Immersive Molecular Visualization with Omnidirectional Stereoscopic Ray Tracing and Remote Rendering

John E. Stone
Beckman Institute
University of Illinois at Urbana-Champaign
Urbana, IL, USA
Email: johns@ks.uiuc.edu

William R. Sherman
Pervasive Technology Institute
Indiana University
Bloomington, IN, USA
Email: shermanw@indiana.edu

Klaus Schulten
Department of Physics
University of Illinois at Urbana-Champaign
Urbana, IL, USA
Email: kschulte@ks.uiuc.edu

***Abstract*—Immersive molecular visualization provides the viewer with intuitive perception of complex structures and spatial relationships that are of critical interest to structural biologists. The recent availability of commodity head mounted displays (HMDs) provides a compelling opportunity for widespread adoption of immersive visualization by molecular scientists, but HMDs pose additional challenges due to the need for low-latency, high-frame-rate rendering. State-of-the-art molecular dynamics simulations produce terabytes of data that can be impractical to transfer from remote supercomputers, necessitating routine use of remote visualization. Hardware-accelerated video encoding has profoundly increased frame rates and image resolution for remote visualization, however round-trip network latencies would cause simulator sickness when using HMDs. We present a novel two-phase rendering approach that overcomes network latencies with the combination of omnidirectional stereoscopic progressive ray tracing and high performance rasterization, and its implementation within VMD, a widely used molecular visualization and analysis tool. The new rendering approach enables immersive molecular visualization with rendering techniques such as shadows, ambient occlusion lighting, depth-of-field, and high quality transparency, that are particularly helpful for the study of large biomolecular complexes. We describe ray tracing algorithms that are used to optimize interactivity and quality, and we report key performance metrics of the system. The new techniques can also benefit many other application domains.**

***Keywords*-immersive visualization; molecular visualization; remote visualization; ray tracing; head mounted displays;**

I. Introduction

Immersive stereoscopic visualization can provide the viewer with an intuitive perception of complex spatial relationships that are of critical interest for many application domains within science and engineering. The molecular visualization field has historically been an early adopter of high-quality rendering algorithms such as ray tracing (RT), ambient occlusion (AO) lighting, stereoscopic displays, and immersive visualization techniques, as a means of enhancing the perception of shape and for spatial reasoning — helpful for the study of the mechanics of large biomolecular complexes [1]–[4]. A continuing series of advances in graphics algorithms, accelerator hardware, and display hardware have enabled a progression of immersive visualization techniques over the past three decades.

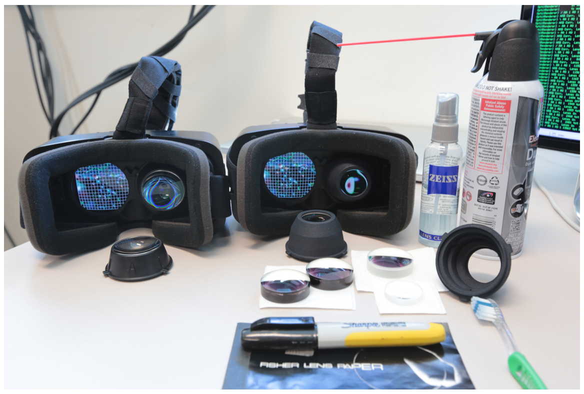
Fig. 1. Photograph of VMD molecular scene on Oculus DK2 HMDs with stock oculars (left) and new high quality oculars with self-assembled optics (right). The HMDs are shown with left ocular removed, allowing the OLED display panel to be seen, showing the OSP stereo perspective reprojection with the HMD lens geometric distortion correction mesh superimposed.

The widely used molecular visualization and analysis tool VMD was originally developed for immersive visualization in the CAVE, but evolved to become a general purpose tool used on mainstream workstations [5]–[8] and petascale computers [9]. Further, VMD incorporates extensive support for high quality rendering with built-in ray tracing engines going back over a decade [10], and recent VMD releases support interactive ray tracing on many-core CPUs, GPUs, and visualization clusters [1], [4].

The recent availability of commodity head mounted displays (HMDs), see Fig. 1, and mobile-phone based counterparts provide a compelling opportunity for widespread use of immersive visualization by the scientific community. However, HMDs pose significant challenges for visualization software due to the combined need for low-latency and high-frame-rate rendering to prevent the onset of so-called simulator sickness.

Analysis of biomolecular structures benefits from sophisticated lighting and shading techniques such as ambient occlusion lighting [11], [12], depth-of-field focal blur, motion blur, translucency [13], realistic camera models [14], and the reflection and refraction effects that have long been hallmarks of RT techniques. The flexibility of RT enables the use of these and many more techniques, including correctly-shaded AO for stereoscopic images, for arbitrary panoramic projections and camera models, which are beyond the capabilities of conventional rasterization-based approaches.

The ever growing size and detail of biomolecular structures obtained by experimental imaging has led to state-of-the-art hybrid structure fitting methods that involve molecular dynamics simulations encompassing over 100 million atoms, which produce multi-terabyte trajectories. Routine visualization and analysis activities of molecular scientists require access to the output of ongoing simulations as they progress, but the regular transfer of such large amounts of data is impractical [4], [9]. The incorporation of hardware-accelerated video encoding in modern CPUs and GPUs permits high resolution interactive visualization between powerful remote visualization servers and comparatively modest client computers, including tablets and mobile phones. However, round-trip network latencies commonly exceed the thresholds required to prevent simulator sickness when using head mounted displays.

We have implemented a unique two-phase rendering system that combines omnidirectional stereoscopic RT with high performance view-dependent rasterization to provide one or multiple users with high-quality immersive visualization. The system overcomes network latencies and provides high-fidelity rendering previously inaccessible within immersive visualization systems. We describe our RT algorithms used to optimize interactivity while maintaining high quality rendering, and report key performance metrics of the system. The key contributions of our approach are:

- **Omnidirectional stereoscopic projection:** We present a projection approach that supports widely used spherical and cubic texture mapping operations in commodity graphics hardware, enabling high frame rate (over 100 Hz) view-dependent reprojection to create stereoscopic perspective renderings for HMDs, CAVEs, and other immersive displays.
- **Interactive progressive RT engine for omnidirectional stereoscopic projections:** We present design and implementation elements of our high performance progressive RT engine based on data-parallel CUDA kernels running on local GPUs or remote GPU clusters. We describe algorithm design and performance considerations for features such as ambient occlusion lighting, fog and depth cueing, and depth of field focal blur in the context of omnidirectional RT for molecular visualization.
- **Complete system for remotely-rendered immersive molecular visualization with HMDs:** We present a novel combination of omnidirectional stereo projection, interactive progressive RT, remote H.264 video streaming, and multi-GPU parallel RT with high-frame-rate local rasterization and lens distortion correction to facilitate high-fidelity immersive visualization of complex molecular scenes on commodity head mounted displays. We evaluate the remote rendering system using a GPU cluster located on the public internet, 2,100 miles from the visualization workstation and an attached HMD.
- **Effective immersive molecular visualization with omnidirectional projections and advanced lighting:** We describe several of the unique challenges that arise when combining high-fidelity lighting and rendering techniques with densely packed molecular scenes rendered with omnidirectional projections.

II. Related Work

Panoramic rendering of molecular graphics has long been a topic of interest for creating presentations for full-dome theaters and other panoramic projection systems, beginning with the earliest work by Max [15], [16] for IMAX and OMNIMAX, and extended by Greene et al. with reprojection and filtering of four-faced cubic projections for OMNIMAX [17]. More recently Simon et al. [18] and Bourke [19] describe techniques for creating omnidirectional stereoscopic images synthesized from a large number of perspective images with subsequent image warping and reprojection, employing conventional low-cost shading techniques.

Ray tracing is well known for its capacity to support sophisticated projections, camera models, and advanced lighting techniques. The incorporation of panoramic projections and lens distortions into offline RT algorithms was described by Max [16]. With recent technological advances, interactive RT has become possible on commodity hardware [20]–[23], leading to its incorporation in visualization software. Krone et al. use GPU ray casting within rasterization-based molecular visualization tools for high quality display of molecular surfaces consisting of spheres, and spherical and toroidal patches [24]. Later efforts employ ray casting for other curved surfaces such as cylinders [25]. Marsalek et al. [26] describe the incorporation of interactive RT into their molecular visualization tool, enabling high quality rendering of curved surfaces, hard shadows, and high quality transparent surface rendering for conventional perspective projections. Recently, Knoll et al. and Reda et al. describe parallel CPU and GPU RT approaches for interactive visualization of large materials-science simulations on a variety of high resolution and immersive tiled displays, incorporating hard shadows and high quality transparency [27]–[29]. Reiners et al. have recently described a stereoscopic simulation system based on interactive GPU-accelerated RT [30]. Their system implements hard shadows, reflections, and other classical RT features, and achieves frame rates suitable for use in virtual environments.

Our work differs from the RT based molecular visualization systems above because our approach implements advanced techniques such as ambient occlusion lighting, depth of field focal blur, and uses parallel stochastic sampling and progressive image refinement to accumulate the substantially larger numbers of samples (as much as $200\times$) required for image convergence when rendering scenes that incorporate these effects. We are unaware of any previously published work that combines immersive visualization techniques with high quality progressive RT, nor RT-based systems that can support remote rendering scenarios while also achieving the performance levels required for comfortable use of HMDs.

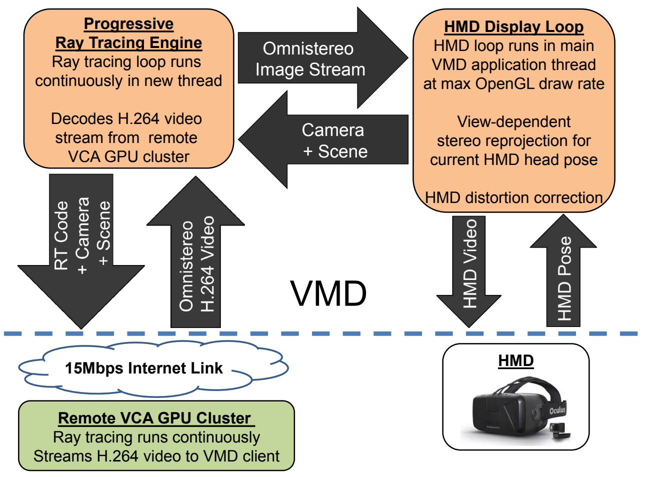
Progressive
Ray Tracing Engine
Ray tracing loop runs continuously in new thread
Decodes H.264 video stream from remote VCA GPU cluster
Omnistereo Image Stream
Camera + Scene
HMD Display Loop
HMD loop runs in main VMD application thread at max OpenGL draw rate
View-dependent stereo reprojection for current HMD head pose
HMD distortion correction
RT Code + Camera + Scene
Omnistereo H.264 Video
VMD
HMD Video
HMD Pose
15Mbps Internet Link
HMD
Remote VCA GPU Cluster
Ray tracing runs continuously
Streams H.264 video to VMD client

Fig. 2. Overview of major rendering system components and data flow. The orange progressive ray tracing engine and HMD display loop blocks are both implemented within the molecular visualization application.

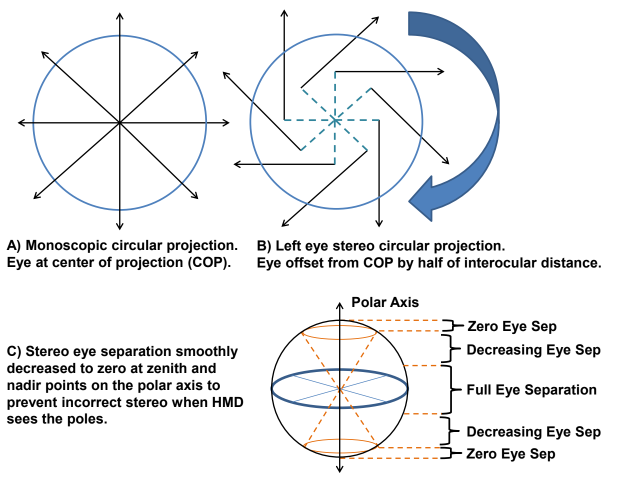
A) Monoscopic circular projection. Eye at center of projection (COP).
B) Left eye stereo circular projection. Eye offset from COP by half of interocular distance.
Polar Axis
C) Stereo eye separation smoothly decreased to zero at zenith and nadir points on the polar axis to prevent incorrect stereo when HMD sees the poles.
Zero Eye Sep
Decreasing Eye Sep
Full Eye Separation
Decreasing Eye Sep
Zero Eye Sep

Fig. 3. Omnidirectional stereoscopic projection approach. (a) A conventional non-stereo omnidirectional projection. (b) Circular stereoscopic projection adapted from (a), as described by Peleg and Ben-Ezra [31] (c) Our omnidirectional stereoscopic projection (OSP) adapts (b) with modulation of the stereo eye separation distance, to prevent *backward-stereo* when either the zenith or nadir points of the polar axis are in view.

III. System Overview

The rendering system described herein is composed of several major software modules shown schematically for remote GPU cluster rendering in Fig. 2. The major software modules are shown along with the primary types of information they exchange and the direction of information flow. The progressive RT engine and the HMD display loop are both modules within the molecular visualization tool, and are directly coupled to its internal data structures for access to the molecular scene geometry, materials, and viewing parameters. The steps to render omnidirectional images and display view-dependent reprojections on an HMD are:

1) The molecular visualization application builds or updates the scene graph and sets progressive RT parameters
2) Progressive RT generates a new omnidirectional stereo sphere or cube map image pair locally, or by decoding an incoming H.264 video frame from a remote GPU cluster
3) The HMD update loop maps the RT output buffer, and uploads the stereo image pair as an OpenGL texture map
4) The HMD update loop queries the HMD on-board inertial sensors to determine the current head pose
5) The HMD update loop reprojects the scene for the current head pose by texture mapping onto a sphere or cube using the stereo image, rendering to an intermediate OpenGL framebuffer object (FBO)
6) The HMD update loop uses the FBO written in the previous step as a texture, drawing it to the OpenGL display framebuffer with warping and scaling operations to correct for HMD lens distortions, presenting the final image to the HMD,
7) User input is processed, and appropriate actions are taken, looping back to the beginning until completion

The process of displaying on conventional monitors is the same, except for the omission of the HMD-specific lens distortion correction steps. The system uses multithreading to maintain asynchronous execution of progressive RT and HMD display updates, with the HMD view update loop and user input handling residing in the main VMD thread.

IV. Omnidirectional Stereoscopic Projections

Omnidirectional stereoscopic projections (OSPs) and related panoramic projections capture all or much of the plenoptic function (all rays of light passing through a point in space) for a given eye location [32], [33]. OSPs produce images that can be reprojected for arbitrary viewing orientations and fields of view using standard rasterization techniques. A noteworthy benefit of OSPs is that the same OSP image can be used for any number of reprojections — i.e. independent users can look around from the same shared camera location. Of critical importance for the current application is the fact that OSP view-orientation-independence can be exploited to allow progressive refinement RT of the omnidirectional scene while one or more viewers look freely in any direction, until the camera is moved to another location.

A widely used application of omnidirectional projections for rasterization is environment texture mapping. Environment texture maps are typically stored either as a latitude-longitude sphere projection (e.g. equirectangular) or as a cubic projection (unfolded cube) [34]. Because of their established use we opted to implement variants of these two projections to render, store, transmit, and reproject omnidirectional scenes for HMDs. The popular latitude-longitude sphere and cubic projections both work well for photographic applications and for monoscopic rendering (see Fig. 3(a)), but as-is they do not support stereoscopy — rendering a pair of such projections with laterally displaced eye locations produces *backward-stereo* images when viewing to the rear, and *no stereo* effect when looking to the sides. Peleg and Ben-Ezra describe a so-called circular stereoscopic projection for photographic capture applications that provides a correct stereoscopic view over the full 360°circle. They describe image capture by sweeping a slit camera over circular path, laterally offset from the center of projection, with the camera view direction roughly tangent to the circle [31]. The circular stereoscopic projection is illustrated in Fig. 3(b). Simon et al. [18] and Bourke [19] describe the use of circular projections and image warping, for stereoscopic viewing. Simon et al. note that stereo views reconstructed from the circular projection approach are perfect for the viewing direction, but that they gradually distort off-axis from the view direction in the periphery of a user's field of view, emphasizing that the view is *"always correct where you are looking"*. When rendering for HMDs with a variety of off-axis optical aberrations, we concur that this is a very satisfactory compromise.

Since we wish to produce OSP images for each eye in the form of latitude-longitude or cubic projections that are well supported by rasterization hardware, we combine these approaches with the circular projection technique, and we resolve one remaining obstacle for effective use with HMDs. The circular projection is correct when the viewer constrains their viewing direction near the horizon, but not when the zenith or nadir points on the OSP polar axis are visible, since the viewer will see *backward-stereo* images in the hemisphere behind the pole (from the viewer's perspective). We further adapt the circular projection, as shown in Fig. 3(c), by modulating the stereoscopic eye separation such that it begins at normal eye separation near the horizon, and is smoothly decreased, reaching zero by the time either the zenith or nadir points on the polar axis are visible to the user, producing a monoscopic image. The modulation of eye separation can be computed as a function of the cross product of the viewing direction and the polar axis, and the display half-field-of-view angle. Finally, our OSP approach allows the system to handle the completely arbitrary HMD head poses necessary for independent multi-user viewing of the same OSP stereo image. In the single-user case the HMD pose could be used to guide automatic reorientation of the projection to prevent monoscopic viewing near the poles.

V. Interactive Progressive Ray Tracing

Ray tracing has long been used for high quality image synthesis due to its versatility, allowing incorporation of a wide variety of sophisticated optics, lighting, and shading techniques [13], [14]. The primary drawback for RT as compared with rasterization or other rendering approaches has always been the relatively high computational cost of traversing scene geometry by very large numbers of rays. The pursuit of interactive RT led to considerable research on techniques for increasing traversal efficiency and parallelism [20]–[22].

The transition of modern CPUs and GPUs toward highly parallel SIMD-oriented hardware architectures, combined with the availability of data-parallel programming languages such as CUDA [35], OpenCL [36], ISPC [37], and IVL [38] has created the foundation for state-of-the-art RT algorithms and implementations. The ascent of high level RT frameworks such as NVIDIA OptiX [23], and Intel OSPRay, as well as low level kernel libraries such as Intel Embree [39] and AMD FireRays, enables the development of fast, hardware-optimized RT engines while minimizing the need for hand-coded assembly language kernels. These RT toolkits raise the level of programming abstraction using SPMD-oriented (single program, multiple data) dialects of C and C++, and eliminate barriers for the implementation of high performance RT engines on commodity platforms.

A. *Ray tracing for molecular visualization*

Ray tracing, ray casting, and ray marching methods have played a significant role in molecular visualization going back at least two decades. Their appeal is in large part due to their facility for high quality direct rendering of spheres and cylinders [4], [8], [10], [24], [40] and other curved surfaces [41], and for radial basis functions and volumetric data [29].

Contemporary use of RT in high quality offline batch rendering of molecular structures and movies of their dynamics takes advantage of features such as ambient occlusion lighting, high quality transparent surface rendering, shadows and shadow filtering by transmissive surfaces, depth of field, and reflections [1]–[4], [42]. Ambient occlusion (AO) lighting greatly assists comprehension of shape [11], [12]. The use of AO lighting is particularly beneficial for elucidating important features of biomolecular complexes, such as pores, pockets, and cavities which can be difficult to identify with direct lighting, even when used with fog or depth cueing. Direct lighting and hard shadows, while computationally inexpensive, often act to obscure features of molecular structure without careful light placement. One of the major benefits of AO lighting from the perspective of molecular scientists is that it is largely automatic, typically requiring adjustment of only one or two scaling parameters.

Ray tracing produces the highest quality results for AO lighting, and it would generally be the rendering method of choice when it is sufficiently fast. Many AO lighting approximations have been developed for use with rasterization, but each with its own limitations. Screen-space ambient occlusion (SSAO) approximations estimate AO lighting based on the contents of the rasterization depth buffer [43]. SSAO approaches are subject to inconsistent behavior during close-up views since they do not account for off-screen geometry, and for stereoscopic viewing where AO is inconsistent between eyes. Other AO techniques are often limited in terms of lighting fidelity, particle counts, or the geometric primitives they support, thus limiting their utility for large molecular systems and scene content [44], but this remains an area of active research [25]. Progressive RT yields a fully general AO solution that produces high quality images.

B. *GPU-accelerated ray tracing engine*

The interactive RT engines currently implemented in VMD are based on C/C++ (Tachyon [9], [10] on any platform), on OSPRay/ISPC (TachyonL-OSPRay on Intel x86 and Xeon Phi CPUs), and on OptiX/CUDA (TachyonL-OptiX [1], [4] on NVIDIA GPUs). The present work extends the TachyonL-OptiX GPU-accelerated RT engine which has the requisite combination of video streaming support, maturity, and performance [1], [4], however we note that the same techniques would be equally applicable to other interactive RT engines when combined with comparable video streaming technology and performant hardware. The TachyonL-OptiX RT engine in VMD is a variant of the Tachyon RT engine based on the OptiX GPU-accelerated RT framework [23] and the CUDA GPU programming toolkit [35]. The OptiX RT framework provides the basic machinery to manage load balancing and scheduling of ray intersection testing and shading, and to create and traverse RT acceleration structures [45]–[47]. A key component of OptiX used by our system is an RT-specific just-in-time (JIT) compiler that combines groupings of seven types of user-supplied OptiX RT *programs* (CUDA kernels and device functions for individual RT steps) into one or more so-called "megakernels" formed by merging user-supplied kernels with OptiX-supplied kernels for acceleration structure traversal and work scheduling, thereby yielding a complete RT pipeline [48]. The VMD RT engines all have direct access to the molecular scene graph data structures that also drive the conventional rasterization-based display modes of the program, thereby minimizing memory footprint and I/O.

The VMD-specific RT kernels are the most performance-critical components of the RT engine and they implement all of the geometry- and shading-material-specific operations required for molecular visualization. Some of the custom RT kernels and GPU device functions required by the system include: primary ray generation, geometry bounds tests, ray-geometry intersection tests, material shaders, AO lighting, directional lighting, depth-of-field sampling, accumulation buffer handling, and random number generation. The RT engine uses CUDA C++ templates to create 256 compile-time-specialized *closest-hit* material shading kernels that are optimized for two combinations of five different global rendering parameters and three different material properties. At runtime, the RT engine selects optimized *closest-hit* material shading kernels from the table of compile-time-specialized kernel variants, and combines them with *ray generation* kernels for primary rays, *any-hit* shadowing kernels, and *miss* kernels, using the OptiX runtime JIT compilation mechanism [4].

C. Progressive ray tracing

The goal for all progressive RT techniques is to provide the viewer with an interactively updating image that improves as the RT progresses, accumulating and filtering stochastic samples to produce intermediate images for display [49]–[51]. Progressive RT runs in a tight loop, continuously updating the displayed image by launching and accumulating one or multiple images, so-called "subframes", in each iteration. Each subframe generates new stochastic samples that are incorporated with other subframes and previous samples to produce an updated image for display.

A key area where our progressive RT engine has been customized for immersive molecular visualization is in its allocation of stochastic samples between image refinement and AO lighting convergence. Considerable experience working with users of the interactive RT features of VMD and their reported preferences has guided us to implement rendering heuristics known to benefit molecular scientists. One observation is that users prefer crisp renderings with quality equal to or greater than that of OpenGL from the first displayed subframe. Most progressive RT implementations favor uncorrelated sampling of AO and other broad area lighting. Uncorrelated AO sampling produces high quality converged images, but creates strong pixelated speckle noise until the image approaches convergence with a large number of accumulated samples. For the purposes of molecular visualization we are generally uninterested in achieving absolutely perfect lighting convergence, and when viewing a moving or dynamic scene, it is undesirable to see significant speckle noise. To that end, our progressive renderer uses a single random number stream to drive AO sampling for all pixels in the same subframe. In this way, all pixels take AO lighting samples in the same direction, thereby eliminating speckle noise, yielding images for use in presentations or publications, even with undersampled lighting. An undesirable effect that can arise from the use of a single random number stream for stochastic AO samples is minor "flutter" in average image brightness during the first few samples. The flutter effect can be eliminated by increasing the AO sample count per hit point and by accumulating multiple concurrently rendered subframes together, as described below. Both approaches are used in our new progressive RT engine.

Our progressive RT engine uses a moving average to track its update rate and dynamically adjust the number of samples computed by successive subframes with the goal of maintaining a target update rate, e.g. 30 Hz. Users can override the automatic subframe sample adjustment and force a specific setting, e.g. to optimize for quality and image convergence rate rather than frame rate. With the two-phase OSP rendering technique, the progressive RT update rate is completely decoupled from the rate of HMD head pose updates and rendering, so it only needs to be fast enough to provide acceptably smooth world translations or head location updates.

Once the number of subframe samples has been determined, the progressive RT engine chooses a sample allocation strategy that favors image-space refinement, AO lighting refinement, or a hybrid, depending on scene content. When only direct lighting and hard shadows are used, the system prioritizes image-space refinement. When AO lighting is enabled, lighting refinement is prioritized. When both AO lighting and depth-of-field focal blur are in use, image-space refinement is prioritized to accelerate convergence of bokeh (blurring effect) from geometry beyond the zone of critical focus. The combination of AO lighting with depth of field is challenging, requiring hundreds of samples for fully converged images with no remaining speckle in bokeh, though perfect convergence is not typically required.

On multi-GPU systems, subframes can be decomposed spatially and parallelized across multiple GPUs to achieve higher performance, but in practice, parallel efficiency is limited by uneven work distribution in different regions of the rendered scene. Larger numbers of GPUs can be used more effectively by computing many independent subframes in parallel, yielding a greatly increased number of progressive samples per second while maintaining a more uniform work distribution among GPUs [4]. By using a hybrid approach that decomposes both among subframes and spatially within subframes, tens of GPUs can be effectively utilized for interactive progressive rendering of complex scenes. The high-quality

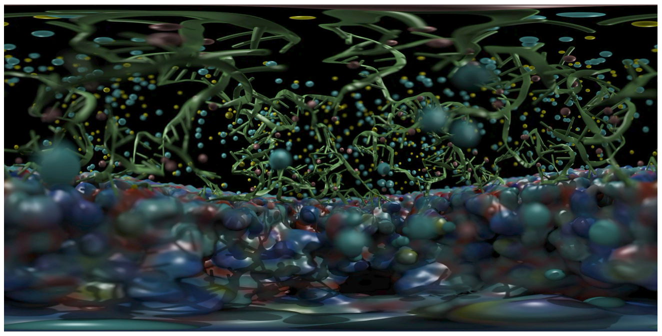
Fig. 4. Left eye latitude-longitude image from an omnidirectional stereoscopic projection of the Satellite Tobacco Mosaic Virus (STMV) scene containing over 626,000 objects, shown with ambient occlusion lighting, depth of field focal blur, two directional lights, and shadows.

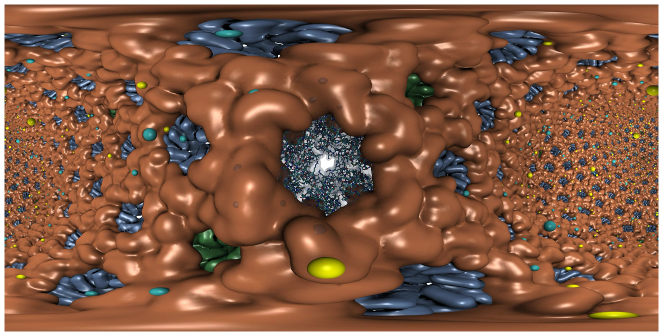
Fig. 5. Left eye latitude-longitude image from an omnidirectional stereoscopic projection of the HIV-1 capsid scene containing 9.7-million objects, shown with ambient occlusion lighting using a maximum occlusion distance cutoff, a point-light "headlight" at the camera location, and shadows.

OSP progressive RT loop and the low-latency HMD update loop are decoupled in independent threads, thereby allowing the HMD to free run while polling for new OSP images.

D. Omnistereo and panoramic ray tracing challenges

Omnidirectional and panoramic stereo projections of biomolecular scenes pose a number of unusual challenges. The standard planar formulations of depth-of-field focal blur, fog and depth cueing, as well as view frustum clipping all assume planar projections. For correct rendering in OSPs, all of these rendering features and effects had to be reimplemented with radial or spherical formulations.

Biomolecular scenes are densely populated with geometry. The crowded molecular environment can be a dark place due to heavy shadowing with both AO and direct lighting. The addition of a point light "headlight" at the center of projection can prevent excessive darkening when viewing interiors of fully-enclosed organelles, virus capsids, pores, and other heavily shadowed viewpoints. A drawback of the headlight approach is that it produces "flat" looking shading since it is very close to the camera position. A more broadly useful approach is to modify the AO shadow test algorithm to ignore shadows from occluding geometry beyond a maximum radius from the hit point. By selecting a maximum occlusion distance roughly half the interior radius of a fully enclosed molecular complex, AO lighting can be effectively used, retaining key benefits for shape perception, and maintaining ease-of-use.

Structures too close to the camera create uncomfortable stereoscopic views that can make navigation difficult. Close proximity to molecular structures in all directions can make this a potentially significant impediment to viewing comfort. To address this problem, an optional clipping sphere feature was added to the RT engine. Objects inside the sphere (near the viewer's eyes) are smoothly faded toward complete transparency and are ultimately clipped out entirely. To prevent clipped geometry from casting shadows, the clipping sphere also affects secondary rays.

VI. Remote Ray Tracing and Video Streaming

The use of remote visualization from clusters and supercomputers can eliminate the need for large file transfers and it creates an opportunity to exploit large scale parallel computing techniques within interactive visualization and analysis scenarios. As future all-atom structures and simulations continue to grow in size and timescale, the data size and computing requirements for effective visualization will make routine use of remote rendering critically important.

Beginning with OptiX 3.8, a remote device API enables the use of interactive RT-optimized hardware — the NVIDIA Visual Computing Appliance (VCA). A VCA installation consists of a GPU cluster interconnected by a fast InfiniBand network fabric. Each VCA node is composed of a fast host system and an array of eight high-performance Quadro M6000 GPUs. The OptiX progressive RT API allows work to be redirected from local GPUs to remote VCA cluster nodes by transparently transmitting application-provided RT executable code, scene geometry, material properties, and associated viewing parameters to the VCA nodes.

Our progressive RT engine supports remote rendering and we have used it to render very large molecular complexes with complex geometry, shading, and lighting scenarios. For the purposes of interactive OSP rendering, which requires very high image resolutions, we find that the best frame rate is achieved using a single 8-GPU VCA node, due to overheads associated with CPU-based tone mapping and InfiniBand network communication for parallel image compositing. Using a single VCA node, we have achieved remote RT frame rates of over 30 FPS for 2160×1440 OSP images. A critical aspect of the design of our remote rendering system that makes it usable with latency-sensitive HMDs is the asynchronous decoupling of the HMD display loop from remote RT and all associated round-trip network latencies. We typically observe long-haul round-trip network latencies that average 65 ms, which is far too high for the use of HMDs. We note that even at the speed of light, the round-trip latency for a 2,000 mile network link is 21 ms.

The image stream produced during progressive rendering on the remote VCA cluster is sent over the long-haul network using H.264 video encoding, and is decoded on the molecular visualization client system prior to presentation on a local display or HMD. The H.264 video stream uses conventional

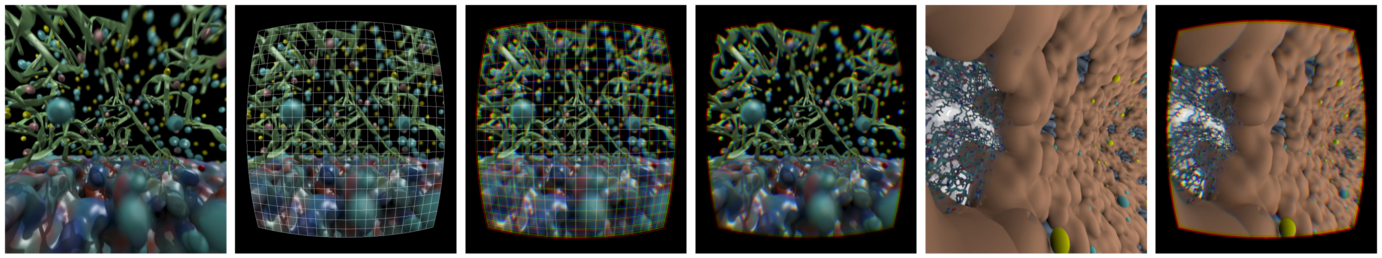

Fig. 6. Left eye HMD images for Satellite Tobacco Mosaic Virus capsid and RNA (four leftmost), and the HIV-1 capsid (two rightmost), with roughly 100° FoV. The images show the HMD reprojection of the OSP images in Figs. 4 and 5, with HMD lens distortion correction steps. The STMV images show the left eye OSP reprojection with: (from left) no lens distortion corrections; geometric lens distortion correction only (mesh shown); geometric and chromatic aberration corrections (mesh shown); final left eye HMD image. The HIV images show (from left): no lens distortion correction; final left eye HMD image.

4:2:0 chrominance downsampling (quarter resolution chrominance encoding), which anecdotally doesn't seem to harm image quality during use. Since the OSP rendering produced by our system has very high resolution, typically 3072×1536 or greater, the available network bandwidth, target H.264 bit rate and target H.264 frame rate are all important factors for streaming progressive RT updates to the client system. Through experimentation we have found several H.264 video encoding parameters that yield good results when used with a VCA cluster located 2,100 miles from the molecular visualization client system. We set the H.264 stream for a bit rate of 15Mbps and a target frame rate of 24 FPS.

VII. Omnidirectional Viewing and HMDs

The omnidirectional stereo projections (OSPs) produced by our system are ideally suited for display using HMDs connected to a conventional laptop or workstation, or potentially by mobile phone-based displays, but they can also be shown in stereoscopic full-dome theaters, CAVE-like displays, or on conventional 3-D TVs and monitors. Below we focus on the use of HMDs as they have a particularly challenging requirement for smooth, low-latency, high-frame-rate display updates — not just for improved immersion, but most importantly, to prevent users from experiencing so-called simulator sickness.

A. Omnidirectional stereo image display

Our RT engine can produce a variety of omni-stereo projections including latitude-longitude (or *equirectangular* — mapping from a rectangular image to spherical latitude-longitude angles), a cube map (spherical projection mapped onto the six faces of a cube), and dome-master (hemispherical projection mapped to a circular region inscribed within a square image). Each OSP image contains a stereo pair in an over-under or side-by-side arrangement. The ray-traced OSP image is then reprojected and rasterized (e.g. by OpenGL) for the display by texturing it onto a spherical or cubic projection surface, with the camera at the origin inside, using a field of view that matches the HMD or other display hardware. For stereoscopic display, the left and right eyes images are reprojected and rasterized independently.

It is sometimes desirable to provide the user with graphical information in the form of heads-up display (HUD) overlays or other interfaces, or to draw special avatar geometry that represents the user's wand, hand, or arm pose along with sighting lines for various picking or manipulation operations. User interfaces and 3-D augmentation geometry can be drawn and composited on top of the OSP rendering prior to display using conventional texturing and depth buffering operations. The presentation of the OSP renderings on a particular display is finalized by applying any display-specific warping or distortion correction operations, which is particularly important for commodity HMDs. Figure 6 shows HMD eye images in various stages of applying distortion corrections.

B. Oculus Rift DK2 HMD display implementation

The rendering system described here was developed using the Oculus Rift DK2 HMD (hereafter DK2), which is intended for early prototyping and development of VR end-user applications (see Fig. 1). The DK2 incorporates a single HD resolution (1920×1080) low-persistence Super-AMOLED display, on-board Inertial Measurement Unit (IMU) sensors, and three sets of ocular lens assemblies that accommodate variations in user eyesight. The DK2 presents users with a 960×1080 image for each eye, covering an approximately 100°diagonal field of view, and it has a 75 Hz display refresh rate (13.3 ms refresh interval). The extreme corners of each eye's image area are not visible, and due to gradually increasing off-axis blur from the optics in the ocular assemblies, the effective resolution of each eye appears roughly halved for the outermost 40% radius of the field of view. Although Oculus VR provides an SDK for the DK2 HMD, the Linux platform heavily used by molecular scientists is not currently supported. We developed HMD display and distortion correction software optimized both for the stock DK2 HMD and for DK2 HMDs with enhanced ocular optics [52], shown in Fig. 1 on right.

The singlet ocular lenses used in the DK2 and similar HMDs produce strong pincushion geometric distortion and lateral chromatic aberration that can be effectively corrected with modest additions to the rendering pipeline, and at low cost to performance. Pincushion distortion can be neutralized by applying the inverse barrel distortion prior to display. Chromatic aberration is caused by a lens having a wavelength-dependent focal length, which in the case of an HMD ocular assembly in turn creates a wavelength-dependent image scale. Correction of the strongest lateral chromatic aberrations in simple lenses can be achieved with per-color-channel image scaling. When rendering OSPs, correction of HMD lens distortions cannot be applied during RT, and must instead be performed during final image reprojection and display. Because distortion correction is performed during display, RT of OSP images remains completely decoupled from the HMD or other display device, and the same OSP rendering can be viewed by multiple viewers concurrently or collaboratively.

C. HMD display update loop

The 75 Hz display refresh rate of the DK2 is the natural performance target for our system. To achieve the smoothest immersive visualization experience, it is desirable to achieve HMD display update rates that exceed the display refresh rate, ensuring that any transient CPU or GPU activity does not adversely affect HMD viewing. To this end, we use a variety of techniques to implement an HMD display loop with the aim of HMD update rates that consistently exceed 100 Hz (10 ms redraw interval) on commodity PC hardware.

The key benefit of using OSP renderings as the basis for HMD viewing is that they allow decoupling the progressive RT update loop from the HMD display update loop. Decoupling of the two rendering phases is the key means to accommodate computationally demanding rendering techniques that would otherwise be inaccessible to users of HMDs, and likewise for tolerating round-trip latencies associated with remote visualization over long-haul networks. The RT engine and HMD display update loops run asynchronously in independent threads. In the case of local rendering, the HMD update loop exchanges image data directly with the RT engine. In the case of remote rendering, the remotely executing RT engine feeds a video encoding thread, and locally a video decoding thread feeds the HMD update and display thread. In both local and remote rendering cases, the HMD update loop polls for new OSP renderings. If a new OSP rendering is available, the HMD update loop maps the new image into host memory and binds it to an OpenGL texture map. Regardless of whether a new OSP rendering has been received, the HMD update loop proceeds with view-dependent HMD display. The HMD update loop proceeds by 1) computing the current HMD head pose quaternion; 2) reprojecting and rasterizing the stereoscopic eye views from the OSP and the current head pose; 3) adding any overlays; 4) performing final HMD-specific distortion corrections; and, finally, 5) committing the image to the HMD display.

When vertical retrace synchronization is enabled, the update loop blocks on a final OpenGL buffer swap, which helps prevent the HMD display update loop from monopolizing the host CPUs, the display GPU, and the host PCIe bus. We implemented two optional optimizations that can be used to achieve higher performance and in some cases smoother HMD display when using a host machine with comparatively slow CPUs or when rendering locally. The optimizations eliminate redundant OpenGL texture uploads and break up the series of steps required for display of new images from progressive RT, adding in intermediate HMD pose updates and display calls, thereby reducing the worst case HMD update interval.

VIII. Performance Results and Discussion

To evaluate the performance of the rendering system, tests were conducted on representative molecular scenes. Tests were run on a host workstation consisting of a 4-core Intel Core i7-4771 CPU @ 3.5 GHz with 16 GB of RAM and an NVIDIA Quadro M6000 GPU driving an Oculus Rift DK2 HMD. Remote RT tests were performed using an NVIDIA VCA GPU cluster, composed of nodes that each contain eight Quadro M6000 GPUs. The cluster used for the performance tests was located in Santa Clara, CA, over 2,100 miles away from the test workstation in Urbana, IL. Network ICMP "ping" tests indicated that the round-trip network latency between the workstation and the VCA cluster averaged 65 ms. We note that latencies on this order are far too high for direct-to-HMD rendering approaches without inducing simulator sickness. For all remote RT tests, the target H.264 video bit rate was set to 15 Mbps, and the target frame rate was set to 30 FPS.

An atomic-detail structure of the Satellite Tobacco Mosaic Virus (STMV) was used to evaluate RT performance for moderate-to-large biomolecular complexes, as shown in Fig. 4. The STMV structure is comprised of 955,226 atoms, including capsid proteins, interior RNA, water, and ions. The scene shows molecular surfaces for the virus capsid, so-called "cartoon" ribbon representations for the RNA, and van der Waals spheres for ions, and positions the camera inside the capsid near the interior RNA. The STMV scene contained 2,648 spheres, 963 cylinders, 622,476 triangles, and two lights.

An atomic-detail structure of the HIV-1 capsid was used to evaluate progressive RT performance for scenes with high geometric complexity. The HIV-1 structure is comprised of 2.6-million atoms, visualized using a combination of several graphical representations, as shown in Fig 5. The scene used two of the special RT features implemented in our system specifically for omnidirectional rendering: a "headlight" point light centered at the viewer's head location, and a user-defined maximum occlusion distance applied to the ambient occlusion lighting effect. The HIV-1 scene shows molecular surfaces for the hexameric and pentameric subunits in the virus capsid, transparent molecular surfaces and licorice representations for a highlighted hexamer, and van der Waals spheres for the ions in solution in the interior of the capsid. The HIV-1 scene positions the camera on the inside of the virus capsid near the highlighted hexamer. The scene contained 12,695 spheres, 22,118 cylinders, and 9.7-million triangles.

Table I summarizes performance for the STMV and HIV-1 scenes over a wide range of RT parameters. Tests were run with combinations of direct lighting and ambient occlusion (AO) lighting, and in combinations including depth-of-field (DoF) focal blurring. Spherical OSPs were used, with OSP 3072×1536 image dimensions. Since remote RT runtime scales roughly linearly with the total number of pixels, we omit lower resolution tests and results easily predicted from the timings listed in Table I. We generally observe overall higher RT sample throughputs (*image pixel count* × *samples per pixel* × *subframes per update* × *update rate*) for higher resolution

TABLE I
VMD remote OSP RT performance for a VCA GPU cluster located 2,100 miles from the VMD workstation and HMD. Oculus Rift DK2 HMD head pose and display updates were maintained at a constant 75 Hz, matching its display refresh rate. All tests computed two subframes in parallel per update, thus accumulating two times the sample counts shown in the subframe samples column for each display update. Ray tracing workload increases as (*pixel count* × *light count* × *sample count* × *subframe count*).

Scene and Rendering Features	Ray Traced Omnistereo Image Size	Per-subframe Samples AA : AO (AO Per Hit)	RT Update Rate (FPS)
STMV Shadows	3072×1536	1 : 0	22.2
		2 : 0	18.1
		4 : 0	10.3
STMV Shadows, AO	3072×1536	1 : 1	18.2
		1 : 2	16.1
		1 : 4	12.4
STMV Shadows, AO, DoF	3072×1536	1 : 1	16.1
		2 : 1	11.1
		2 : 2	8.5
HIV-1 Shadows	3072×1536	1 : 0	20.1
		2 : 0	18.1
		4 : 0	10.2
HIV-1 Shadows, AO	3072×1536	1 : 1	17.4
		1 : 2	12.2
		1 : 4	8.1

OSP images. We expect that this is due to slightly better utilization of GPU hardware due to the increased parallelism (and thus improved GPU processing element load balance) within individual progressive RT subframes. Similarly, we observe generally increased overall RT sample throughputs when computing larger numbers of samples in each subframe, likely the result of reduced GPU kernel launch overheads, and improved work scheduling. At higher OSP image resolutions, internal VCA subframe compositing work can begin to limit RT update rates, and it becomes more difficult to maintain smooth H.264 video streaming at moderate bit rates.

We find that RT update rates above 10 Hz allow the HMD viewer to comfortably fly the camera within the molecular scene, since HMD updates are still maintained at a constant 75 Hz. We note that conventional molecular visualization approaches based on planar projections and OpenGL rasterization often have difficulty maintaining high interactivity when viewing large complexes such as the HIV-1 capsid scene. Recently, Tan et al. have reported that HEVC/H.265 video encoding offers up to a 59% reduction in bit rate as compared with MPEG-4 AVC/H.264 for perceptually comparable video quality [53]. HEVC/H.265 encoding is a significant future opportunity for increasing streaming performance at 4K (3840×2160) image resolutions and beyond. Dynamic video rescaling, improved sampling, and sparse subframe sampling are further directions for future work.

IX. Conclusion

The rendering approach described here allows the use of high quality RT techniques in a highly interactive immersive molecular visualization system. Our approach overcomes unavoidable latencies associated with remote rendering so that latency-sensitive HMDs can be used for remote visualization of very large and geometrically complex molecular scenes. We have presented RT algorithms that have proven valuable for rendering OSPs, and we have reported system performance for challenging molecular visualization scenarios with our implementation in VMD [5]. We have outlined opportunities for future work that could positively impact our approach. An exciting opportunity for future development lies in multi-user collaborative visualization, and support for mobile phone-based HMDs as self-contained clients [54]. We feel our approach is general and that it could be used by many other domains within science and engineering.

Acknowledgments

The authors acknowledge support from NIH grants 9P41GM104601 and 5R01GM098243-02, the CUDA Center of Excellence at the University of Illinois, and the Blue Waters sustained-petascale computing project supported by NSF awards OCI-0725070 and ACI-1238993, the state of Illinois, and “The Computational Microscope” NSF PRAC awards OCI-0832673 and ACI-1440026.

References

[1] J. E. Stone, K. L. Vandivort, and K. Schulten, “GPU-accelerated molecular visualization on petascale supercomputing platforms,” in *Proceedings of the 8th International Workshop on Ultrascale Visualization*, ser. UltraVis ’13. New York, NY, USA: ACM, 2013, pp. 6:1–6:8.

[2] M. Sener, J. E. Stone, A. Barragan, A. Singharoy, I. Teo, K. L. Vandivort, B. Isralewitz, B. Liu, B. C. Goh, J. C. Phillips, L. F. Kourkoutis, C. N. Hunter, and K. Schulten, “Visualization of energy conversion processes in a light harvesting organelle at atomic detail,” in *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, ser. SC ’14. IEEE Press, 2014, (4 pages).

[3] J. R. Perilla, B. C. Goh, J. Stone, and K. Schulten, “Chemical visualization of human pathogens: the Retroviral Capsids,” *Proceedings of the 2015 ACM/IEEE Conference on Supercomputing*, 2015, (4 pages).

[4] J. E. Stone, M. Sener, K. L. Vandivort, A. Barragan, A. Singharoy, I. Teo, J. V. Ribeiro, B. Isralewitz, B. Liu, B. C. Goh, J. C. Phillips, C. MacGregor-Chatwin, M. Johnson, L. F. Kourkoutis, C. N. Hunter, and K. Schulten, “Atomic detail visualization of photoynthetic membranes with GPU-accelerated ray tracing,” *Parallel Computing*, 2016, in Press.

[5] W. Humphrey, A. Dalke, and K. Schulten, “VMD – Visual Molecular Dynamics,” *J. Mol. Graphics*, vol. 14, pp. 33–38, 1996.

[6] J. E. Stone, J. Gullingsrud, P. Grayson, and K. Schulten, “A system for interactive molecular dynamics simulation,” in *2001 ACM Symposium on Interactive 3D Graphics*, J. F. Hughes and C. H. Séquin, Eds. New York: ACM SIGGRAPH, 2001, pp. 191–194.

[7] J. E. Stone, A. Kohlmeyer, K. L. Vandivort, and K. Schulten, “Immersive molecular visualization and interactive modeling with commodity hardware,” *Lect. Notes in Comp. Sci.*, vol. 6454, pp. 382–393, 2010.

[8] J. E. Stone, K. L. Vandivort, and K. Schulten, “Immersive out-of-core visualization of large-size and long-timescale molecular dynamics trajectories,” *Lect. Notes in Comp. Sci.*, vol. 6939, pp. 1–12, 2011.

[9] J. E. Stone, B. Isralewitz, and K. Schulten, “Early experiences scaling VMD molecular visualization and analysis jobs on Blue Waters,” in *Extreme Scaling Workshop (XSW), 2013*, Aug. 2013, pp. 43–50.

[10] J. E. Stone, “*An Efficient Library for Parallel Ray Tracing and Animation*,” Master’s thesis, Computer Science Department, University of Missouri-Rolla, April 1998.

[11] M. S. Langer and S. W. Zucker, “Shape-from-shading on a cloudy day,” *J. Optical Society of America A*, vol. 11, no. 2, pp. 467–478, Feb 1994.

[12] M. S. Langer and H. H. Bülthoff, “Perception of shape from shading on a cloudy day,” Max-Planck-Institut fur Biologische Kybernetik, Tech. Rep. 73, October 1999.

[13] R. L. Cook, T. Porter, and L. Carpenter, “Distributed ray tracing,” in *Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH ’84. New York, NY, USA: ACM, 1984, pp. 137–145.

[14] C. Kolb, D. Mitchell, and P. Hanrahan, "A realistic camera model for computer graphics," in *Proceedings of the 22Nd Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '95. New York, NY, USA: ACM, 1995, pp. 317–324.

[15] N. L. Max, "ATOMLLL: ATOMS with shading and highlights," *SIGGRAPH Comput. Graph.*, vol. 13, no. 2, pp. 165–173, Aug. 1979.

[16] N. Max, "Computer graphics distortion for IMAX and OMNIMAX projection," in *Nicograph '83 Proceedings*, Dec 1983, pp. 137–159.

[17] N. Greene and P. S. Heckbert, "Creating raster omnimax images from multiple perspective views using the elliptical weighted average filter," *IEEE Comput. Graph. Appl.*, vol. 6, no. 6, pp. 21–27, Jun. 1986.

[18] A. Simon, R. Smith, and R. Pawlicki, "Omnistereo for panoramic virtual environment display systems," in *Virtual Reality, 2004. Proceedings. IEEE*, March 2004, pp. 67–279.

[19] P. Bourke, "Synthetic stereoscopic panoramic images," in *Interactive Technologies and Sociotechnical Systems*, ser. Lecture Notes in Computer Science, H. Zha, Z. Pan, H. Thwaites, A. Addison, and M. Forte, Eds. Springer Berlin Heidelberg, 2006, vol. 4270, pp. 147–155.

[20] S. Parker, W. Martin, P.-P. J. Sloan, P. Shirley, B. Smits, and C. Hansen, "Interactive ray tracing," in *Proceedings of the 1999 Symposium on Interactive 3D Graphics*, New York, NY, USA, 1999, pp. 119–126.

[21] I. Wald and P. Slusallek, "State of the Art in Interactive Ray Tracing," in *Eurographics 2001 - STARs*. Eurographics Association, 2001.

[22] I. Wald, H. Friedrich, A. Knoll, and C. Hansen, "Interactive isosurface ray tracing of time-varying tetrahedral volumes," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 6, pp. 1727–1734, 11 2007.

[23] S. G. Parker, J. Bigler, A. Dietrich, H. Friedrich, J. Hoberock, D. Luebke, D. McAllister, M. McGuire, K. Morley, A. Robison, and M. Stich, "OptiX: a general purpose ray tracing engine," in *ACM SIGGRAPH 2010 papers*, ser. SIGGRAPH '10. New York, NY, USA: ACM, 2010, pp. 66:1–66:13.

[24] M. Krone, K. Bidmon, and T. Ertl, "Interactive visualization of molecular surface dynamics," *IEEE Transactions on Visualization and Computer Graphics*, vol. 15, pp. 1391–1398, 2009.

[25] B. Kozlikova, M. Krone, N. Lindow, M. Falk, M. Baaden, D. Baum, I. Viola, J. Parulek, and H.-C. Hege, "Visualization of Biomolecular Structures: State of the Art," in *Eurographics Conference on Visualization (EuroVis) - STARs*, R. Borgo, F. Ganovelli, and I. Viola, Eds. The Eurographics Association, 2015.

[26] L. Marsalek, A. Dehof, I. Georgiev, H.-P. Lenhof, P. Slusallek, and A. Hildebrandt, "Real-time ray tracing of complex molecular scenes," in *Information Visualisation (IV), 2010 14th International Conference*, July 2010, pp. 239–245.

[27] K. Reda, A. Knoll, K.-I. Nomura, M. Papka, A. Johnson, and J. Leigh, "Visualizing large-scale atomistic simulations in ultra-resolution immersive environments," in *Large-Scale Data Analysis and Visualization (LDAV), 2013 IEEE Symposium on*, Oct 2013, pp. 59–65.

[28] A. Knoll, I. Wald, P. A. Navrátil, M. E. Papka, and K. P. Gaither, "Ray tracing and volume rendering large molecular data on multi-core and many-core architectures," in *Proceedings of the 8th International Workshop on Ultrascale Visualization*, ser. UltraVis '13. New York, NY, USA: ACM, 2013, pp. 5:1–5:8.

[29] A. Knoll, I. Wald, P. Navratil, A. Bowen, K. Reda, M. E. Papka, and K. Gaither, "RBF volume ray casting on multicore and manycore CPUs," *Comput. Graph. Forum*, vol. 33, no. 3, pp. 71–80, Jun. 2014.

[30] D. Reiners, C. Cruz-Neira, and C. Neumann, "Photorealistic 3D omnidirectional stereo simulator," in *Proc. SPIE 9392*, ser. The Engineering Reality of Virtual Reality, Mar. 2015.

[31] S. Peleg and M. Ben-Ezra, "Stereo panorama with a single camera," in *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, vol. 1, 1999, p. 401 Vol. 1.

[32] E. H. Adelson and J. R. Bergen, "The plenoptic function and the elements of early vision," in *Computational Models of Visual Processing*. MIT Press, 1991, pp. 3–20.

[33] L. McMillan and G. Bishop, "Plenoptic modeling: An image-based rendering system," in *Proceedings of the 22Nd Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '95. New York, NY, USA: ACM, 1995, pp. 39–46.

[34] N. Greene, "Environment mapping and other applications of world projections," *Computer Graphics and Applications, IEEE*, vol. 6, no. 11, pp. 21–29, Nov 1986.

[35] J. Nickolls, I. Buck, M. Garland, and K. Skadron, "Scalable parallel programming with CUDA," *ACM Queue*, vol. 6, no. 2, pp. 40–53, 2008.

[36] J. E. Stone, D. Gohara, and G. Shi, "OpenCL: A parallel programming standard for heterogeneous computing systems," *Comput. in Sci. and Eng.*, vol. 12, pp. 66–73, 2010.

[37] M. Pharr and W. Mark, "ispc: A SPMD compiler for high-performance CPU programming," in *Innovative Parallel Computing (InPar), 2012*, May 2012, pp. 1–13.

[38] R. Leissa, S. Hack, and I. Wald, "Extending a C-like language for portable SIMD programming," in *Proceedings of the 17th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, ser. PPoPP '12. New York, NY, USA: ACM, 2012, pp. 65–74.

[39] I. Wald, S. Woop, C. Benthin, G. S. Johnson, and M. Ernst, "Embree: A kernel framework for efficient CPU ray tracing," *ACM Trans. Graph.*, vol. 33, no. 4, pp. 143:1–143:8, Jul. 2014.

[40] C. Sigg, T. Weyrich, M. Botsch, and M. Gross, "GPU-based ray-casting of quadratic surfaces," in *Proceedings of the 3rd Eurographics / IEEE VGTC Conference on Point-Based Graphics*, ser. SPBG'06. Eurographics Association, 2006, pp. 59–65.

[41] M. Chavent, B. Levy, and B. Maigret, "MetaMol: High-quality visualization of molecular skin surface," *J. Mol. Graph. Model.*, vol. 27, no. 2, pp. 209 – 216, 2008.

[42] G. T. Johnson, L. Autin, D. S. Goodsell, M. F. Sanner, and A. J. Olson, "ePMV embeds molecular modeling into professional animation software environments," *Structure*, vol. 19, no. 3, pp. 293 – 303, 2011.

[43] L. Bavoil and M. Sainz, "Multi-layer dual-resolution screen-space ambient occlusion," in *SIGGRAPH 2009: Talks*, ser. SIGGRAPH '09. New York, NY, USA: ACM, 2009, pp. 45:1–45:1.

[44] M. Tarini, P. Cignoni, and C. Montani, "Ambient occlusion and edge cueing for enhancing real time molecular visualization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 5, pp. 1237–1244, 2006.

[45] M. Stich, H. Friedrich, and A. Dietrich, "Spatial splits in bounding volume hierarchies," in *Proceedings of the Conference on High Performance Graphics 2009*, ser. HPG '09. New York, NY, USA: ACM, 2009, pp. 7–13.

[46] K. Garanzha, J. Pantaleoni, and D. McAllister, "Simpler and faster HLBVH with work queues," in *Proceedings of the ACM SIGGRAPH Symposium on High Performance Graphics*, ser. HPG '11. New York, NY, USA: ACM, 2011, pp. 59–64.

[47] T. Karras and T. Aila, "Fast parallel construction of high-quality bounding volume hierarchies," in *Proceedings of the 5th High-Performance Graphics Conference*, ser. HPG '13. New York, NY, USA: ACM, 2013, pp. 89–99.

[48] A. Robison, "Domain specific compilation in the NVIDIA OptiX ray tracing engine," in *Proceedings of the Sixth Workshop on Declarative Aspects of Multicore Programming*, ser. DAMP '11. New York, NY, USA: ACM, 2011, pp. 1–2.

[49] L. Bergman, H. Fuchs, E. Grant, and S. Spach, "Image rendering by adaptive refinement," *SIGGRAPH Comput. Graph.*, vol. 20, no. 4, pp. 29–37, Aug. 1986.

[50] D. P. Mitchell, "Generating antialiased images at low sampling densities," in *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '87. New York, NY, USA: ACM, 1987, pp. 65–72.

[51] J. Painter and K. Sloan, "Antialiased ray tracing by adaptive progressive refinement," in *Proceedings of the 16th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '89. New York, NY, USA: ACM, 1989, pp. 281–288.

[52] L. Sun, B. Guenter, N. Joshi, P. Therien, , and J. Hays, "Lens factory: Automatic lens generation using off-the-shelf components," *ArXiv e-prints*, vol. arXiv:1506.08956, 2015.

[53] T. Tan, R. Weerakkody, M. Mrak, N. Ramzan, V. Baroncini, J. Ohm, and G. Sullivan, "Video quality evaluation methodology and verification testing of HEVC compression performance," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 26, no. 1, pp. 76–90, Jan 2016.

[54] J. E. Stone, M. J. Hallock, J. C. Phillips, J. R. Peterson, Z. Luthey-Schulten, and K. Schulten, "Evaluation of emerging energy-efficient heterogeneous computing platforms for biomolecular and cellular simulation workloads," in *Parallel and Distributed Processing Symposium Workshop (IPDPSW), 2016 IEEE International*. IEEE, May 2016.