GPU Accelerated Direct Volume Rendering on an Interactive Light Field Display

Marco Agus, Enrico Gobbetti, José Antonio Iglesias Guitián, Fabio Marton, Giovanni Pintore

Visual Computing Group, CRS4, Pula, Italy – `http://www.crs4.it/vic/`

Abstract

We present a GPU accelerated volume ray casting system interactively driving a multi-user light field display. The display, driven by a single programmable GPU, is based on a specially arranged array of projectors and a holographic screen and provides full horizontal parallax. The characteristics of the display are exploited to develop a specialized volume rendering technique able to provide multiple freely moving naked-eye viewers the illusion of seeing and manipulating virtual volumetric objects floating in the display workspace. In our approach, a GPU ray-caster follows rays generated by a multiple-center-of-projection technique while sampling pre-filtered versions of the dataset at resolutions that match the varying spatial accuracy of the display. The method achieves interactive performance and provides rapid visual understanding of complex volumetric data sets even when using depth oblivious compositing techniques.

Categories and Subject Descriptors (according to ACM CCS): B.4.2 [Input/Output and Data Communications]: Input/Output Devices Image Display I.3.3 [Computer Graphics]: Picture/Image Generation I.3.7 [Computer Graphics]: Three-dimensional graphics and realism

1. Introduction

The rapid development of programmable graphics hardware is making it possible to interactively render volumes with high visual fidelity on commodity PCs. Resolving the spatial arrangement of complex three-dimensional structures in images produced by such techniques is however often a difficult task. In particular, medical data produced by CT or MRI scans often contain many overlapping structures, leading to cluttered images which are difficult to understand. Enhancing depth and shape perception in volumetric rendering is thus a very active research area, which is tackled under different angles. Recent contributions include methods for supporting real-time rendering and user interaction, improving rendering quality with advanced photorealistic models, e.g., including specular reflections and shadows, or developing non-photorealistic approaches to emphasize model features by illustrative techniques, e.g, vicinity shading, edge detection, halos, or cutaways. An orthogonal research direction consists of improving volumetric understanding by presenting results on displays able to elicit more depth cues than the conventional 2D monitor or providing improved color reproduction. For instance, Ghosh et al [GTH05] have shown how a high dynamic range display can substantially improve volume understanding through perceptually optimized transfer functions. In this work, we focus on enhancing spatial understanding of 3D data through perceptual cues for accommodation, stereo and motion parallax delivered by a light field display, i.e., a display supporting high resolution direction selective light emission. This direction looks very promising, since there is evidence that ego- and/or model-motion as well as stereopsis are essential cues to achieve rapid direct perception of volumetric data [BBD∗07, ME04]. Recent advances in 3D display design demonstrate that high resolution display technology able to reproduce natural light fields is practically achievable [BFA∗05, JMY∗07]. Rendering for such displays requires generating a large number of light beams of appropriate origin, direction, and color, which is a complex and computationally intensive task. Moreover, the displays optical characteristics impose specialized rendering methods.

In this article, we present and demonstrate a GPU accelerated volume ray casting system interactively driving a multi-user light field display based on projection technology. The display, driven by a single programmable GPU through a DVI link, is based on a specially arranged projector array placed behind an anisotropic holographic screen. This setup provides full continuous horizontal parallax in a sizeable zone in front of the screen. The restriction to horizontal parallax reduces light field complexity, making the real-time rendering
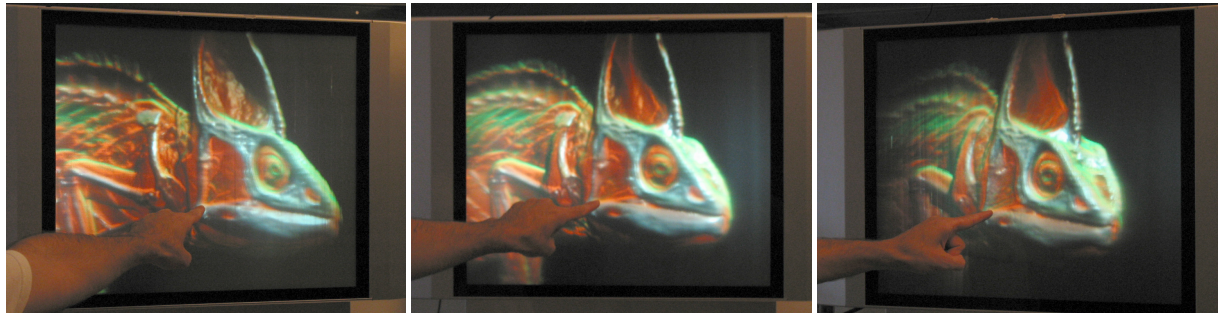
Figure 1: *Volume ray casting of the Chameleon dataset. Images of the light field display taken with a hand held video camera moving in the display workspace. Note the parallax effects and the view-dependent illumination effects.*

problem more tractable. The characteristics of the display are exploited by a specialized rendering technique able to provide multiple freely moving naked-eye viewers the illusion of seeing virtual volumetric objects floating at fixed physical locations in the display workspace (see figure 1). Central to our approach is a GPU ray-caster that follows rays generated by a multiple-center-of-projection (MCOP) technique, while sampling pre-filtered versions of the dataset at resolutions that match the varying spatial accuracy of the display. Our main contributions thus include:

- a general MCOP technique for producing perspective correct images on a class of horizontal parallax only light field displays based on anisotropic diffusers; the solution provides a correct solution for viewers at a known distance and height from the screen and a good approximation for all other positions;
- a GPU accelerated framework implementing volume raycasting on a light field display using a combination of vertex and fragment shaders;
- the description and demonstration of a prototype hardware/software system achieving interactive performance on non-trivial datasets on a single PC configuration;

Although not all the techniques presented here are novel in themselves, their elaboration and combination in a single system is non trivial and represents a substantial enhancement to the state-of-the-art.

2. Related work

Interactive 3D display technology. A huge number of approaches have been proposed to support naked-eye multiscopic visualization, and a full review of the subject is out of scope for this paper. We provide here only a rapid survey of the subject, with a particular emphasis on the most closely related approaches. For a good review on the subject we refer the reader to [Dod05]. The key technical feature characterizing 3D displays is direction-selective light emission, which is obtained most commonly by volumetric, holographic, or multi-view approaches. Volumetric displays synthesize light fields by projecting light beams on refractive/reflective media positioned or moved in space (e.g., [MMMR00,FDHN01,

RS00]). Commercial displays are readily available (e.g., from Actuality Systems). The main disadvantages are the limited scalability of the approach, and the difficulty in presenting occlusion effects. The latter problem has been recently solved in the displays presented by [JMY*07] and [CNH*07], which employ an anisotropic diffuser covering a rapidly spinning mirror illuminated by a single high speed video projector synchronized with mirror rotation. Such a setup allows for 360° viewing, but, because of mechanical constraints, is practical only for limited image sizes and model complexity. Pure holographic techniques are based on generating holographic patterns to reconstruct the light wavefront originating from the displayed object, e.g., using acousto-optic materials [SHLS*95], optically addressed spatial light modulators [SCC*00], or digital micro-mirror devices [HMG03]. Although this approach can theoretically provide the most compelling imagery, the principle itself imposes limitations on realistically achievable image sizes, resolution, speckle, with consequent narrow fields of view, alongside enormous computing capacity required to reach acceptable refresh rates for true interaction. In current prototypes, still confined in research labs, the display hardware is very large in relation to the size of the image (which is typically a few centimeters in each dimension). Typical multi-view displays, often based on an optical mask or a lenticular lens array, show multiple 2D images in multiple zones in space. They support multiple simultaneous viewers, but at the cost of restricting them to be within a limited viewing angle. Multi-view displays are often based on an optical mask or a lenticular lens array. Optical masks introduce significant light loss when there are more than two views. Moreover, the barrier structure becomes visible as the number of views increases. On the other hand, lenticular displays magnify the pixel matrix of the projecting devices creating dark zones between viewing slots. The Cambridge multi-view display is a classic design in this area [DML*00], and a number of manufacturers (Philips [vPF96], Sharp [WHJ*00], Opticality [RR05], Samsung, Stereographics, Zeiss) produce monitors based on variations of this technology. Typical state-of-the-art displays typically use 8–10 images, i.e., directions, at the expense of resolution. Matusik et al. [MP04] demonstrated a proto-

type based on this technology and assembled with sixteen 1024x768 projectors and a lenticular screen. As in our case, the setup requires one projector per view. However, their screen achieves vertical diffusion not by diffusing light vertically from the screen as in our display, but by focusing light horizontally onto a diffuse surface, yielding a different projection geometry. A 3D stereo effect is obtained when the left eye and the right eye see different but matching information. The small number of views of multi-view systems based on masks or lenticulars produces, however, cross-talks and discontinuities upon viewer's motion [Dod96]. The display used in this work [BFA*05], produced by the Holografika company, uses the distributed image generation approach of projector-based multi-view technology, but removes some of the intrinsic optical limitations, as it offers a fully continuous blend among views thanks to the light shaping capabilities of a holographically recorded screen. One limitation of most 3D display solutions, also shared by the display used in this work, is that only horizontal parallax is provided. Yang et al. [YHLJ06] presented an improved system based on a cluster of projectors that provides parallax both horizontally and vertically. This solution provides a more faithful light field reconstruction but requires the generation of a much larger number of rays to achieve the same spatial accuracy, which makes it currently practical only for very small image areas or narrow fields of views.

Projecting graphics to the 3D display. The image generation methods employed in conjunction with 3D displays must take into account the display characteristics both in terms of geometry and resolution of the reproduced light fields. In our work, we employ a multiple-center-of-projection technique to produce images that exhibit correct stereo and motion parallax cues. Our projection algorithm relates to previous work in light field rendering and holography. The multiple-viewpoint-rendering approach [Hal98] harnesses perspective coherence to improve the efficiency of rendering multiple perspective image sequences. Halle et al. [HBKU91] proposed a method where static holographic stereograms account for the viewer's distance but not their height. The light field display recently presented by Jones et al. [JMY*07] also uses a MCOP approach, similar to ours, but their display geometry is radically different. Previous work in rendering for light field displays have used, typically, standard orthographic or perspective projections [RWC*98, CNH*07]. This approach simplifies rendering using a fixed function graphics pipeline but produces perspective distortions when applied to displays based on anisotropic light shaping elements [JMY*07]. None of these works takes into account the finite angular size of the light beams to adapt sampling rates as a function of distance from the screen. A framework for studying sampling and aliasing for 3D displays has recently been proposed by Zwicker et al [ZMDP06].

GPU accelerated volume visualization on multi-view displays. Many sophisticated techniques for real-time volume rendering have been proposed in the past, taking advantage of CPU acceleration techniques, GPU acceleration using texture mapping, or special purpose hardware. In the last few years, improvements in programmability and performance of GPUs have made GPU solutions the main option of choice for real-time rendering. We refer the reader to the recent book of Engel et al. for a recent survey [EHK*06]. Our method is based on the GPU ray-casting approach [KW03, RGW*03]. As in recent single-pass GPU ray-casters [SSKE05], we exploit GPU vertex shaders to render proxy geometry that activates a fragment shader performing the actual ray-casting. Our factorization of the ray computation operations is however different, since our MCOP rendering pipeline cannot rely on the interpolation performed by the rasterizer to pass down combined 3D and projected data from the vertex shader. Our technique for MCOP rendering using GPU shaders is also related to work of Hou et al. [HWSG06], which focuses on simulating reflections and refractions, and Jones et al. [JMY*07], which, however, exploits only vertex shaders for geometry projection. In this work, adaptive performance is obtained by controlling image sizes and sampling rates. Others have proposed acceleration methods for stereo volume rendering, which mainly use the information from one view re-projected to the other [HK96, KLS99, WZQK04]. This results in improved speed, which is however obtained at the cost of image quality [HK96, AH94]. These re-projection approaches are not typically applied to auto-stereoscopic rendering [PPN*00], since they can lead to considerable noise when switching from a view to the next. Since our angular sampling rate is very high ($< 1°$), an adaptation of these approaches to rendering on a light field display is worth exploring.

3. Display concept

The display used in this work is based on projection technology and uses a specially arranged projector array and a holographic screen. We summarize here the main concepts behind it. More information on the technology is presented elsewhere [BFA*05].
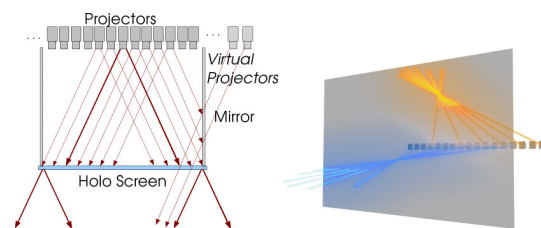
Figure 2: *Display concept. Left: Each projector emits light beams toward a subset of the points of the holographic screen. Side mirrors increase the available light beams count. Right: A large number of light beams can create a spatial point (voxel).*

The projectors are densely arranged behind the screen, and

all of them project their specific image onto the holographic screen to build up a light field (see figure 2). By positioning mirrors at the sides of the display, it is possible to reflect back onto the screen the light beams that would otherwise be lost, thus creating virtual projectors that increase the display field of view. Each projector emits light beams toward a subset of the points of the holographic screen. At the same time, each screen point is hit by more light beams coming from different projectors. The holographic screen is the key element in this design, as it is the optical element enabling selective directional transmission of light beams. It is a holographically recorded, randomized surface relief structure able to provide controlled angular light divergence. The light diffusion characteristic of the screen is the critical parameter influencing the angular resolution of the system, which is very precisely set in accordance with the system geometry. In the horizontal parallax design, the projectors are arranged in a horizontal linear array and the angular light distribution profile induced by the screen is strongly anisotropic. Horizontally, the surface is sharply transmissive, to maintain a sub-degree separation between views. Vertically, the screen scatters widely so the projected image can be viewed from essentially any height. The angular light distribution profile introduced by the holographic screen is characterized by a wide plateau and steep Gaussian slopes precisely overlapping in a narrow region in the horizontal direction. This results in a homogeneous light distribution and continuous 3D view with no visible crosstalk within the field of depth determined by the angular resolution. A full parallax system would be created using a screen with narrow transmission profiles both in the horizontal and vertical direction. This design would require, however, a matrix of projectors for generating a much larger number of rays, significantly increasing the computational cost of image generation. Since humans perceive depth using horizontally-offset eyes and move their viewpoint more easily from side to side than up and down, the horizontal parallax only approach is adequate for most applications and provides significant speed-up.

4. Light field geometry

With proper software control, the light beams leaving the various pixels can be made to propagate in specific directions, as if they were emitted from physical objects at fixed spatial locations. Reconstructing the light field of a rendered scene amounts to precomputing the projection parameters associated to each of the projectors and to using them for generating multiple views for the same image.

Geometric and photometric calibration data can be obtained with many of the existing automated multi-projector calibration techniques [BMY05]. In particular, geometric calibration is straightforward using the classic two-step approach in which position and frustum of each projector are found through parametric optimization of an idealized pinhole model and any remaining error is corrected using a post-rendering 2D image warp that "moves" the pixels to the cor-

rect idealized position. In our case, the parametric model is particularly compact, given the simple geometry of the display. We assume that the screen is centered at the origin with the y axis in the vertical direction, the x axis pointing to the right, and the z axis pointing out of the screen. Each projector is then modeled by a pinhole emitter with origin at $\mathbf{E} = (E_x, E_y, E_z)$ and projecting an image on the plane $z = 0$. The projected image geometry is defined by a 2D rectangle $\mathbf{R}^-, \mathbf{R}^+$ orthogonal to the Z axis. The effect of the lateral mirrors is captured by creating virtual projectors and recovering their parameters in the calibration procedure as done for the real ones.

Projecting graphics to the display. This simple linear perspective model defines how light is projected onto the screen, but is not sufficient to define how a 3D graphics application should project their models to the display, because it ignores the transformation performed by the holographic screen. Since the screen is selective only in the horizontal direction, but scatters widely in the vertical one, the displayed light field's dimensionality is reduced, and the application must decide how to deal with the missing degree of freedom. In practice, at any moment in time, a given screen pixel has the same color when viewed from all vertical viewing angles. In order to provide a full perspective effect, the vertical viewing angle must thus be known, which amounts at fixing the viewer's height and distance from screen.

Therefore, the renderer assumes a virtual viewer at height V_y and distance V_z from the screen. Given a world space point $\mathbf{P}$, its screen projected position $\mathbf{S}$ for a given emitter E can thus be computed for the x coordinate by intersecting the ray originating from the emitter E with the screen plane at $z = 0$ and for the y coordinate by intersecting it with the ray arriving to the virtual viewer $\mathbf{V}$ (see figure 3):

$$
\begin{aligned}
S_x &= E_x - E_z \cdot \frac{E_x - P_x}{E_z - P_z} \\
S_y &= V_y - V_z \cdot \frac{V_y - P_y}{V_z - P_z}
\end{aligned}
\tag{1}
$$

For rendering, e.g., in a rasterization application, the world space position $\mathbf{S}$ is remapped to normalized projected coordinates by transforming to the image rectangle and associating a depth (for Z-buffering) based on distance to the screen:

$$
\begin{aligned}
H_x &= \frac{(S_x - E_x) \cdot 2 - (R_x^+ + R_x^-)}{R_x^+ - R_x^-} \\
H_y &= \frac{(S_y - E_y) \cdot 2 - (R_y^+ + R_y^-)}{R_y^+ - R_y^-} \\
H_z &= -P_z/V_z
\end{aligned}
\tag{2}
$$

With these few operations, we can determine for any 3D point $\mathbf{P}$ where it should be drawn on a given projector to produce a perspective correct image. The solution is exact for all viewers at the same distance from screen and height
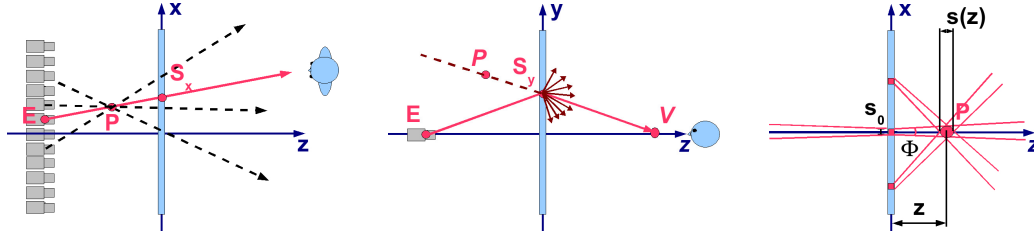
Figure 3: *Light field geometry: Left: horizontally, the screen is sharply transmissive and maintains separation between views. Center: vertically, the screen scatters widely so the projected image can be viewed from essentially any height. Right: the finite angular size of the light beams determines the voxel dimension as a function of distance from the screen.*

as the virtual observer and proves in practice to be a good approximation for all other viewing positions in the display workspace. A similar approach is taken by Jones et al. [JMY*07] for their 360° light field display. Their approach also employs a MCOP perspective that combines two different viewpoints **P** (for horizontal coordinates) and **V** (for vertical coordinates). In their case, however, the geometry of the display is significantly more complex, and computing the projection requires both a ray/plane and a ray/cylinder intersection per 3D point.

Depth dependent spatial resolution. The display design has consequences not only on the projection equation but also imposes limits on spatial resolution that depends on depth. Each beam leaving the screen has (approximately) a finite angular size Φ. This, however, introduces a finite resolution effect – that is independent from the screen pixel resolution – in the reconstructed three dimensional scene. In fact, the size s of the smallest voxel that can be reproduced depends on the distance of its center from the screen and from the beam angular size Φ, and can be approximated by

$$s(z) = s_0 + 2\|z\| \tan(\Phi/2) \qquad (3)$$

where s_0 is the pixel size on the screen surface (see figure 3 right). In other words, the achievable spatial resolution decreases with the distance from the screen. This is intuitive because the illusion of existence of a particular spatial point is generated by pyramidal beams crossing at a specific 3D position (see figure 3 right). This fact dictates how volumes should be sampled when rendering for the display and also practically limits the field-of-depth of the display, i.e., the maximum distance from the screen at which objects are faithfully reconstructed.

5. GPU-based Volume Ray Casting

We exploit the characteristics of the display to develop a specialized volume rendering technique able to provide multiple freely moving naked-eye viewers the illusion of presence of virtual volumetric objects floating at fixed physical locations in the display workspace. Our light-field display aware volume rendering process follows the two-pass approach typical of contemporary multi-projector displays [BMY05] (see figure 4). In the first rendering pass, a per projector view of the
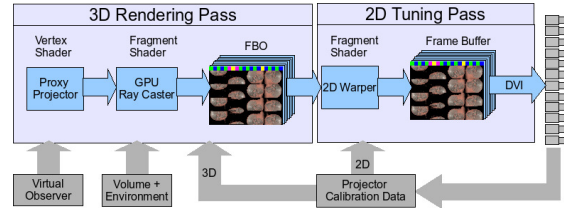
Figure 4: *Light field display aware volume rendering process: In a first 3D rendering pass a volume ray-caster executed entirely on the GPU generates a view per projector using a small linear model for projector alignment. Non-linear view distortions and color correction is applied in a second warping pass.*

scene is rendered off-screen to a frame buffer object using the idealized light field geometry model. In the second pass, performed purely in 2D image space, small non-linear view and color distortions are corrected by streaming the first pass texture through a fragment shader that warps the geometry and modifies colors thanks to per-pixel lookup tables stored as precomputed textures.

Our 3D rendering framework is based on GPU ray-casting, which has recently emerged as a flexible and efficient framework for real-time volume rendering. In this approach, per projector images of a volume are rendered by casting rays through each pixel, and performing resampling and compositing of successive sample contributions along these rays. The entire volume is stored in a single 3D texture, and resampling is performed by fetching samples from this texture with trilinear texture filtering. For each pixel, a fragment program steps from sample location to sample location in a loop and performs compositing until the volume is exited or full opacity is reached. By applying an appropriate optical model many desired kinds of interaction between light and the volumetric object can be realized.

Fragment generation. The GPU ray-caster is activated by rendering a proxy geometry whose projection on the image plane covers all pixels emitting rays passing through the volume. The classic approach is to render the volume's bounding box, with per vertex attributes storing the corresponding

ray entry and exit points in the volume. Since our display requires a MCOP perspective, we cannot rely on the standard fixed-function pipeline to perform this step, since the MCOP cannot be recast into the traditional homogeneous matrix. In addition, long straight lines appear curved in the projection, and linear interpolation of vertex attributes does not produce correct per-pixel ray parameter values. Thus, we use as proxy a coarsely tessellated version (8x8 quads) of a slightly enlarged bounding volume and stream its vertices through a vertex shader that performs our MCOP projection using equations 1 and 2 and passes down to the fragment stage the normalized and unnormalized projected coordinates.

Ray computation and compositing. In order to be able to step along rays, the ray entry position, direction, and length must be computed for each fragment. Our shader receives from the vertex stage an unnormalized projected position **S** computed from equation 1, which represents the screen pixel's position expressed in the display reference frame. The direction **d** of the ray passing through that point can then be computed using the MCOP projection parameterized by emitter E and virtual viewer parameters **V** (see figure 3):

$$\mathbf{d} = \left[\frac{S_x - E_x}{E_z}, \frac{S_y - V_y}{V_z}, -1\right] \quad (4)$$

The infinite ray defined by **S**, **d** is then transformed in local volume coordinates, assumed to be coincident with texture coordinates, by applying a world-to-texture homogeneous transform matrix stored in the OpenGL model-view register. Once the ray is transformed in the local texture coordinates, the ray entry point and integration lengths are computed by clipping the line **S**, **d** against the unit box. If the length is null, the fragment is considered invalid and discarded, otherwise the renderer steps through the ray in a loop and performs volume resampling and compositing until the volume is exited or full opacity is reached. Since spatial resolution is depth dependent, special care must be taken in this step to match resampling and integration kernels with actual representable voxel size (see section 4).

In our approach, we store the volume texture in a Mipmap pyramid and access it by controlling directly the Mipmap access through the *tex3Dlod* intrinsic. At each sampling step, we compute the depth of the current sample by transforming its position in texture space to world (screen aligned) space. Since we are interested only in the single z coordinate this transformation can be performed using a single dot product. The most appropriate mipmap level l is then chosen by considering the spatial resolution at depth z from the screen finding the coarsest level with a sufficiently small voxel size. This can be done by the following equation:

$$l = median(0, L_{max}, \lfloor \log_2 \frac{s(z)}{v_0} \rfloor) \quad (5)$$

where $s(z)$ is computed according to equation 3, L_{max} is the maximum available Mipmap level and v_0 is the voxel size at the finest Mipmap level. Adaptive stepping is also possible

by using $s(z)$ as variable step-size. Additional optimizations, in particular empty space skipping, can be implemented on top of this basic approach. Such optimizations are orthogonal to this work and not considered in this paper.

In appendix A, we provide the Cg fragment program source code containing main ray-caster steps. The code assumes that helper functions are defined for basic geometric operations and leaves undefined the actual compositing technique.

6. Implementation and results

We have implemented a prototype hardware and software system based on the design discussed in this paper. The software system proposed in this paper consists of a framework written in C++ and OpenGL, a set of Cg shaders that implement the basic ray-casting engine, and a number of shader functions that implement different compositing techniques. The display hardware is manufactured by Holografika and is capable of visualizing 7.4M beams/frame by composing optical module images generated by 96 fast 320x240 LCD displays fed by FPGA input processing units that decode an input DVI stream. The on screen 2D pixel size of the display is $s_0 = 1.25mm$, and the angular accuracy is $0.8°$.

The DVI channel feeding the display works at 1280x1024 at 75Hz. Each 1280x1024 frame collects 16 320x240 projector images, plus a color-encoded header in the top rows that encodes the ids of the projectors that have to be updated. A full 3D frame is created by sequentially generating all the projector images into the frame buffer. In this work, the graphics application runs on an Athlon64 3300+ PC with a NVIDIA8800GTX graphics board working in twin-view mode. One DVI output is used for control on a 2D monitor, while the second one feeds the 3D display.

Interactive sequences. It is obviously impossible to fully convey the impression provided by our holographic environment on paper or video. As a simple illustration of our system's current status and capabilities, an accompanying video shows interactive sequences recorded live using a moving camera. Representative video frames are shown in figure 5. The sequences were recorded with a hand held video camera freely moving in the display workspace. In order to assess the distortion caused by the MCOP approach, the video includes sequences presenting both vertical and horizontal motions, with the camera moving far from the virtual viewer position. We recorded short inspections and free-hand manipulation of different public domain datasets at the resolution of 256^3 voxels at 8bit/sample: The Visible Human Male head [†], the Chameleon CT scan [‡], and a contrasted rotational angiogra-

---

Figure 5: *Frames from live recordings. These images, taken from the accompanying video, show successive instants of interactive exploration of different CT datasets.*

phy of a human head with aneurysm §. Currently, our prototype volume ray caster implements a number of composition strategies, that include Maximum Intensity Projection (MIP), Simulated X-Ray, as well as Direct Volume Rendering with a Phong illumination model, boundary enhancement and view-dependent transparency [BG07]. For rendering, the volume is stored in a single 4 component texture that contains the un-scaled precomputed gradient in the RGB part and the voxel value in the A part. An additional one-dimensional RGBA texture contains an interactively modifiable look-up table. In the accompanying video, the Chameleon and Visible Human head datasets are rendered using a shaded volume rendering, while the rotational angiography dataset employs a maximum intensity projection.

As demonstrated in the video, objects appear to moving viewers floating in the display space and can be manipulated by translating, rotating, and scaling them with a six degree of freedom tracker, as well as by modifying the transfer function. Note the parallax effects and the good registration between displayed object space and physical space, which demonstrate the multi-user capability of the display and the

good performance of the MCOP viewing approach. Please also note that specular highlights correctly follow the recording camera's viewpoint, contributing to volume readability. As illustrated by the video, the perceived image is fully continuous. This is qualitatively very different from other contemporary multiview technologies, which force users into approximately fixed positions, because of the abrupt view-image changes that appear when crossing discrete viewing zones [MP04].

The main limiting factor during interaction was given by the single GPU volume renderer performance, since at full resolution (1 ray/pixel and 1 sample/voxel), many millions of rays need to be propagated for hundreds of steps through the volume. In order to improve interactive frame rates, we have thus chosen the solution of lowering the sampling rate during interaction, by reducing the pixel count (2.5mm precision on screen) and doubling the integration step-size. Since the display projectors are sequentially updated in batches of 16 320x240 images, a misalignment between tiles can become visible when objects are moved with a too slow refresh rate (see figure 6). It is important to note that even when a single "static" 3D view is displayed, users can exploit accommodation, stereo and motion parallax to gain understanding of complex shapes. Some of these cues can be also obtained with traditional systems, but only by incorporating interactive

---

§ Source: Volume Dataset Repository at the WSI/GRIS, University of Tübingen, Germany

manipulation in the rendering system. In that case, users will have to move the object or the viewpoint to provide the visual system with enough depth information. The task is not simple and immediate, and depth information is easily lost when the user stops interacting.
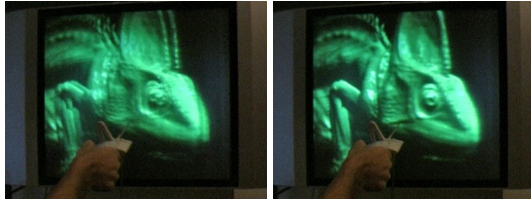

Figure 6: *Dynamic tile misalignment effect. The 96 display projectors are updated in batches of 16 projectors each, which can lead to artifacts in dynamic motion if the rendering rate is too slow (left). These artifacts disappear when the image is static (right).*

Enhanced 3D understanding. To quickly evaluate whether the light field rendering infrastructure is able to help in rapidly gaining understanding of complex 3D shapes we performed two simple experiments.
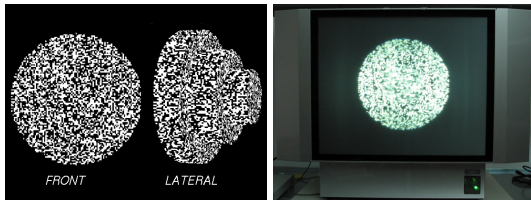
Figure 7: *Random dot masking test. Left: Stimulus. Right: central view as presented by the display.*

In a first synthetic benchmark using random-dot masking, we employed a simplified version of Julesz's spiral ramp surface: a 3-layer cylindrical wedding-cake model that subjects viewed along its concentric axis (see figure 7). By adjusting the model's parameters and converting it to a rectilinear volume, two sets of model-stimuli were rendered using our framework: one with a uniform large field of depth ($\pm 10cm$ centered on the display screen) and one where the field was almost flat ($\pm 1cm$). Each of eleven, pre-screened, subjects completed four experiments, each consisting of eight trials in a two-interval forced-choice (2IFC) design whereby they indicated in which interval they perceived the greatest field of depth. The experiments tested one-eye static, one-eye head-swaying, two-eye static, and two-eye head-swaying observation in that order. Scores improved also in that order: from 49%, i.e., indistinguishable from a random answer in the binary test, with a monocular static view, to 82% correct scores for the monocular head-swaying test, up to 100% when all cues are available. The results indicate that in the absence of cues mundanely available to 2D displays the light field rendering system elicits useful stereoscopic and motion-parallax depth cues.
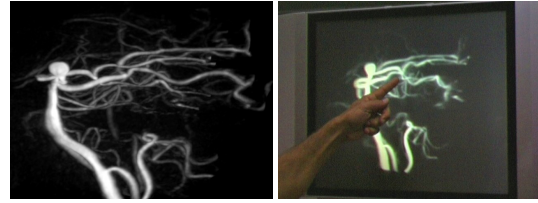
Figure 8: *MIP volume rendering of rotational angiography scan of a head with aneurysm. Left: Since the technique is depth-oblivious, the position and crossing of vascular structures is not detectable. Right: the light field display provides rapid volumetric understanding.*

In a second, less formalized, test, we consider a rotational angiography scan of a head with aneurysm rendered using the commonly employed MIP modality. MIP is a simple volume rendering composing approach where the maximum value encountered along a ray is used to determine the color of the corresponding pixel. It is considered very useful for visualizing angiography data sets, since the data values of vascular structures with contrast agent are higher than the values of the surrounding tissues and are therefore not masked by other data. From a single MIP image, however, recovering 3D structures is impossible, since the technique does not provide adequate depth cues (see figure 8). That's why current best medical practices suggest to always correlate MIP images with direct volume-rendered images, in order to demonstrate 3D relationships [FNH*06]. This dual representation approach, however, adds complexity to the image analysis tasks, since MIP images, besides being more familiar to the medical community, require also substantially less parameter tuning and produce results which are more reproducible by different operators/systems. When looking at the test angiography dataset on our light field display, we verified that users are able to very rapidly recover all depth cues and to instantaneously recognize the vascular structure because of the combination of stereo and motion parallax. Previous work has verified the importance of stereopsis for this task [ME04], as well as the importance of dynamic cues [BBD*07]. Our system seamlessly combines both cues, and, in addition achieves parallax effects through ego-motion rather than interaction.

7. Conclusions and Future Work

The prototype discussed here is clearly meant to work as an enabling technology demonstrator, as well as a testbed for integrated volumetric rendering and light field display research. A first conclusion than can be drawn from our work is that high quality volumetric rendering on light field displays is currently achievable even when using single GPU desktop solution for the rendering task. In order to deal with large scale datasets, we are already working on a out-of-core parallel multi-GPU renderer. We are also investigating how to improve rendering speed by incorporating optimization that exploit the high degree of coherence present in a 3D view. There is obviously more to interactive volumetric rendering

on 3D displays than employing standard composition techniques. Devising which of the recently proposed expressive visualization techniques are appropriate for 3D displays is beyond the scope of this paper, which focuses on enabling rendering technology. Exploring this domain through the design and implementation of highly interactive techniques that leverage the unique features of an interactive multi-viewer 3D environment is a challenging area for future work. Future work will also focus on perceptual evaluation of the 3D display with respect to depth discrimination tasks. In this context, we also plan to evaluate the effects induced by our multiple-center-of-projection approximation.

References

[AH94] ADELSON S., HANSEN C.: Fast stereoscopic images with Ray-Traced volume rendering. In *Symposium on Volume Visualization* (1994), pp. 3–10.

[BBD*07] BOUCHENY C., BONNEAU G.-P., DROULEZ J., THIBAULT G., PLOIX S.: A perceptive evaluation of volume rendering techniques. In *Proc. ACM APGV* (2007), pp. 83–90.

[BFA*05] BALOGH T., FORGÁCS T., AGOCS T., BALET O., BOUVIER E., BETTIO F., GOBBETTI E., ZANETTI G.: A scalable hardware and software system for the holographic display of interactive graphics applications. In *Eurographics Short Papers Proceedings* (2005), pp. 109–112.

[BG07] BRUCKNER S., GRÖLLER M. E.: Style transfer functions for illustrative volume rendering. *Computer Graphics Forum 26*, 3 (Sept. 2007), 715–724.

[BMY05] BROWN M., MAJUMDER A., YANG R.: Camera-based calibration techniques for seamless multi-projector displays. *IEEE Trans. Vis. Comput. Graph 11*, 2 (2005), 193–206.

[CNH*07] COSSAIRT O., NAPOLI J., HILL S., DORVAL R., FAVALORA G.: Occlusion-capable multiview volumetric three-dimensional display. *Applied Optics 46*, 8 (Mar. 2007), 1244–1250.

[DML*00] DODGSON N. A., MOORE J. R., LANG S. R., MARTIN G., CANEPA P.: Time-sequential multi-projector autostereoscopic 3D display. *J. Soc. for Information Display 8*, 2 (2000), 169–176.

[Dod96] DODGSON N. A.: Analysis of the viewing zone of the Cambridge autostereoscopic display. *Applied Optics: Optical Technology & Biomedical Optics 35*, 10 (1996), 1705–1710.

[Dod05] DODGSON N. A.: Autostereoscopic 3D Display. *Computer 38*, 8 (2005), 31–36.

[EHK*06] ENGEL K., HADWIGER M., KNISS J., REZK-SALAMA C., WEISKOPF D.: *Real-time Volume Graphics*. AK-Peters, 2006.

[FDHN01] FAVALORA G., DORVAL R., HALL D., NAPOLI J.: Volumetric three-dimensional display system with rasterization hardware. In *Proc. SPIE* (2001), vol. 4297, pp. 227–235.

[FNH*06] FISHMAN E. K., NEY D. R., HEATH D. G., CORL F. M., HORTON K. M., JOHNSON P. D.: Volume rendering versus maximum intensity projection in CT Angiography: What works best, when, and why. *RadioGraphics 2006 26*, 3 (May–June 2006), 905–922.

[GTH05] GHOSH A., TRENTACOSTE M., HEIDRICH W.: Volume rendering for high dynamic range displays. In *Eurographics/IEEE VGTC Workshop on Volume Graphics* (2005), pp. 91–98.

[Hal98] HALLE M.: Multiple viewpoint rendering. In *Proc. SIGGRAPH* (1998), pp. 243–254.

[HBKU91] HALLE M. W., BENTON S. A., KLUG M. A., UNDERKOFFLER J. S.: Ultragram: a generalized holographic stereogram. In *Proc. SPIE* (July 1991), vol. 1461, pp. 142–155.

[HK96] HE T., KAUFMAN A.: Fast stereo volume rendering. In *Proc. Visualization* (1996), pp. 49–ff.

[HMG03] HUEBSCHMAN M., MUNJULURI B., GARNER H.: Dynamic holographic 3-d image projection. *Optics Express 11* (2003), 437–445.

[HWSG06] HOU X., WEI L.-Y., SHUM H.-Y., GUO B.: Real-time multi-perspective rendering on graphics hardware. In *Proc. 17th Eurographics Workshop on Rendering* (2006), pp. 93–102.

[JMY*07] JONES A., MCDOWALL I., YAMADA H., BOLAS M. T., DEBEVEC P. E.: Rendering for an interactive 360 degree light field display. *ACM Trans. Graph 26*, 3 (2007), 40.

[KLS99] KOO Y.-M., LEE C.-H., SHIN Y.-G.: Object-order template-based approach for stereoscopic volume rendering. *Journal of Visualization and Computer Animation 10*, 3 (1999), 133–142.

[KW03] KRUEGER J., WESTERMANN R.: Acceleration techniques for GPU-based volume rendering. In *Proc. Visualization* (2003), pp. 287–292.

[ME04] MORA B., EBERT D. S.: Instant volumetric understanding with order-independent volume rendering. *Computer Graphics Forum 23*, 3 (2004), 489–497.

[MMMR00] MCKAY S., MAIR G., MASON S., REVIE K.: Membrane-mirror based autostereoscopic display for teleoperation and telepresence applications. In *Proc. SPIE* (2000), vol. 3957, pp. 198–207.

[MP04] MATUSIK W., PFISTER H.: 3D TV: a scalable system for real-time acquisition, transmission, and autostereo-

scopic display of dynamic scenes. *ACM Transactions on Graphics 23*, 3 (Aug. 2004), 814–824.

[PPN*00] PORTONI L., PATAK A., NOIRARD P., GROS-SETIE J.-C., VAN BERKEL C.: Real-time auto-stereoscopic visualization of 3D medical images. In *Proc. SPIE* (Apr. 2000), Mun S. K., (Ed.), vol. 3976, pp. 37–44.

[RGW*03] ROETTGER S., GUTHE S., WEISKOPF D., ERTL T., STRASSER W.: Smart hardware-accelerated volume rendering. In *Proc. VISSYM* (2003), pp. 231–238.

[RR05] RELKE I., RIEMANN B.: Three-dimensional multiview large projection system. In *Proc. SPIE* (2005), vol. 5664.

[RS00] ROBERTS J. W., SLATTERY O.: Display characteristics and the impact on usability for stereo. In *Proc. SPIE* (2000), vol. 3957, p. 128.

[RWC*98] RASKAR R., WELCH G., CUTTS M., LAKE A., STESIN L., FUCHS H.: The office of the future: A unified approach to image-based modeling and spatially immersive displays. In *Proc. SIGGRAPH* (1998), pp. 179–188.

[SCC*00] STANLEY M., CONWAY P., COOMBER S., JONES J., SCATTERGOOD D., SLINGER C., BANNISTER B., BROWN C., CROSSLAND W., TRAVIS A.: A novel electro-optic modulator system for the production of dynamic images from giga-pixel computer generated holograms. In *Proc. SPIE* (2000), vol. 3956, pp. 13–22.

[SHLS*95] ST.-HILLAIRE P., LUCENTE M., SUTTER J., PAPPU R., SPARRELL C. G., BENTON S.: Scaling up the mit holographic video system. In *Proc. 5th SPIE Symposium on Display Holography* (1995), pp. 374–380.

[SSKE05] STEGMAIER S., STRENGERT M., KLEIN T., ERTL T.: A simple and flexible volume rendering framework for graphics-hardware-based raycasting. In *Eurographics/IEEE VGTC Workshop on Volume Graphics* (2005), pp. 187–195.

[vPF96] VAN BERKEL C., PARKER D., FRANKLIN A.: Multiview 3d-lcd. In *Proc. SPIE* (1996), vol. 2653, p. 32.

[WHJ*00] WOODGATE G. J., HARROLD J., JACOBS A. M. S., MOSELEY R. R., EZRA D.: Flat-panel autostereoscopic displays: characterisation and enhancement. In *Proc. SPIE* (2000), vol. 3957, p. 153.

[WZQK04] WAN M., ZHANG N., QU H., KAUFMAN A. E.: Interactive stereoscopic rendering of volumetric environments. *IEEE Transactions on Visualization and Computer Graphics 10*, 1 (2004), 15–28.

[YHLJ06] YANG R., HUANG X., LI S., JAYNES C.: Toward the light field display: Autostereoscopic rendering via a cluster of projectors. In *Eurographics Short Papers Proceedings* (2006).

[ZMDP06] ZWICKER M., MATUSIK W., DURAND F., PFISTER H.: Antialiasing for automultiscopic 3D displays.

In *Proc. Eurographics Symposium on Rendering* (2006), pp. 73–82.

Appendix A: Fragment shader code

The following fragment shader code illustrates our technique for GPU ray casting on the 3D display. The code assumes that helper functions are defined for basic geometric operations and leaves undefined the actual compositing technique.

```
uniform float3 Ew;      // Projector position
uniform float  Vd;      // Virtual observer distance
uniform float  Vh;      // Virtual observer height
uniform float  log2V;   // log2 voxel size
uniform float  DL;      // stepsize
uniform float  S0;      // screen pixel size
uniform float  tanPhi; // angular spread factor

struct fragment_out { float4 color: COLOR0; };

struct ray_t { float3 org; float3 dir; float len; };

ray_t ray_at_xy(float x, float y, float4x4 mv_inv) {
  ray_t result;
  // Compute ray passing through pixel in world coords
  float4 ray_o = float4(x, y, 0, 0);
  float4 ray_d = float4(-(Ew.x-x)/Ew.z, (Vh-y)/Vd, -1, 0);

  // Transform from world to texture coordinates
  float3 tex_o = mul(mv_inv, ray_o).xyz;
  float3 tex_e = mul(mv_inv, ray_o+ray_d).xyz;

  // Clip ray to volume box
  result.dir = normalize(tex_e-tex_o);
  float2 tmin_tmax = clip_to_unit_box(tex_o, result.dir);
  result.org = tex_o+result.dir*tmin_tmax.x;
  result.len = tmin_tmax.y-tmin_tmax.x;

  return result;
}

fragment_out fragment_shader(vertex_out IN) {
  fragment_out OUT;

  // Compute ray in tex coords
  ray_t ray = ray_at_xy(IN.Qw.x, IN.Qw.y,
                  glstate.matrix.inverse.modelview[0]);
  if (ray.len < Epsilon) {
    // Out of bounds
    discard;
  } else {
    // Ray composition
    float3 delta_dir = ray.dir*DL;  int N = int(ray.len/DL);
    float3 uvw=ray.org; float3 I=float3(0,0,0); float a=0.0;
    for (int i = 0; (a < a_max) && (i < N); ++i) {
      // Find distance from screen
      float z = abs(dot(glstate.matrix.modelview[0][2],
                float4(uvw.xyz, 1.0)));

      // Compute mipmap level and sample texture
      float mipmap_l=log2(S0+2*tanPhi*z)-log2V;
      float4 gggv = mipmap_value(uvw.xyz, mipmap_l);

      // Apply transfer function and shading
      C_i = ... ; a_i = ...;

      // Accumulation and stepping
      I = I + (1.0-a)*C_i; a = a + (1.0-a)*a_i;
      uvw += delta_dir;
    }
    OUT.color = float4(I,a);
    return OUT;
  }
}
```