may vary and thus the tone mapping process will be differently adjusted for each of them.

The behavior of tone mapping operators is driven by several parameters typical to a given algorithm. For the method used in our system these parameters are the luminance adaptation level and maximum visible luminance value. The tone mapping process executed on each projection system must work with the same values of these parameters to achieve coherent results between displays. This requires a centralized model for computation and distribution of these parameters which we describe in Section 5.4.

5.2 Visual Adaptation in CAVE

The most important parameter of the tone mapping algorithm is the current luminance adaptation level that describes the luminance value to which the human visual system is currently adjusted. This parameter is usually estimated as a logarithmic average of all displayed pixels [21, 9]. To achieve coherent results on all displays in the CAVE we would need to calculate the average of luminance values of pixels on all walls. However this approach is not correct for the CAVE environment – the luminance characteristics are often too varied between different displays (walls) and it is never the case that a viewer observes the image on all walls at once.

A suitable model for this case is the foveal adaptation, already introduced to interactive tone mapping by Scheel et al. [26]. This model is derived from the fact that the adaptation process is mainly influenced by the light projected on a small area around the center of retina – fovea. Therefore, the level of adaptation should be adjusted to the region at which an observer is currently looking. Although the fovea extends roughly to only $1.5^\circ - 2^\circ$ in diameter, the cone density reaches its minimum around 10° of eccentricity in the visual angle units [15]. In our system we assume that the size of the region affecting the level of visual adaptation is 10°. We obtain the information about viewing direction from the head tracking system which is used to calculate the coordinates of the point on the viewing plane (wall) at which the observer is looking, and a distance of the observer to this point. This information allows us to identify the region on the viewing plane that affects the level of adaptation. It is important to note that the foveal region can be located on one display, but it can also occupy two or three neighboring displays (walls). On the other hand, the other displays do not affect the level of adaptation at all. Therefore, we accumulate the information about luminance values in the foveal region from all displays, and estimate the level of luminance adaptation in the CAVE using centralized computation model (refer to Section 5.4).

5.3 Temporal Aspects of Adaptation

The level of luminance adaptation can change significantly over time because of either overall change in the level of scene illumination or due to rapid head movements (for instance when an observer looks first at a source of light and then at a surface in a shadow). The HVS does not react to such changes instantly, but undergoes processes of dark and light adaptation. To account for this we model temporal aspects of HVS adaptation to luminance changes, namely we follow the scheme introduced by Pattanaik et al. [21]. With this, we are able to estimate the temporal invisibility of information on the LCD panel that can occur if the observer is exposed to extreme luminance levels during driving conditions. We consider the temporal aspect of adaptation during the calculation of luminance adaptation level.

5.4 Implementation

The implementation of our method for displaying HDR images in the CAVE can be divided into three parts: the estimation of luminance affecting the adaptation level, centralized calculation of the adaptation level including the temporal aspect, and the actual process of tone mapping.

For the estimation of luminance affecting the adaptation level we perform a low resolution gray-scale (luminance) rendering of the full frame to an off-screen buffer. The rest of the process is executed on CPU. We first locate the foveal region on the given viewing plane and then calculate the logarithmic sum of luminance values in this region. We also find the maximum luminance value rendered for the given display. This information, along with the number of pixels affecting the level of adaptation on this display (wall) is sent to the *visual adaptation model* using the TCP/IP socket connections.

The algorithm, which calculates the *visual adaptation model*, runs as a separate server application on one of the display machines. The IP address of this server is known and supplied as a parameter to the rest of the rendering machines during the system start-up. The server application runs in cycles, performing custom number of cycles per second – the rate should correspond to the current rendering frame rate. In each cycle, we collect the information about regions affecting the foveal adaptation from all rendering machines using the TCP/IP socket connections. We find the maximum displayed luminance on all displays, and find the goal level of luminance adaptation which is the logarithmic average of luminance values in the foveal region. We then calculate a new state of the HVS adaptation using the approach proposed by Pattanaik et al. [21]. The current level of luminance adaptation and the maximum displayable luminance is then sent back to all rendering machines.

The rendering routine on the display machines contains the code which receives the information on the current state of HVS from the socket connection to the server application. We perform the update of this parameters just before the execution of the tone mapping. The update is done asynchronously meaning, that if there are no data from the visual adaptation model server, the previous state is reused. The tone mapping algorithm is implemented on graphics hardware as a fragment program. It is executed on the fly during the rendering of HDR frame for the display.

6. RESULTS

The results presented in this section are obtained in the CAVE environment. The CAVE is operated by 11 PCs, each is a Dual XEON 3.06 MHz with 2 GB RAM. The PCs are connected to each other with a 1 Gbit Ethernet network. One of the PCs is allocated as a master and manages 10 slave PCs, each of which in turn controls one of 10 projectors. 10 projectors create stereo images on the 5 walls of the CAVE. Each slave PC is equipped with GeForce Quattro FX 3000G graphics card with 256 MB of graphics memory.

The car model that we have used consists approximately of 500K triangles and is illuminated by either a photographed or analytically generated HDR environment map. Figure 5 shows the result of our interactive rendering of the car interior in the CAVE with the stereo projection.

The mesh optimization and the computation of spherical harmonics coefficients (refer to Section 4.1) performed by Geometry and PRT Preprocessors (refer to Figure 1) takes about 100 minutes on one processor. Since the preprocessing computations are performed only once for a given car interior model, they are not time critical and we did not make any effort to run them in parallel which would be clearly possible. The size of full model with precomputed radiance transfer vectors is about 60 MB, which easily fits into the graphics card memory.

The framerate of GPU-based PRT rendering (refer to Section 4.2) is about 10 fps for the synchronized displays in the CAVE (refer to Figure 5). For each frame a current HDR environment map is projected into the SH basis (refer to Equation 3) and all pixels are tone mapped (refer to Section 5).

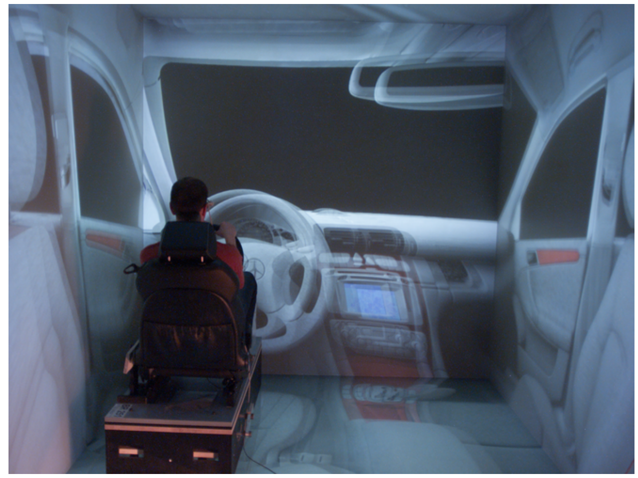


Figure 5: Interactive rendering of car interior. Photograph of the CAVE system with the stereo projection. The view is strongly distorted because it shows the car interior as seen from the perspective of the person wearing glasses with tracking device, but not the photocamera used to take this picture.

One processor on each slave PC is busy with sending data for OpenGL rendering and another one remains idle. We use the idle processors available on all PCs to update the LCD panel texture as the observer moves. The texture is computed in a progressive way. Each PC takes into account the current camera position to compute a rough image of the LCD panel using a moderate number of samples. At present we use 40 samples per one texel and resulting image of the LCD panel is shown in Figure 6. As soon as the rough image computation is finished, the PC sends it to all other PCs using the network broadcast. The packet sent over the network contains the rough image of the LCD panel and the camera position for which it has been computed. Each receiving PC compares the camera position in the packet to its own current camera position. If the positions are the same, both images are summed up. As a result, when the observer stands still, only 1–2 seconds are needed to sum up a sufficient number of rough images from all 11 PCs available in the CAVE to produce the LCD panel texture of high quality (Figure 7). As soon as the observer position changes the system again displays an updated rough version of the LCD panel (Figure 6).

The overall processing required for HDR data display in the CAVE includes adaptation estimation, tone mapping and visual adaptation model. The approach to the estimation of luminance adaptation using low resolution rendering proved to be efficient. The rendering of depth-values and using the z-test (refer to Section 4.2) improves the performance significantly by limiting tone mapping only to visible pixels. The accuracy of low resolution analysis of the frame buffer is sufficient. Some incoherence of pixel values between frames caused by down sampling is smoothed by adaptation process and does not result in perceivable flickering. Network traffic generated by sending adaptation data over Ethernet is very low and it rarely happens that the data do not arrive on time to rendering machines. Eventual reuse of the adaptation data from the last frame during tone mapping also did not lead to visible distortions of brightness appearance

Figure 8: Screen shot from our application presenting the results of global illumination modeling in the car interior. The left image is a result displayed without tone mapping. The right image is tone mapped using the method described in Section 5.

Figure 6: Rough image of the LCD panel computed by path tracing using 40 samples per texel. The image of this quality is shown just after the observer position has changed.

Figure 7: Converged image of the LCD panel. The image of this quality is shown after 2 seconds of computations by the cluster PCs.

between displays. We present comparison of the displayed results with and without tone mapping in Figure 8.

7. CONCLUSIONS

We presented efficient global illumination and tone mapping solutions for the CAVE applications involving static geometry and dynamic lighting conditions. We showed a successful application of our techniques for the car interior modeling in the CAVE, and we efficiently utilized all available there CPU and GPU resources to achieve interactive rendering performance. While the overall appearance of the car interior is judged realistic, we developed a high-quality predictive solution for the LCD panel rendering. Such predictivity is required to use our system for ergonomic and psychophysic studies concerning the readability of information for various types of LCD panels in changing lighting conditions. An important aspect in such studies is the temporal adaptation of human perception to various levels of luminance, which we address in our tone mapping.

As future work we intend to investigate the suitability of all frequency PRT lighting techniques in our application (recently, some papers have been announced for publication at the Siggraph 2004 conference). Also, we plan to use a HDR video camera with the fish-eye lens mounted atop of the car roof to capture dynamic environment maps during real driving conditions. Such a HDR video can be later used in our global illumination computation to reconstruct those driving conditions in the CAVE.

8. ACKNOWLEDGEMENTS

We would like to thank Michael Arnold for organizing our stay at DaimlerChrysler AG Virtual Reality Center and help with other organizational issues within the project, Thomas Ganz for providing us with measured display reflectivity data, Matthias Bues for help with integration of our rendering module into Lightning software, and DaimlerChrysler AG VRC for allowing us to access their CAVE equipment.

9. REFERENCES

[1] *ISO 13406-2, 2001, Ergonomic requirements for work with visual displays based on flat panels - Part2: Ergonomic requirements for flat panel displays.* 2001.

[2] S. Agarwal, R. Ramamoorthi, S. Belongie, and H. W. Jensen. Structured Importance Sampling of Environment Maps. *ACM Transactions on Graphics*, 22(3):605–612, 2003.

[3] D. Baum, S. Mann, K. Smith, and J. Winget. Making radiosity usable: Automatic preprocessing and meshing techniques for the generation of accurate radiosity solutions. In *Computer Graphics (Proceedings of SIGGRAPH 91)*, volume 25, pages 51–60, 1991.

[4] P. Christensen. Adjoints and importance in rendering: An overview. *IEEE Transactions on Visualization and Computer Graphics*, 9(3):329–340, 2003.

[5] P. Christensen, D. Lischinski, E. Stollnitz, and D. Salesin. Clustering for glossy global illumination. *ACM Transactions on Graphics*, 16(1):3–33, 1997.

[6] C. Damez, K. Dmitriev, and K. Myszkowski. State of the Art in Global Illumination for Interactive Applications and High-quality Animations. *Computer Graphics Forum*, 22(1):55–77, 2003.

[7] K. Devlin, A. Chalmers, A. Wilkie, and W. Purgathofer. Tone Reproduction and Physically Based Spectral Rendering. In *Eurographics 2002: State of the Art Reports*, pages 101–123. Eurographics, 2002.

[8] K. Dmitriev, S. Brabec, K. Myszkowski, and H.-P. Seidel. Interactive Global Illumination Using Selective Photon Tracing. In *Proc. of the 13th Eurographics Workshop on Rendering*, pages 25–36, 2002.

[9] F. Drago, K. Myszkowski, T. Annen, and N. Chiba. Adaptive Logarithmic Mapping For Displaying High Contrast Scenes. *Computer Graphics Forum, proceedings of Eurographics 2003*, 22(3):419–426, 2003.

[10] F. Durand and J. Dorsey. Interactive Tone Mapping. In *Rendering Techniques 2000: 11th Eurographics Workshop on Rendering*, pages 219–230, 2000.

[11] A. Edmonds. *Angular Momentum in Quantum Mechanics*. Princeton University, Princeton, NJ, 1960.

[12] J. Ferwerda, S. Pattanaik, P. Shirley, and D. Greenberg. A Model of Visual Adaptation for Realistic Image Synthesis. In *Proceedings of SIGGRAPH 96*, Computer Graphics Proceedings, Annual Conference Series, pages 249–258, Aug. 1996.

[13] M. Floater and K. Hormann. Surface parameterization: a tutorial and survey. In *Multiresolution in Geometric Modelling*. Springer, 2004.

[14] N. Goodnight, R. Wang, C. Woolley, and G. Humphreys. Interactive Time-Dependent Tone Mapping Using Programmable Graphics Hardware. In *Rendering Techniques 2003: 14th Eurographics Symposium on Rendering*, pages 26–37, 2003.

[15] D. Hood and M. Finkelstein. *Handbook of Perception and Human Performance*, volume I: Sensory Processes and Perception, chapter 5: Sensitivity to light, pages 5–4 – 5–6. NY: John Wiley and Sons, 1986.

[16] J. Kajiya. The rendering equation. In *Computer Graphics (ACM SIGGRAPH '86 Proceedings)*, pages 143–150, 1986.

[17] S. Kang, M. Uyttendaele, S. Winder, and R. Szeliski. High Dynamic Range Video. *ACM Transactions on Graphics*, 22(3):319–325, 2003.

[18] J. Kautz. *Realistic, Real-Time Shading and Rendering of Objects with Complex Materials*. Cuvillier Verlag, Göttingen, 2003.

[19] J. Kautz, P.-P. Sloan, and J. Snyder. Fast, Arbitrary BRDF Shading for Low-Frequency Lighting Using Spherical Harmonics. In *13th Eurographics Workshop on Rendering*, pages 301–308, June 2002.

[20] J. Lehtinen and J. Kautz. Matrix Radiance Transfer. In *Proceedings of the 2003 symposium on Interactive 3D graphics*, pages 59–64, 2003.

[21] S. Pattanaik, J. Tumblin, H. Yee, and D. Greenberg. Time-Dependent Visual Adaptation for Realistic Image Display. In *Proceedings of ACM SIGGRAPH 2000*, Computer Graphics Proceedings, Annual Conference Series, pages 47–54, July 2000.

[22] A. Preetham, P. Shirley, and B. Smits. A practical analytic model for daylight. In *Proceedings of SIGGRAPH 99*, Computer Graphics Proceedings, Annual Conference Series, pages 91–100, 1999.

[23] P.Yeh and C. Gu. *Optics of Liquid Crystal Displays*. John Wiley & Sons, 1999.

[24] R. Ramamoorthi and P. Hanrahan. An Efficient Representation for Irradiance Environment Maps. In *Proceedings of ACM SIGGRAPH 2001*, Computer Graphics Proceedings, Annual Conference Series, pages 497–500, 2001.

[25] M. Reichert. *A Two-Pass Radiosity Method to Transmitting and Specularly Reflecting Surfaces*. M.Sc. thesis, Cornell University, 1992.

[26] A. Scheel, M. Stamminger, and H.-P. Seidel. Tone reproduction for interactive walkthroughs. In *Proceedings of Eurographics 00*, volume 19, 3 of *Computer Graphics Forum*, pages 301–312, 2000.

[27] P.-P. Sloan, J. Hall, J. Hart, and J. Snyder. Clustered Principal Components for Precomputed Radiance Transfer. *ACM Transactions on Graphics*, 22(3):382–391, 2003.

[28] P.-P. Sloan, J. Kautz, and J. Snyder. Precomputed Radiance Transfer for Real-Time Rendering in Dynamic, Low-Frequency Lighting Environments. *ACM Transactions on Graphics*, 21(3):527–536, 2002.

[29] B. Smits, J. Arvo, and D. Greenberg. A Clustering Algorithm for Radiosity in Complex Environments. In *Computer Graphics (ACM SIGGRAPH '94 Proceedings)*, pages 435–442, 1994.

[30] P. Tole, F. Pellacini, B. Walter, and D. P. Greenberg. Interactive Global Illumination in Dynamic Scenes. *ACM Transactions on Graphics*, 21(3):537–546, 2002.

[31] I. Wald, T. Kollig, C. Benthin, A. Keller, and P. Slusallek. Interactive Global Illumination. In *Proceedings of the 13th Eurographics Workshop on Rendering*, pages 15–24, 2002.

[32] I. Wald, P. Slusallek, C. Benthin, and M. Wagner. Interactive Rendering with Coherent Ray Tracing. *Computer Graphics Forum*, 20(3):153–164, 2001.

[33] B. Walter, G. Drettakis, and S. Parker. Interactive Rendering using the Render Cache. In *Proceedings of the 10th Eurographics Workshop on Rendering*, pages 235–246, 1999.