CHAPTER 14

High-Performance Computing Applications for Visualization of Large Microscopy Images

Rajvikram Singh, Abel W. Lin , Jurgen P. Schulze, Steve T. Peltier, Maryann E. Martone and Mark H. Ellisman, University of California, San Diego

Large data visualization problems are prevalent in microscopy and find some reprieve in high-performance computing (HPC). Clusters and multi-CPU architectures help in accelerating applications such as feature extraction, image processing, and analysis of large 2D and 3D datasets. Cluster driven tile-displays have recently become popular end points for large data exploration because of their high-resolution capability and scalability. Certain algorithms and strategies have played a key role in designing parallel applications for these high-resolution displays. Issues regarding performance tuning of graphics, processing, and networking subsystems have also become important factors in building efficient scientific visualization pipelines for microscopy data.

14.1 Mesoscale Problem: The Motivation

Researchers analyzing brain tissue samples at multiple resolutions are faced with a well-known problem when traversing scales spread across several orders of magnitudes: the higher the resolution, the smaller the scope. When an investigator is zoomed in on a sample at subcellular resolution he or she has lost context of where the region of interest (ROI) lies with reference to other ROIs. This gap between dimensional scales makes it difficult to understand how higher order structures are constructed from finer building blocks. Problems traversing scales are particularly acute in the dimensional range that is now called *mesoscale*, which is the dimensional range spanning hundreds of microns to nanometers. Structural elements within this range include subcellular structures, cell-cell interactions, and macromolecular constituents. Within the nervous system, the mesoscale encompasses those structures most associated with information processing (i.e., synaptic complexes, subcellular microdomains, and the fine structures of axons and dendrites). Understanding mesoscopic structures within the brain presents a unique challenge because of the extended nature of nerve cells, the cellular and molecular complexity of nervous tissue, and the intricate arrangement of cellular processes. Although the nervous system is perhaps the most extreme in terms of mesoscopic complexity, we are limited in our ability to understand even a well-ordered tissue such as muscle across large expanses in fine detail.

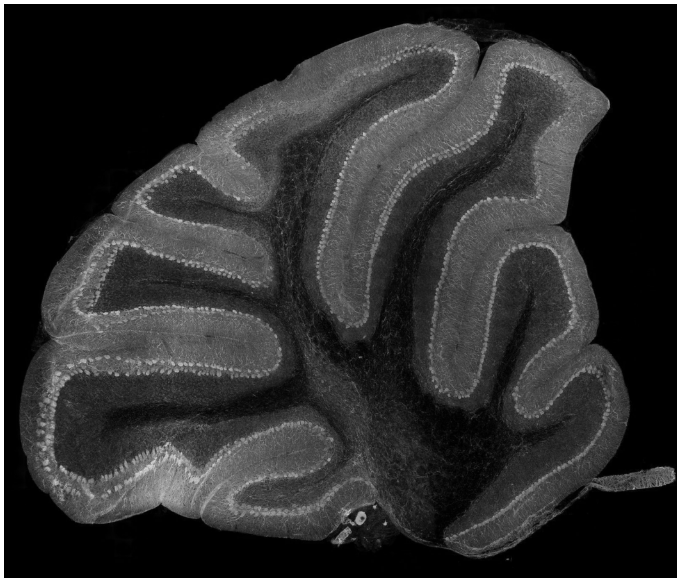
The mesoscale gap arises in part from the requirement to use multiple imaging technologies to examine a specimen across scales. Each technology requires different expertise, specimen preparation techniques, and contrast mechanisms, and also requires a severe reduction in the amount of tissue. For example, if the pipeline begins with an entire brain, the end results in one small block of tissue, $< 0.5\ \mathrm{mm}^3$. These requirements make it difficult for individual researchers to bridge scales, both because single researchers may not be familiar with a given technology and because there is significant loss of context as the scope decreases with increasing resolution of imaging technologies. Bridging techniques such as multiphoton microscopy and electron tomography, correlated microscopy is a key methodology for acquiring the necessary multiscale data in order to fill in the resolution gaps between gross structural imaging and protein structure: data which is central to bridging the mesoscale gap and to the elucidation of the nervous system.

From a computer science perspective, the mesoscale presents a number of challenges. The two major ones are large data sizes and ultra-high resolution content. The typical size of a dataset collected by a microscope, capable of acquiring ultra-wide field mosaics, ranges from a couple of gigabytes to a few terabytes on disk. The content resolution ranges from a few hundred megapixels to hundreds of gigavoxels. The largest CCD sensors for electron microscopes are now approaching 64 megapixels. With image formats of 8,000 $\times$ 8,000 pixels, these systems can be used to acquire wide-field mosaics (2D, 3D, and 4D) that exceed hundreds of gigabytes of content [1].

For most analysis applications, researchers want to juxtapose many such datasets next to each other for visual comparison and analysis. Due to the large memory footprints of these datasets, it is not possible to load them entirely in the

Q1

Figure 14.1 A section of the mouse heart as seen under National Center for Microscopy and Imaging Research's 8K $\times$ 8K CCD camera. The camera has a collective resolution of 64 megapixels and the specimen stage can be moved in X and Y to collect tiles. These tiles can then be stitched together form ultrawide field of view mosaic.



Q2

Figure 14.2 The figure shows a scaled down version of an 18,000 $\times$ 16,000 pixel montage created by data acquired from a light microscope. This ''small'' dataset has a resolution of 288 megapixels and is $\sim$70 times the resolution of the best LCD monitor available in 2008.

video memory or even in the RAM of typical workstations. To add to the complexity, since many research groups are collaboratively working on these datasets, they are usually stored offsite, typically at a data center. In order to move a 100-gigabyte dataset over a university fast Ethernet network it typically takes over 2 hours. Moving large data between the stages of the visualization pipeline not only requires efficient storage and query engines but networks that are several magnitudes faster by current standards.

14.2 High-Performance Computing for Visualization

The sheer size of the datasets generated by the field of microscopy has challenged computer scientists to come up with unique approaches for each phase of the visualization pipeline. Figure 14.3 shows a typical scientific visualization pipeline. The computational and rendering components of the pipeline are usually clusters or supercomputers hosted by universities and research organizations. Often these clusters are geographically located several hundreds of miles apart and at times in different countries since microscopy is a very collaboration-rich field. However, the final result of the visualization has to be delivered back to the researchers who initiated the process. To compound the problem, visualization applications are expected to be interactive in nature. This means that the popular model of queued job-submission in high-performance computing does not apply anymore since the users are expecting an instantaneous feedback. The causality of a user's mouse interaction perpetuates data to be pulled from data centers and delivered to the cluster-farms for computation and rendering. The result of the rendering,

Figure 14.3 A typical scientific visualization pipeline.

which is usually graphical primitives or pixels, is then delivered to the display at the user's end. All this needs to happen within a period of few hundred milliseconds to seconds in order for the visualization to be interactive. The following section describes how computational grids and high-speed optical networks are used for solving such data and compute intensive tasks.

14.2.1 Data Acquisition

Filling in the mesoscale information gap requires data from a number of different imaging instruments. Researchers can now collect correlated, multiscale data from electron and laser scanning light microscopes to create interconnected 2D, 3D, and 4D geometries to study structure function relationships within key cellular and tissue subsystems.

Over the past decade, this data generation has accelerated at an exponential rate, and scientific imaging instruments (e.g., electron and light microscopes) have been automated to now deliver large datasets, some exceeding 1 terabyte. From a resource perspective, such immense data sizes require seamless access to computational, data management, and visualization resources that scale beyond what can be effectively deployed by individual research groups [11, 21].

14.2.2 Computation

Visualization in microscopy involves some very computationally intensive processes such as volume rendering and tomographic reconstructions of large datasets. These tasks are time consuming because of the sheer number of calculations involved. For example, tomographic reconstruction of an 8K $\times$ 8K dataset with approximately 60 angles can take up to 4 days on a multi-CPU state-of-the-art computer. Researchers have been drawn towards high-performance computing in hope of finding some reprieve in clusters and computation grids [12, 13]. A significant amount of effort has been directed towards getting biological codes to run on parallel architectures in order to cut down the turnaround time. For shared computational resources, users have to use the job-submission model where a task is assigned to CPUs by a scheduler. The assignment is governed by existing load on the machines and the number of jobs submitted. Typically it is not possible to estimate a worst case turnaround time for a job and users might have to wait from anywhere between, say, a few hours to a few days. Ideally, resources such as CPU cycles, memory, and network bandwidth should be dedicated for a job so an upper bound can be estimated on the time needed. In the best case users would like to get a visual feedback of a lower resolution model of the data so that they have the option of adjusting the parameters in real-time instead of waiting for the task to finish computing the large data.

Besides faster computation it is also important to streamline this process for ease of use since the end users are typically not aware with the intricacies of grid computing. Web interfaces and the concept of a workflow are often employed to enrich the user experience and to guide them through the visualization pipeline [14–19]. This also ensures that the users are abstracted from the idiosyncrasies of the underlying computational architecture while providing the flexibility of future expansion of the resources.

14.2.3 Data Storage and Management

Modern microscopes are capable of generating several gigabytes of data on a daily basis. This translates to terabytes of raw data every year that needs to be made accessible to different collaborators for analysis and then archived for long-term storage. Researchers typically store the raw data generated by instruments on fast local disk arrays until it has been pruned and processed for analysis. It is at this point that the data is uploaded to data centers where it can be collaboratively viewed and analyzed by several groups. Specialized data centers have cropped up tending to the large data storage and serving needs of research groups across the globe. These centers not only provide scalable arrays of fast spinning disks but are accessible over fast optical networks. Efficient query of relevant data is a key component in the visualization pipeline and requires dedicated resources such as those provided by these data centers. It is not possible to keep all data on spinning media, and older, less-relevant data are usually relegated to slower, high-density media such as tapes where it is still accessible by the users.

As microscopes increase in resolution and data acquisition capabilities, data growth is not expected to slow down in the near future. A robust and scalable storage resource strategy is required for the archival and management of these large datasets [11, 21]. Researchers are constantly investigating new algorithms, architectures, and media to help them in managing the growing storage needs of the community [20].

14.2.4 Moving Large Data with Optical Networks

In its January 2001 edition, *Scientific American* published an article about the growing demand for network bandwidth, reporting that the growth rate for network bandwidth far exceeded the growth rate of processors as predicted by Moore's law [2]. In just the past few years it has become possible for research organizations and universities to buy dedicated lambdas or optical light paths. With the necessary networking hardware, it is possible to connect rare resources like expensive instruments, supercomputers, and clusters over these high-speed networks. This trend was one of the main reasons for inception of the OptIPuter project [3]. The OptIPuter was a National Science Foundation project that was aimed at developing an advanced distributed computing infrastructure for collaborative data exploration in the fields of biological and earth sciences. The project helped set up several light paths between universities across countries including the United States, Canada, the United Kingdom, The Netherlands, and Japan, to name a few. The project also aimed at experimenting with the intriguing idea of user controlled light paths (UCLPs) [4, 5] where users for the first time get to schedule dedicated network resources and fiber between sites for experiments. This potentially allows a group of researchers to custom tailor a scientific visualization pipeline with dedicated bandwidth of several gigabits per second for moving large data. This approach would allow exploration of large data collaboratively by research groups separated by large geographical distances. Once the experiment is over, the resources would be freed and reconfigured for a different purpose. A significant amount of research has been directed towards developing new network protocols and performance tuning existing Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) stacks inside operating systems to utilize these new networks more efficiently.

14.2.5 Challenges of Visualizing Large Data Interactively

As mentioned earlier, it is more challenging to model systems and software for interactively exploring large datasets since the queued job-submission model prevalent in high performance computing does not provide the desired quality of service (QoS) guarantees. The queued best-effort approach suffers from a serious drawback where results of a task might be returned after several minutes to a few hours. Since the users are expecting instantaneous feedback every time they interact with the software, the pipeline cannot possibly process the entire dataset at interactive rates. Parsing several gigabytes of data is intensive work even for the fastest parallel computers, and resource allocation has to be done in a dedicated manner to guarantee a certain QoS. One commonly employed scientific visualization scheme for handling large data involves mip-mapping [6]. A large dataset is sent through a preprocessing stage where multiple resolutions of the data are created and stored in a tree-like structure. To aid in fast query or recovery of this data, it is also indexed or chopped into smaller pieces. During the actual rendering phase, the users interact with a low-resolution version of the dataset. The software will automatically start filling in the high-resolution detail once the user has chosen a region of interest. Also depending on the zoom level and the region explored, only the data that is actually needed by the graphical hardware is read from storage and rendered on the screen. Thus effectively at any given point, the software is only dealing with a small manageable portion of the large raw data.

Apart from the problem of handling large datasets, the rendered output can span several hundreds of megapixels. Tile-displays have proven to be an effective solution for solving the need for large pixel real estates [7]. Figure 14.4 shows one such display being used by researchers for visualizing high-resolution content. The collective resolution of these displays can run up to a few hundred megapixels. They are usually run by a cluster where a single node drives one or more tiles and all the nodes are interconnected by gigabit Ethernet or equivalent hardware. The

Figure 14.4 Researchers using NCMIR's 40-megapixel display wall for conducting collaborative data exploration experiments. The display is managed by Scalable Adaptive Graphics Environment (SAGE), which allows multiple graphical applications to use the tile-display like a shared desktop space.

software responsible for managing the displays is inherently distributed in nature and can synchronize the visualization across tiles. Ideally we would like to use the entire display as one giant desktop where multiple applications can simultaneously reside. Datasets can be placed adjacent to each other for visual comparison using these applications. These windowed applications can either run locally on one or more local nodes or remotely across the network on a cluster.

Tile-displays at times span entire room lengths and can be cumbersome to interact with because of the large physical space they cover. Because of their large size, the paradigm of a keyboard and a 2D mouse on a fixed workstation does not apply very well. Users sometimes want to analyze the high-resolution content up close and sometimes want to step back to do a visual comparison of multiple datasets. Multiple users want to interact with different sections of the display simultaneously and want their own mouse pointer and keyboard. This is different from the typical one-desktop-one-user paradigm used by desktop operating systems. Thus, the displays bring with them a host of unique human-computer interface problems that require unique solutions. Computer scientists have experimented with alternate input devices such as wireless 3D mice and wireless tablet PCs which can be held by the user as they walk around the room and help provide user input to the displays. Groups are also working on camera-based face, gaze, and hand-tracking systems which will one day help do away with the need for carrying any physical input device on the person.

14.3 Visualizing Large 2D Image Data

Large montages are generated by biologists on a daily basis and are created by stitching together contiguous overlapping tiles of images acquired by the microscopes in the X-Y plane. Final image resolutions vary from a few hundred megapixels to several gigapixels and storage on disk varies from a few hundred megabytes to terabytes. These datasets are perfect cases for using the mip-mapping approach [6] to achieve maximum interactivity. The software loads appropriate detail at different zoom levels by paging image data to video memory in a view-dependent manner. As noted earlier, the datasets cannot be used directly in their raw format and have to be sent through a preprocessing step in order to generate a hierarchical multiresolution structure. The tree is arranged in a way where lower or higher resolution images of a region can be accessed by traversing up or down the levels. Figure 14.5 depicts such a quad-tree where every node has four children and the child nodes are at higher resolution than their parents.

For the sake of simplicity, splitting is done with square tiles where the sides of the squares are power of 2. The edge cases can be buffered with null or black pixels to maintain power of 2 dimensions. The other advantage of using this approach is that OpenGL textures are managed as squares with power of 2 edges. Thus it is easy to do the mapping between files on disk and textures in graphics hardware memory and the load times are consistent. It also helps memory management by minimizing fragmentation. The other big advantage of splitting levels into smaller tiles is when different cluster nodes are trying to load their portion of the data from the shared network file system it drastically minimizes contention for access to the same portion of data. Smaller files are more easily cached in memory by the file system and can thus be transferred over the network more efficiently.

MagicCarpet [22] is software that supports mip-mapped based visualization of large 2D datasets (see Figure 14.6). It allows interactive viewing of multiple and possibly time-dependent datasets and even vector data. It has a simple intuitive user interface which allows users to zoom, pan, and flip through images. It has been observed that even though the mip-map creation needs to be done only once for every dataset, users find it inconvenient since the preprocessing can take several minutes to a few hours. This is a long time period especially in collaborative workspaces where users want to simply present the data to their collaborators during meetings. It is also unintuitive from the user interface perspective for the

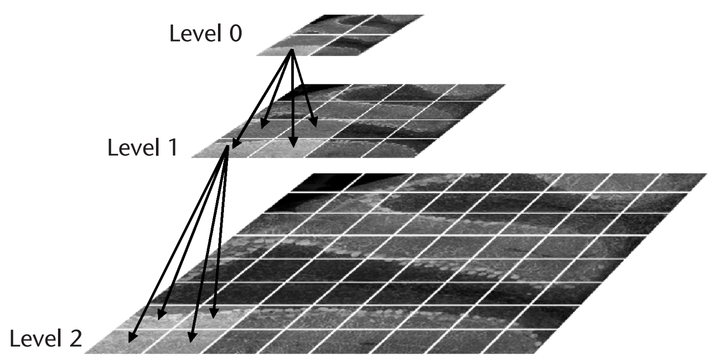


Figure 14.5 The figure shows a quad-tree hierarchy of images. Every parent node has four child tiles. All tiles are uniform squares with edge dimension which are power of two.

Figure 14.6 MagicCarpet running on the LambdaTable. The LambdaTable is a high-resolution, cluster-driven display table that supports multiple simultaneous users. (Image courtesy: Electronic Visualization Laboratory, University of Illinois at Chicago.)

data to go through this step before it can be seen on the displays. Users typically get impatient and sometimes assume that the software has become unresponsive on account of it taking a long time.

However, the task is embarrassingly parallel [23] and can be sped up by several factors by running the preprocessing code on multiple CPUs. The OptIPuter project has made it possible for create a grid of computing resources over high-speed optical networks and the preprocessing step can be offloaded to clusters and supercomputers residing on this fast backplane. This type of grid computing approach is widely popular in the scientific community since it allows groups to share resources beyond the means of most research organizations. We notice a significant speedup by running the preprocessing code on this distributed framework. The amount of speedup depends on the number of CPUs used and resources available but modest tests have shown that figures greater than 10X are easily achievable.

14.4 Visualizing Large 3D Volume Data

Large 3D volumes are a result of montages collected along the Z axis and are best described as a stack of 2D images. These volumes can easily occupy several terabytes of disk space and require techniques like *ray tracing* or *ray casting* [24] for rendering. Ray casting is a popular image-order method that generates realistic renderings by casting viewing rays from each point on the image plane through the data [24]. Samples are taken at discrete intervals along the ray. These samples are used to generate pixels on an image plane. Thus the 3D volume space is projected on a 2D plane. The density of the rays and sampling frequency used decide the resolution of the final image rendered. Some of these lighting models are very complicated, and achieving interactive frame rates even with volumes that can be fitted into the graphics memory can get challenging very fast. (See Figure 14.7.)

There are other similar volume rendering methods such as 3D texture mapping [25] and several possible optimizations for making computation more efficient and even hardware support is built inside most modern GPUs, but the sheer number of calculations involved increases exponentially with volume size. Most techniques also require that the data be contained entirely in RAM if implemented on the CPU, or in texture memory if implemented using graphics hardware [25]. Modified approaches are used when data size exceeds the amount of available memory. The three primary approaches for dealing with large volume data involve data bricking/paging, use of compression, and parallel processing. Data bricking is analogous to the mip-mapping discussed in the previous section where raw data is sent through a preprocessing stage to generate a multiresolution hierarchical data structure. In the case of 3D data, this structure is an octree where every node has eight children as shown in Figure 14.8. As in the case of 2D mip-mapping, individual levels are divided into smaller manageable bricks. Depending on the view frustum requested by the user, bricks are loaded in to the graphics memory or discarded. Modern graphics hardware already supports a notion of paging akin to virtual memory where least recently used memory blocks are purged and overwritten with data from the main memory to aid in fast graphics pipeline processing.

Octreemizer [26] is one such system that uses an octree data structure coupled with a multilevel paging system and predictive cache to roam through large volumetric data. The multilevel cache operates between video memory and main memory, and between main memory and disk. Using a least recently used (LRU) replacement strategy, the predictive cache fetches data based on the direction of user movement.

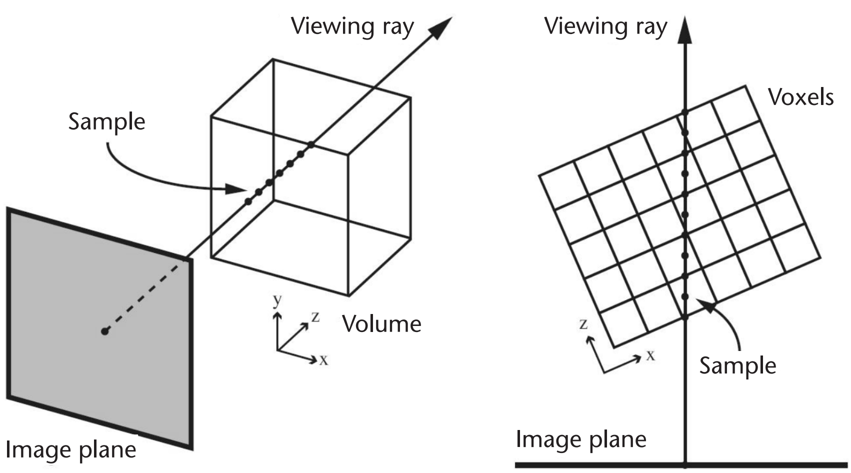


Figure 14.7 Volume rendering using ray casting. A ray starting at a point on the image plane is cast through the volume to evaluate the optical model. Samples are taken at evenly spaced intervals along the ray.

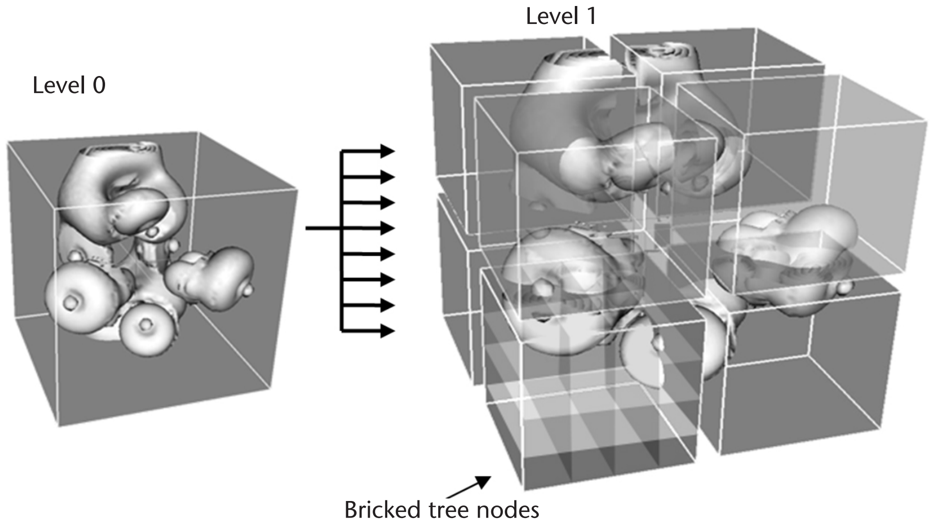


Figure 14.8 Volume data arranged as an octree where every level has eight children. Each node in the tree is further bricked into smaller pieces to aid in efficient loading times and memory management.

Parallel rendering of large volumes with multiple CPUs are primarily done using either an image-order approach or an object-order approach. Both approaches allow the task of rendering to be distributed to multiple CPUs simultaneously and then require an additional step to merge the individual results into a final image. Image-order rendering requires looking back from the final image and deciding on a portion of the image to be generated by each processor. Since this division is disjoint for every processor, they can each work on their sections of the data and render tiles. The compositing step is simple since all the tiles have to be stitched together for the final image. However, care needs to be taken to distribute the tasks efficiently since a processor with mostly black or empty voxels can finish its task earlier than others and will then sit idle while other processors are computing. Object-order rendering usually splits the data between processors in a fixed predetermined manner without worrying about the view frustum of every frame that is generated. During the execution, every CPU is supposed to calculate a projection based on the user-determined view port. However, the final compositing step is more complex in this case since it requires blending of all the individual projections in the right order.

Parallel computation of data is one requirement for handling large volumes interactively. The other requirement is display of the high resolution rendered output. Again cluster-driven tile-displays prove to be an invaluable asset in providing the necessary pixel real-estate to view the results across several hundreds of megapixels. The Volume Rendering Application (VRA) [27] is an image-order based parallel volume rendering system that can render on high-resolution display walls. It employs an octree-based multiresolution data structure to page in the correct bricks and resolution based on the view frustum.

14.5 Management of Scalable High-Resolution Displays

One basic requirement for working with these room-sized tile-displays is that they be treated as one contiguous desktop where multiple visualization applications and/or datasets can be rendered simultaneously. These applications can be a heterogeneous mixture of simple, single CPU programs to more complex parallel codes that run on clusters and supercomputers. Sometimes these applications execute on remote machines and/or clusters and need to stream their high-resolution output to the displays at the user end over fast networks. Examples of such compute-intensive applications are large 2D mosaic viewers and volume rendering software that are used by researchers for viewing and analyzing biological data.

14.5.1 SAGE (Scalable Adaptive Graphics Environment)

SAGE provides a graphics streaming architecture for supporting collaborative scientific visualization environments with potentially hundreds of megapixels of contiguous display resolution [8]. It was specifically designed for high-resolution cluster-driven scalable tile-displays and provides a giant contiguous "desktop" space. Applications that run on SAGE-enabled displays transport their rendered pixel frame buffers over the network [9, 10]. Multiple applications can be shown on these big desktops as windows that can be resized or moved by the users. Large datasets arranged next to each other for visual comparison are an important tool for the biologists since a lot of the analysis is still done manually. (See Figure 14.9.)

Figure 14.9 SAGE driven high-resolution displays allow users to arrange large datasets next to each other for visual comparison. Collaboration tools in the environment also support HD video conferencing.

The network-centric architecture of SAGE allows users to simultaneously run various compute intensive applications, such as 3D volume rendering, 2D montage viewers, and video streaming on local or remote computers. The environment also supports collaboration where the pixels from applications can be displayed at multiple sites simultaneously using network multicast or broadcast. Since the resolution of most graphical applications that run on these displays is very high, streaming requires a lot of bandwidth. For example, an uncompressed 30-fps HDTV stream with a resolution of $1{,}920 \times 1{,}080$ requires $\sim$1.5 Gbps. The OptIPuter infrastructure plays a crucial role in enabling this distributed computing architecture. Rendering and compute clusters can access the high-resolution displays over fast optical networks and stream their pixel frame buffers.

SAGE's streaming architecture is designed so that the output of arbitrary $M \times N$ pixel rendering cluster nodes can be streamed to $Q \times R$ pixel display screens [10], allowing user-definable layouts on the display. The dynamic pixel routing capability lets users freely move and resize each application's imagery over tiled displays in run-time, tightly synchronizing multiple component streams to form a single virtual stream.

14.5.2 COVISE (Collaborative Visualization and Simulation Environment)

COVISE [28] was originally been developed at the High Performance Computing Center Stuttgart (HLRS), and has been commercialized by the Stuttgart based VISENSO GmbH. It is a toolkit to integrate several stages of a scientific or technical application such as grid-generation, simulation, data import, post-processing, and visualization. Each step is implemented as a module. Using a visual user interface, these modules can be connected to a data flow network.

Each of the computational and I/O modules in this workflow can reside on a different computer. This allows distributing the work load among different machines. For instance, the pre- and post-processing modules can run on a visualization server, while the simulation runs on a remote supercomputer. The display modules can run on the workstation of a user, or on a visualization cluster driving a multiscreen visualization environment.

COVISE's virtual reality rendering module OpenCOVER can run on a variety of interactive displays and environments. Figure 14.10 shows COVISE managing the 15-tile rear-projected StarCAVE VR environment. It can even be used on a single computer with a mouse, but then the user cannot take advantage of its immersive capabilities. OpenCOVER is ideally run on tracked stereo environment, using 3D pointing devices. OpenCOVER uses the OpenSceneGraph API for its 3D rendering, which is an object-oriented framework on top of OpenGL. OpenCOVER has the ability to link multiple virtual environments together over the Internet, allowing for collaborative work between users in different buildings of a campus, or even on different continents. OpenCOVER is an open interface, in that the application programmer can write plug-in modules in C++ to create customized virtual reality applications, using COVISE's support of a large variety of virtual reality input and output devices, as well as interaction handling and network communication algorithms.

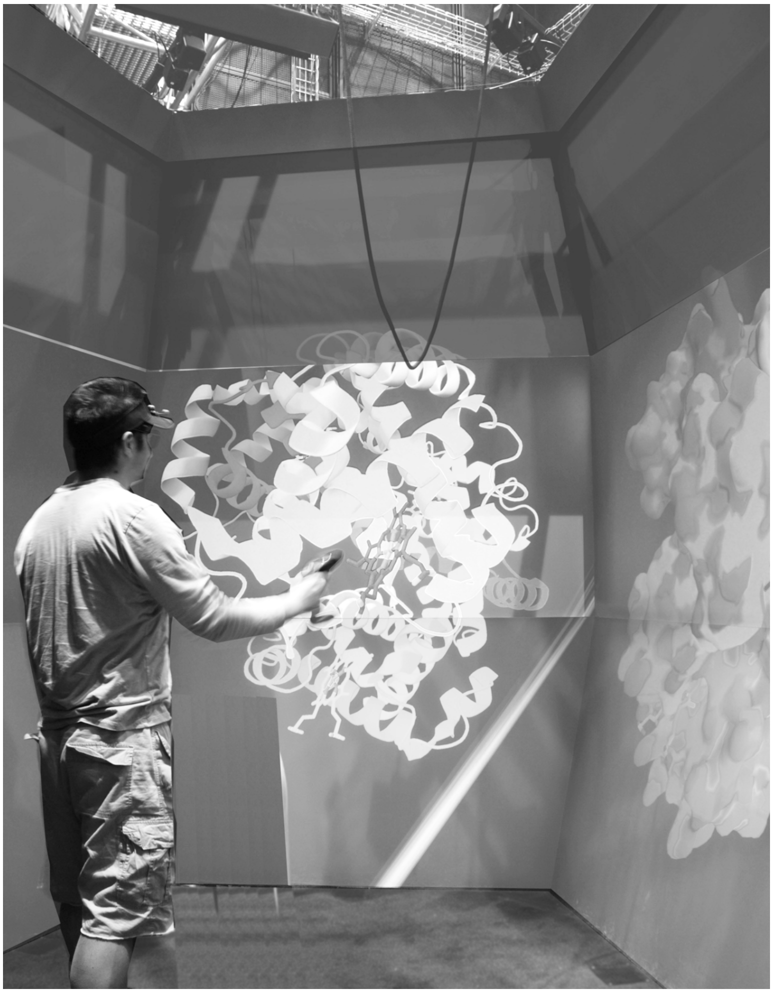


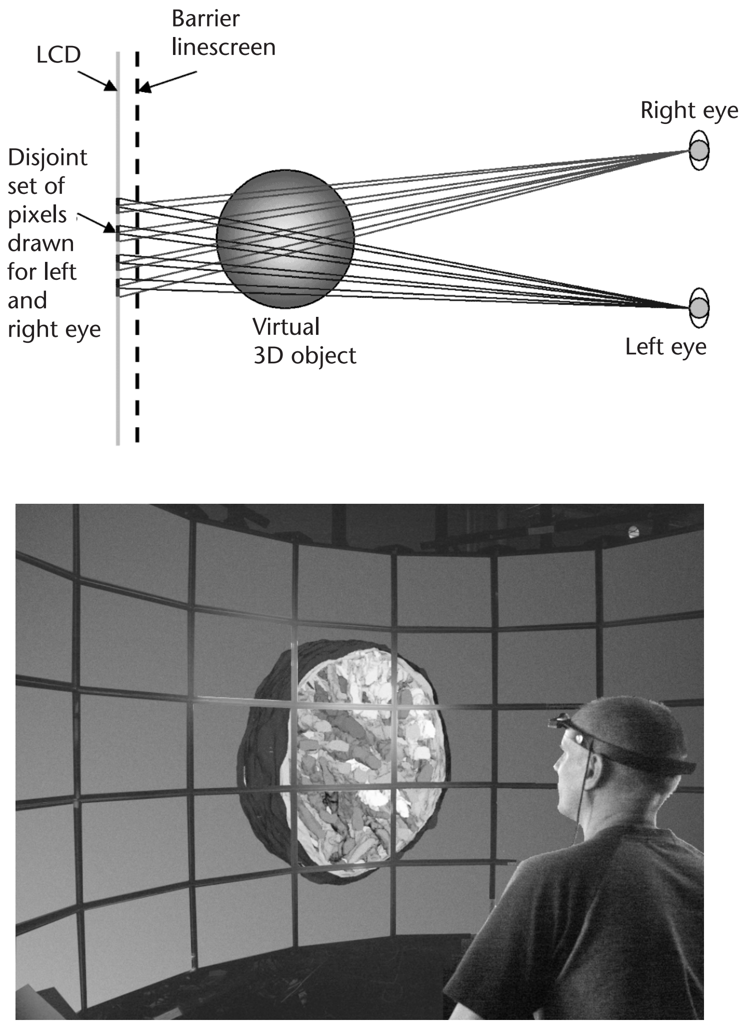
Figure 14.10 COVISE running on the 15-tile StarCAVE at the California Institute for Telecommunications and Information Technology at the University of California San Diego.

14.6 Virtual Reality Environments

Virtual reality display environments have historically proven invaluable for scientific visualization. Large 3D data exploration becomes more intuitive when the users are immersed in the dataset and forced to orient themselves with respect to regions of interest.

14.6.1 CAVE (Cave Automatic Virtual Environment)

The CAVE [29] is room-sized virtual reality apparatus that was invented at the Electronic Visualization Laboratory at the University of Illinois at Chicago. It is made up of high-resolution rear-projection displays and the first prototypes were cube-shaped rooms of dimensions $10 \times 10 \times 10$ ft. Projection is done on the walls and the floor to enable an immersive VR experience. Users wear 3D glasses to see stereo and the system supports multiple simultaneous users. The primary viewer wears a headgear and is head-tracked so that the correct perspective view is generated for his/her eyes. The user can navigate through the 3D scene using a handheld controller like a joystick. The joystick position and orientation in space is tracked too so the user can potentially reach into the data and click on a point of interest. The first versions were run by expensive multi-CPU machines specialized for graphics processing and later versions are run by Linux clusters using commodity graphics and processing hardware.



Q3

Figure 14.11 (a) illustrates how the barrier linescreen technology is used for generating pixel sets which are only visible to one eye at a time. Two such sets enable stereo vision. (b) shows a user interacting with the stereo display without using glasses.

Since its invention in 1991, many CAVE installations have appeared at universities and research organizations across the globe and proven to be invaluable for scientific visualization and data exploration. As seen in Figure 14.10, complex protein and biological structures can be understood better because of the high pixel count of the projected screens. It is even possible to achieve higher resolution by tiling projectors per wall.

14.6.2 Varrier

The Varrier [30, 31] is an auto-stereoscopic scalable virtual reality display that allows users to view stereo images without the need to wear any glasses. Unlike most stereo systems, which are based on projected displays that use active or passive stereo techniques, it uses the barrier stereography technique [30, 31] to generate images for the left and right eye. It can use LCD displays and so it is possible to build a high-resolution tile-display version of this auto-stereoscopic device. In the barrier method, a virtual barrier screen is created and placed in the virtual world in front of the projection plane. An off-axis perspective projection of this barrier screen, combined with the rest of the virtual world, is projected from at least two viewpoints corresponding to the eye positions of the head-tracked viewer. Figure 14.11 illustrates this principle.

The user does not have to deal with glasses but the system still uses a sensor based head-tracking system which lets the system know where the user's eyes are in 3D space. The projections for the eyes are then drawn behind the barrier linescreen accordingly. Like for the CAVE, the users also have a tracked joystick that allows users to pick and poke at 3D data. Work is well underway to develop a fast neural networks and camera-based head-tracking system that will help do away with the sensor headband. Future systems will also employ high-resolution cameras to do real-time hand gesture recognition to replace the joystick. The ultimate goal is to have a completely encumbrance free VR system.

14.7 Future of Large Data Visualization

The popular belief of using high-performance computing to group several CPUs together to achieve the power of future machines plays a key role in solving the large data-exploration problems of today. Multicore, multi-CPU parallel architectures seem to have an answer for handling large data visualization and we will see more of these machines appearing at research organizations across the globe. One recent exciting development has been the advent of programmable GPUs. These inexpensive processors currently host up to 128 cores and have dedicated fast memory and bus bandwidth. Driven by the video game industry, these graphics cards are available as commodity hardware and can be clubbed together for scalability. It is possible to envision a cluster farm of GPUs running scientific visualization codes for interactive data exploration. Due to the inexpensive hardware, they will also enable smaller research organizations to build their own graphics supercomputer without having to depend on expensive shared resources such as those hosted by supercomputing centers.

14.8 Conclusion

The problems generated by mesoscale in microscopy have challenged high-performance computing by bringing together large multimodal, multiscale data collected from a variety of instruments. However, the problems need a good solution in visualization to help in our understanding of the data. Scientific visualization has always proven to be a challenge for the best and fastest computing machines as scientific datasets, such as those generated by microscopy, get bigger every year. It is apparent that computers will be playing catch-up with the data for several years to come. Though the problem looks overwhelming at first glance, with the advent of fast and inexpensive optical networks and graphics and computing hardware, it is possible to tailor high-performance scientific visualization pipelines to help alleviate large data exploration problems to a great degree. Efficient parallel data handling algorithms and high-resolution scalable displays also play a key role in visualizing these datasets.

References

[1] Price, D.L., Chow, S.K., MacLean, N.A.B., Hakozaki, H., Peltier, S., Martone, M.E., and Ellisman, M.H., "High-Resolution Large-Scale Mosaic Imaging Using Multiphoton Microscopy to Characterize Transgenic Mouse Models of Human Neurological Disorders," *Neuroinformatics*, 4(1):65–80, 2006.

[2] http://www.sciamdigital.com/.

[3] http://www.optiputer.net/.

[4] He, E., Wang, X., Vishwanath, V., and Leigh, J., "AR-PIN/PDC: Flexible Advance Reservation of Intradomain and Interdomain Lightpaths," *IEEE GLOBECOM 2006*, June 31, 2006.

[5] Mambretti, J., "The Digital Communications Grid: Creating a New Architectural Foundation for 21st Century Digital Communications Based on Intelligent Lightpaths," in *Annual Review of Communications*, International Engineering Consortium, Vol. 58, 2005, pp. 307–314.

[6] Williams, L., "Pyramidal Parametrics," in *Proceedings of SIGGRAPH '83, Computer Graphics*, 17(3):1–11, July 1983.

[7] Park, K., Renambot, L., Leigh, J., and Johnson, A., "The Impact of Display-rich Environments for Enhancing Task Parallelism and Group Awareness in Advanced Collaboration Environments," Third Annual Workshop on Advanced Collaborative Environments, co-located at the Twelfth IEEE International Symposium on High Performance Distributed Computing (HPDC 12) and Global Grid Forum 8, Seattle, WA, June 22, 2003.

[8] http://www.evl.uic.edu/cavern/sage/.

[9] Jeong, B., Jagodic, R., Renambot, L., Singh, R., Johnson, A., and Leigh, J., "Scalable Graphics Architecture for High-Resolution Displays," *Proceedings of IEEE Information Visualization Workshop 2005*, Minneapolis, MN, October 23–25, 2005.

[10] Jeong, B., Renambot, L., Singh, R., Johnson, A., and Leigh, J., "High-Performance Scalable Graphics Architecture for High-Resolution Displays," Technical Paper, May 1, 2005. Q4

[11] Zhang, S., Price, D.L., Qian, X., Gupta, A., Wong, M., Ellisman, M.H., and Martone, M.E., "A Cell Centered Database (CCDB) for Multi-Scale Microscopy Data Management," *Microscopy and Microanalysis, Proceedings*, August 3–7, 2003, San Antonio, TX.

[12] Lin, A., Su, M., Kulungowski, A., Lathers, A., Mehta, G., Peltier, S., Deelman, E., and Ellisman, M., "Cyberinfrastructure for parallel tomographic reconstruction and

analysis," *4th Intl. Congress on Electron Microscopy*, Nov. 5–8, 2006, San Diego, CA.

[13] Peltier, S.T., Lin, A., Lee, D., Smock, A., Lamont, S., Molina, T., Wong, M., Dai, L., Martone, M.E., and Ellisman, M.H., "The telescience portal for tomography applications," *J. Parallel Distributed Computing*, 63:539–550, 2003.

[14] Lathers, A., Su, M., Kulungowski, A., Lin, A.W., Mehta, G., Peltier, S.T., Deelman, E., and Ellisman, M.H., "Enabling Parallel Scientific Applications with Workflow Tools," *Proceedings of the 6th International Workshop on Challenges of Large Applications in Distributed Environments*, pp. 55–60, 2006.

[15] Molina, T., Yang, G., Lin, A.W., Peltier, S. and Ellisman, M.H., "A Generalized Service-Oriented Architecture for Remote Control of Scientific Imaging Instruments," *Proceedings of the 1st IEEE International Conference on e-Science and Grid Computing*, pp. 56–63, 2005.

[16] Lin, A.W., Dai, L., Mock, J., Peltier, S. and Ellisman, M.H., "The Telescience Tools: Version 2.0, Proceedings of The 1st IEEE International Conference on e-Science and Grid Computing, pp. 56–63, 2005.

[17] Lin, A.W., Dai, L., Ung, K., Peltier, S. and Ellisman, M.H., "The Telescience Project: Applications to Middleware Interaction Components," *Proceedings of the 18th IEEE International Symposium on Computer-Based Medical Systems*, pp. 543–548, 2005.

[18] Lee, D., Lin, A.W., Hutton, T., Akiyama, T., Shinji, S., Lin, F.P., Peltier, S. and Ellisman, M.H., "Global Telescience Featuring IPv6 at iGrid2002," *Future Generation of Computer Systems*, 19(6):1031–1039, 2003.

[19] Peltier, S.T., Lin, A.W., Lee, D., Mock, S., Lamont, S., Molina, T., Wong, M., Martone, M.E., and Ellisman, M.H., "The Telescience Portal for Advanced Tomography Applications," *Journal of Parallel and Distributed Applications, Special Edition on Computational Grids*, 63(5):539–550, 2003.

[20] Ludäscher B., Gupta, A., and Martone, M.E., "A Model-Based Mediator System for Scientific Data Management," in T. Critchlow and Z. Lacroix, (Eds.), *Bioinformatics: Managing Scientific Data*, Morgan Kaufmann, 2003

[21] Martone, M.E., Zhang, S., Gupta, A., Qian, X., He, H., Price, D., Wong, M., Santini, S., and Ellisman, M.H., "The Cell-Centered Database: A database for multiscale structural and protein localization data from light and electron microscopy," *Neuroinformatics*, 1(3):379–396, 2003.

[22] http://www.evl.uic.edu/cavern/mc/index.html.

[23] http://en.wikipedia.org/wiki/Embarrassingly_parallel.

[24] http://en.wikipedia.org/wiki/Ray_tracing_%28graphics%29.

[25] http://portal.acm.org/citation.cfm?doid=378456.378484#.

[26] Plate, J., Tirtasana, M., Carmona, R., and Fröhlich, B., "A Hierarchical Approach for Interactive Roaming Through Very Large Volumes," *Joint Eurographics - IEEE TCVG Symposium on Visualization*, Barcelona, May 2002.

[27] http://www.evl.uic.edu/core.php?mod=4&type=3&indi=355.

[28] http://www.hlrs.de/organization/vis/covise/.

Q5 [29] Cruz-Neira, C., Sandin, D., DeFanti, T., Kenyon, R., and Hart, J., "The CAVE®: Audio Visual Experience Automatic Virtual Environment," *Communications of the ACM*, Vol. 35, No. 6, pp. 65–72.

[30] Sandin, D., Margolis, T., Ge, J., Girado, J., Peterka, T., and DeFanti, T., "The Varrier Autostereoscopic Virtual Reality Display," *ACM Transactions on Graphics, Proceedings of ACM SIGGRAPH 2005*, July 30 to August 4, 2005.

[31] Peterka, T., D. Sandin, D., Ge, J., Girado, J., Kooima, R., Leigh, J., Johnson, A., Thiebaux, M., and DeFanti, T., "Personal Varrier: Autostereoscopic Virtual Reality for Distributed Scientific Visualization," *Future Generation Computing Systems*, October 1–31, 2006.