OPEN ACCESS

An optimized web-based approach for collaborative stereoscopic medical visualization

Mathias Kaspar,[1,3] Nigel M Parsad,[1,2] Jonathan C Silverstein[1,2]

▸ Additional supplementary files are published online only. To view these files please visit the journal online (http://dx.doi.org/10.1136/amiajnl-2012-001057).

[1]Center for Clinical and Research Informatics (CCRI), NorthShore University HealthSystem, Evanston, Illinois, USA
[2]Department of Surgery and Computation Institute, University of Chicago, Chicago, Illinois, USA
[3]Department of Medical Informatics, University Medical Center, Georg-August-University Göttingen, Göttingen, Germany

Correspondence to
Jonathan C Silverstein, Center for Clinical and Research Informatics (CCRI), NorthShore University HealthSystem, Research Institute, 1001 University Place, Evanston, IL 60201, USA; jsilverstein@northshore.org

Received 23 April 2012
Accepted 8 September 2012
Published Online First 9 October 2012

Open Access
Scan to access more free content

To cite: Kaspar M, Parsad NM, Silverstein JC. *J Am Med Inform Assoc* 2013;**20**:535–543.

ABSTRACT

Objective Medical visualization tools have traditionally been constrained to tethered imaging workstations or proprietary client viewers, typically part of hospital radiology systems. To improve accessibility to real-time, remote, interactive, stereoscopic visualization and to enable collaboration among multiple viewing locations, we developed an open source approach requiring only a standard web browser with no added client-side software.

Materials and Methods Our collaborative, web-based, stereoscopic, visualization system, CoWebViz, has been used successfully for the past 2 years at the University of Chicago to teach immersive virtual anatomy classes. It is a server application that streams server-side visualization applications to client front-ends, comprised solely of a standard web browser with no added software.

Results We describe optimization considerations, usability, and performance results, which make CoWebViz practical for broad clinical use. We clarify technical advances including: enhanced threaded architecture, optimized visualization distribution algorithms, a wide range of supported stereoscopic presentation technologies, and the salient theoretical and empirical network parameters that affect our web-based visualization approach.

Discussion The implementations demonstrate usability and performance benefits of a simple web-based approach for complex clinical visualization scenarios. Using this approach overcomes technical challenges that require third-party web browser plug-ins, resulting in the most lightweight client.

Conclusions Compared to special software and hardware deployments, unmodified web browsers enhance remote user accessibility to interactive medical visualization. Whereas local hardware and software deployments may provide better interactivity than remote applications, our implementation demonstrates that a simplified, stable, client approach using standard web browsers is sufficient for high quality three-dimensional, stereoscopic, collaborative and interactive visualization.

OBJECTIVE

Recent developments in hardware and imaging algorithms have led to wider deployment of medical imaging workstations with advanced volume visualization tools.[1 2] To create real-time, interactive, high-fidelity three-dimensional (3D) reconstructions of high-resolution CT and MRI datasets, state-of-the-art graphics processing unit (GPU) workstations and/or clusters are often required. User accessibility to such visualization resources can be physically limited to the local workstation. In contrast, remote visualization, that is rendering volume reconstructions of patient datasets on GPU workstations/clusters that are not local to the user allows for broader usage by multiple participants on client-side computers. These computers can range from standard PCs to portable devices.

Manipulating interactive remote visualizations via web-based systems that are similar to modern hospital information systems would enhance the accessibility even further.[3] We assert that a system that uses only standard web browsers with no added software provides universal access to remote visualization with highest client mobility. In this way, volume visualization of clinical radiological studies becomes ubiquitous among personal device browsers, thereby maximizing the user experience.

We have already outlined the feasibility of such an approach for medical education using CoWebViz, our collaborative, web-based, stereoscopic, visualization system.[4] This paper will describe the optimization of quality and interactivity of CoWebViz under varying network conditions enabling highly accessible interactive 3D stereoscopic visualization. This improved implementation is used for anatomy education at the Pritzker School of Medicine at the University of Chicago.

Stereoscopic presentation technologies provide real depth perception cues and may become a significant tool for clinical scenarios in the future.[5 6] As illustrated in figure 1, our approach allows for a variety of collaborative interactive visualization use cases with participants who may control the stereoscopic 3D visualization from different locations; for example, shared among different hospitals or offices, for daily radiological conferences, medical consultations, surgical treatment planning, or education.

BACKGROUND AND SIGNIFICANCE

Shared visualization approaches

Seamless visualization of remotely rendered medical imaging data is sensitive to the performance of the network. Clinical workflow delays can even be attributed to issues with the network scheme. With client-side rendering, the medical data need to be transferred to each client for visualization (see figure 2A). With server-side rendering (figure 2B), the data stay on the server with the resulting visualization transferred to each client as a series of two-dimensional (2D) images. Whereas the first may take 7–27 min to transfer 0.5–2 GB on a 10 Mbps network, creating a problematic start-up delay, the second approach starts streaming immediately and needs only 8–80 ms for each modified image of 10–100 KB.

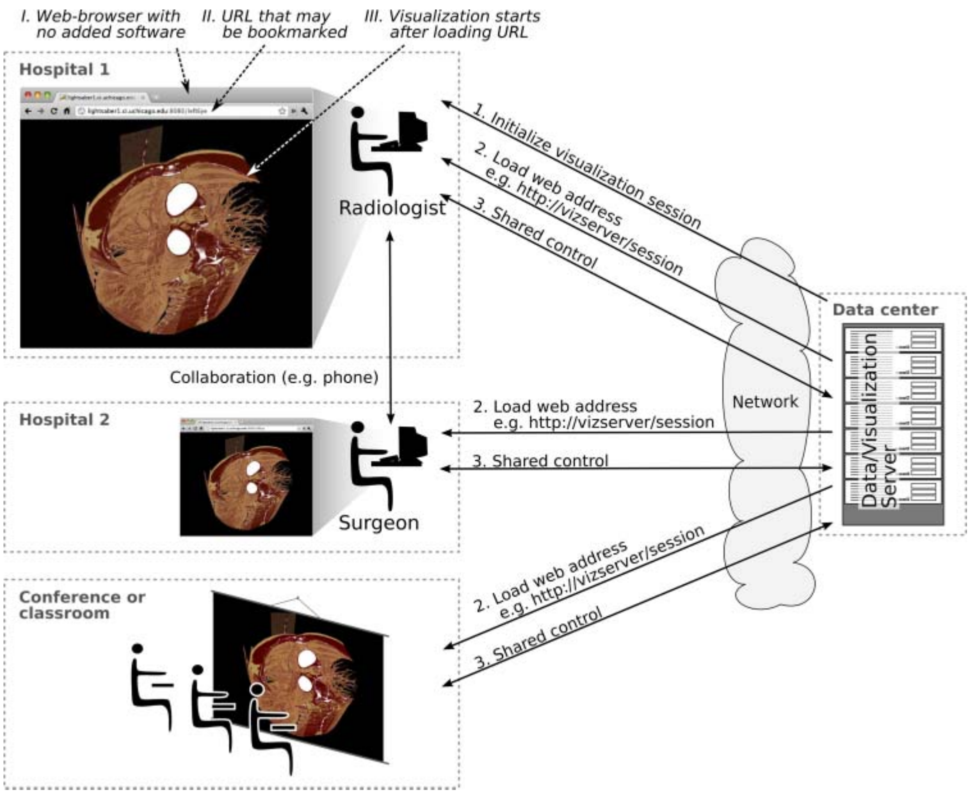
I. Web-browser with no added software
II. URL that may be bookmarked
III. Visualization starts after loading URL
Hospital 1
Radiologist
1. Initialize visualization session
2. Load web address e.g. http://vizserver/session
3. Shared control
Collaboration (e.g. phone)
Hospital 2
Surgeon
2. Load web address e.g. http://vizserver/session
3. Shared control
Network
Data center
Data/Visualization Server
Conference or classroom
2. Load web address e.g. http://vizserver/session
3. Shared control

Figure 1 Scenarios for a collaborative three-dimensional (3D) and 3D stereoscopic visualization system. One person (eg, a radiologist) initializes the visualization and other participants (eg, a surgeon or a whole class) can join for a shared visualization session.

Rendering on the client side delivers good usability and minimal delays resulting in fast interactivity. However, performance is also dependent on the client machine's graphics hardware. Web-based picture archiving and communication systems have been developed that support the client-side viewing of volume data as cross-sectional 2D images with and without browser plug-ins.[7 8] Other systems provide interactive non-stereoscopic 3D visualization that is rendered with web browser plug-ins.[9–11] Web browser plug-in engines such as Java and/or Flash are an additional installation requirement for all of the aforementioned web-based client-side systems. In contrast, Web3D technologies utilize the client-side machine's graphics processing hardware via OpenGL commands within a web browser with no added software.[12 13] However, major security issues have been reported with Web3D and it still depends on the local client's GPU specification, which is highly variant.

Many visualization systems stream visualizations to specialized proprietary client-side viewers[14 15] or web browser plug-ins.[16 17] An important step for remote visualization was the development of virtual network computing (VNC).[18] VNC is a screen scraping application that transfers the whole desktop metaphor to a remote user. Research combining 3D graphics hardware with VNC are well documented and many companies have developed similar applications.[19 20] A step even further is a remote visualization system that parses the base application's graphical user interface, which then can be reorganized in a customized interface on the client side.[21] These projects mostly transfer still images compressed with JPEG, JPEG2000 or PNG or video formats such as MPEG and H.264.[22 23] Visualization can also be supported by grid-computing infrastructures that can provide server discovery mechanisms, parallel image analysis and rendering.[24 25]

Our literature search for interactive visualization systems that use web browsers without any additional software had sparse results. Bohne-Lang *et al*[26] describe a visualization application whose architecture goals shared our design criteria for minimal

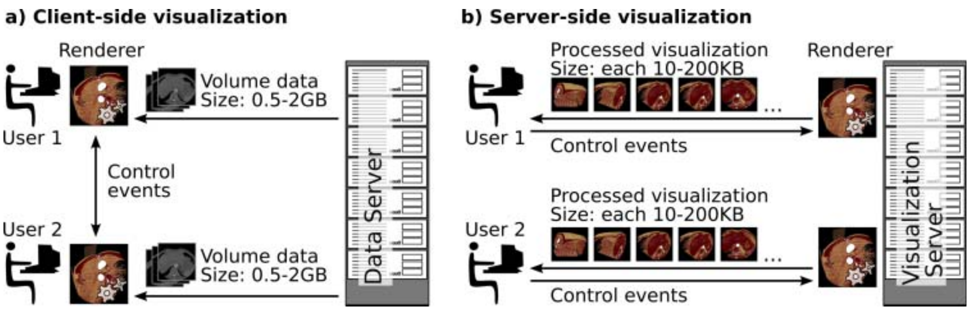
a) Client-side visualization
Renderer
User 1
Volume data Size: 0.5-2GB
Control events
User 2
Volume data Size: 0.5-2GB
Data Server
b) Server-side visualization
Processed visualization Size: each 10-200KB
Renderer
User 1
Control events
Processed visualization Size: each 10-200KB
User 2
Control events
Visualization Server

Figure 2 Client-side versus server-side rendering. Transferring a 0.5–2 GB dataset may take 7–27 min on a 10 Mbps network compared to just 10–100 ms sending the resulting visualization's images.

client-side requirements and robust accessibility, but is not optimized for high interactivity. Jourdain *et al*[27] describe ParaViewWeb, a web-based system that provides access to interactive VTK-based visualization. The user can choose between a Java, a Flash or the JavaScript client, which pulls images from the web browser via a web service interface. Projects for a combined usage of stereoscopic and collaborative visualization via pure web-based applications were not found in the literature.

Another possibility that satisfies no added software requirement is the plug-in-less video streaming introduced with HTML5. However, current HTML5 implementations introduce a buffering delay on the web browser that prevents real-time usage for interactive visualization. Better interactivity can be provided using either consecutively pulled single images from the browser or streaming 'motion JPEG' (MJPEG). Whereas pulling single images requires a separate connection for each image, MJPEG has one long-lasting open connection for all images and subsequently results in far better performance. Equally important, medical visualization, unlike photographic movies or hard-edged text and button displays, has fewer colors and soft edges, making JPEG images highly compressible and indistinguishable from lossless compression (although losslessness may not be justified in interactive visualization).

Previous web-based prototype

CoWebViz's initial prototype provided basic interactive and stereoscopic web-based volume visualization.[28] It was a server application that used MJPEG to stream visualizations with fixed JPEG quality setting and resolution to a client-side web browser without any additional software.

We tested the system successfully on the following unmodified standard web browsers: Google Chrome, Mozilla Firefox and Safari. Internet Explorer requires either additional software installation (eg, ActiveX control, Java applet) or a fallback mechanism with degraded performance by consecutively pulling single images. As web technologies mature, CoWebViz's modular, threaded architecture can be modified to comply.

For our visualization application, we use MedVolViz, our high-performance, distributed computing, medical rendering engine.[29] We run MedVolViz on a nine-node visualization cluster that has two consumer-grade graphics cards per node. This set-up produces real-time, high-quality, 3D stereoscopic volume renderings of clinical CT and MRI datasets with high resolution and interactive frame rates. We define interactivity as having a frame rate of at least 10 fps, with each frame rendered with maximal detail and resolution during visualization modifications (eg, while rotating or windowing).

We evaluated CoWebViz for medical education in our virtual anatomy class for undergraduate University of Chicago biology students. Shared 3D stereoscopic visualization was rendered elsewhere on campus.[4] CoWebViz simultaneously shared MedVolViz visualization with shared control to a group of clinicians, computer scientists and medical students located at Cardiff University, Wales. The Cardiff group has been participants in this collaborative virtual anatomy class since 2006 using antecedent streaming technology.[30] 2D anatomical drawings and cadaver photos were used to introduce the students to anatomical structures while streaming volume rendered reconstructions of patient CT datasets were used to present anatomical relationships in stereoscopy. The shared stereoscopic software set-up consisted of simply opening the collaboration via a bookmarked URL. However when CoWebViz streamed the visualization to multiple geographic destinations, that is network addresses, maintaining an interactive frame rate for the varied network bandwidth conditions required manual adjustments such as changing the JPEG quality setting. This resulted in the need for special adjustments for each unique network destination.

SYSTEM DESIGN

Parallel-processing architecture

Participants connect their browser to the visualization server at variant bandwidth. Adaptive adjustments are therefore essential in order to get optimal performance and image quality for the participant's unique network connection. As such adjustments were not possible with the first prototype's sequential architecture, we used a parallel software architecture that enables multiple participant connections each with their own adaptive image quality settings at the same time.

The architecture, illustrated in figure 3, is a composition of multiple exchangeable modules, with each running as a separate

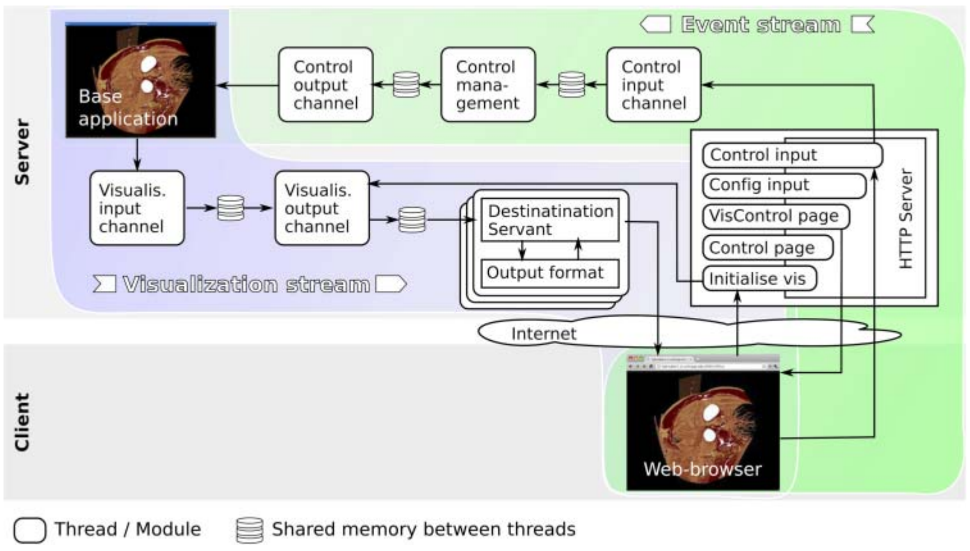

Figure 3 Two paths for visualization distribution and control event transfer are shown. Each rounded rectangle stands for one process with a specific functionality.

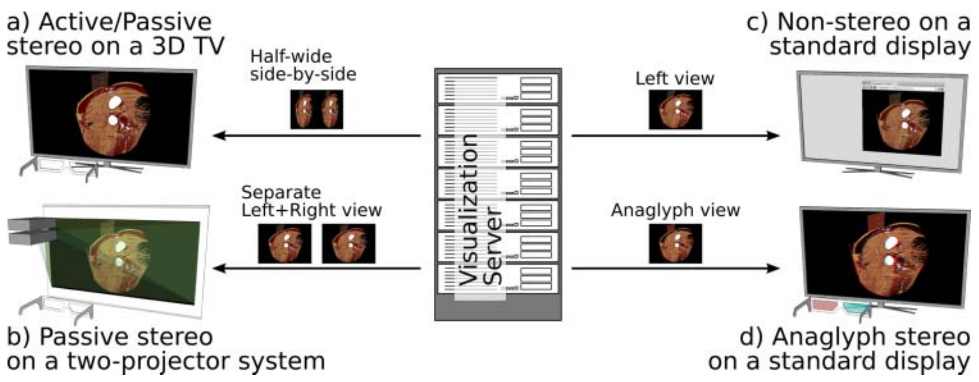
a) Active/Passive stereo on a 3D TV
Half-wide side-by-side
b) Passive stereo on a two-projector system
Separate Left+Right view
Visualization Server
c) Non-stereo on a standard display
Left view
Anaglyph view
d) Anaglyph stereo on a standard display

Figure 4 CoWebViz can be used in stereoscopy on three-dimensional (3D) high definition televisions (A), two-projector set-ups (B) and standard displays (D). It can also be used in monoscopy on standard displays (C).

thread (see rounded rectangles). The modules are combined into two processing chains, one for the image distribution and one for the control event transfer. The image distribution chain uses a visualization input channel to capture and process images from a base visualization application. It then consecutively distributes the newest image to each connected client via a visualization output channel. The control event transfer captures mouse and keyboard events on the client side. The events are transferred to the server via a control input channel where they are processed and forwarded via a control output channel to the base visualization application.

The interchangeable modular system design easily allows for a wide range of alternative configurations and usage scenarios. Input channels can grab visualization from video files, cameras and from base visualization applications (eg, MedVolViz). It is also possible to establish input channels that connect directly to a visualization application or that use external libraries to create the visualization directly in the input channel. Output channels distribute images via HTTP or store them as files on the server. The HTTP output channel allows for distribution of the visualization to multiple participants. Each participant is served via a dedicated processing module to handle the open connection and to format images into a specified format. We use output formats for 'MJPEG' and single JPEG files. Multiple modules can be used simultaneously to allow, for example, image streaming to web browsers and to record the session to a file.

The entire system uses a single built-in web server to handle all transmissions using only a single network port such as standard web server port 80. This gives the system a tremendous deployment advantage, as port 80 is normally not blocked by firewalls allowing for transparent operation.

Collaborative working

Many approaches have been used for controlling access to collaborative applications, such as a user explicitly requesting and being granted control, or restricting control to a single user only. We have added intuitive shared access control to CoWebViz that allows each participant to manipulate the visualization collaboratively. If one participant starts to modify the visualization, no other participants can gain control. After the visualization is not modified for a specified time, any participant can grab the control. Compared to aforementioned strategies, this fluid approach is simple and has the least user management overhead resulting in more dynamic and natural shared interactions, again maximizing the user experience by having the fewest possible software modes and controls for sharing (none).

Stereoscopic display devices such as 3D high definition televisions (HDTV) are now a widely available consumer technology. CoWebViz supports multiple visualization formats (including stereoscopic) allowing participants to choose the specific format that is best suited for their local output device, such as portable devices or standard monitors (with anaglyph glasses) or, if available to the user, larger scale projection-based stereoscopic set-ups or 3D HDTV (with stereo glasses). A session with four participants can simultaneously support all visualization and display types as illustrated in figure 4.

Integration

Due to the modular architecture of web-based systems, this visualization approach allows for a simple integration into other web-based applications. For example, integrating CoWebViz into a web-based hospital information system is possible with only minor technical effort, for example, security and encryption encapsulation. Integration into non-web-based applications is also possible, for example, a specific application for a mobile device.

PERFORMANCE OPTIMIZATION METHODS

Bandwidth optimization

We define a frame as one of a stream of images each with a specific JPEG quality and resolution that is sent from a server to a client over a network. Two consecutive frames differ when there is a modification to the server-side visualization. Stereoscopic medical visualizations are modified infrequently during the majority of sessions resulting in the generation of many identical, and thus, redundant frames. Sending only the modified frames is therefore an effective means to reduce bandwidth use.

None of the tested web browsers lost connection to the server using MJPEG if no new image was sent for 30 min. Some web browsers ordinarily have trouble displaying consecutive images using MJPEG because the most recent image is not displayed until the succeeding image is completely transferred. CoWebViz compares successive frames at the sub-pixel level and sends either modified images or the exact same image again if the visualization has not been modified for 100 ms. This delay value outputs at least 10 frames per second, empirically preserving interactivity over networks of varied capacity.

When used in collaboration, each client's display device may support very different resolutions (eg, handheld devices vs 3D HDTV). Therefore, the maximal supported resolution is automatically set for each client to reduce both the bandwidth usage (throughput) and the additional processor usage that is necessary for image resolution resizing.

Frame rate optimization

Frame rate is a frequency metric that is the fundamental performance measure in computer graphics because it quantifies a visualization's interactivity. For localized volume rendering, frame rate is dependent on hardware performance and software design. For example, a visualization's images can be sent at higher frequencies to a tethered display using faster GPU and/or by implementing parallelized rendering algorithms that take advantage of multiple GPU.

For remote, server-side volume rendering, assuming adequate GPU hardware, network bandwidth is often the most important empirical parameter controlling the client-side frame rate. That is, bandwidth determines the rate at which CoWebViz can practically push images over a network. CoWebViz's frame rate can thus be defined in terms of network bandwidth:

$$\text{Frame rate} = \text{network bandwidth} \times \frac{\text{frames}}{\text{Mbits}} \tag{1}$$

where

$$\text{Frame rate}'\text{s and network bandwidth}'\text{s units are}$$
$$\frac{\text{frames}}{\text{second}} \text{ and } \frac{\text{Mbits}}{\text{second}}, \text{ respectively}$$

To maintain the user's desired frame rate for given network conditions, CoWebViz iteratively adjusts the file size of each image sent over the network. The size of one resized image sent over the network can be expressed as its size per frame, or Mbits/frame. The extent of dynamic image resizing by CoWebViz's automatic adjustment algorithm is quantified by an abstract parameter we have defined as the streaming image quality (ϕ). For MJPEG, the two fundamental parameters that determine ϕ are JPEG quality and image resolution:

$$\begin{aligned}\text{Streaming image quality} &= \phi \\ &= \text{f(JPEG quality, image resolution)}\end{aligned} \tag{2}$$

As commonly cited in the literature, an image's JPEG quality and resolution produces a specific file size (abbreviated FS), which in turn equates a specific Mbits/frame value for a desired fixed minimal frame rate and given network bandwidth.

Performance is a direct consequence of the speed that a modified image is sent to the client and its ϕ. In other words, it is a trade-off between the maximization of the ϕ and the optimization of the frames per Mbit. The automatic adjustment algorithm determines the best possible streaming image quality for each participant for a desired frame rate and given network bandwidth. The algorithm relies on an initial network bandwidth test to determine an image's ideal file size (IFS):

$$\begin{aligned}\text{IFS} = \text{f(initial bandwidth test, number of client} \\ \text{views, user}'\text{s desired frame rate)}\end{aligned} \tag{3}$$

If the user starts to modify the visualization, a predetermined, set value for ϕ is used to compress the first modified image, resulting in a current image file size (CFS). For the next modified image, the difference between the preceding image's CFS and the IFS is used to calculate an optimized ϕ:

$$\text{IF CFS} \left\{ \begin{array}{l} < \text{IFS then increase } \phi \\ > \text{IFS then increase } \phi \end{array} \right\} \tag{4}$$

The algorithm then continuously optimizes ϕ to determine an optimal file size (FS(ϕ)) for each succeeding modified image that is as close to the IFS as possible. If no events occur for a specified time, ϕ is set to its maximal value (eg, maximum JPEG quality at maximum image resolution).

Rearranging equation (1) allows us to solve for FS as a function of ϕ:

$$\text{FS}(\phi) = \frac{\text{Mbits}}{\text{frame}} = \frac{1}{\text{frame rate}} \times \text{network bandwidth} \tag{5}$$

FS(ϕ) versus network bandwidth can clearly be expressed as an equation of a line where the y-intercept is zero and the slope is the inverse of the user-desired frame rate, or frame rate period. The goal of the automatic adjustment algorithm is to maintain a visualization's interactivity regardless of network bandwidth. In the HOME network case, we sacrifice streaming image quality to maintain interactivity. In the LAN case, when we have sufficient interactivity, we increase quality as much as the bandwidth allows.

Network latency optimization

Using MJPEG to push images to web browsers, we observed lagging visualization modifications on the client that occurred when images are sent too quickly. As HTTP primarily uses transmission control protocol connections, where every sent byte arrives at the destination, the frame rate has to be controlled from the server side. We reduced this lag by adding an artificial delay after each transmitted image that is unique to the network connection and image file size. We calculate the delay via an initial bandwidth test with the following equation:

$$\text{Delay} = \frac{\text{FS}(\phi)}{\text{network bandwidth}} \tag{6}$$

Equation (5) discusses the ideal case for optimal file size determination. Rearranging, we note that the network-specific delay and ideal frame rate period are identical. Delay can therefore be thought of as the empirically determined frame rate period that is necessary for images to be streamed at a user-defined frame rate without network lag.

$$\text{FS}(\phi) = \text{delay} \times \text{network bandwidth} \tag{7}$$

Using three example networks of varying bandwidth capacity, figure 5 elucidates CoWebViz's automatic adjustment algorithm's conceptual framework as quantified by equations (1) to (7) into one graph:

Event rate optimization

Web browsers were initially developed for stateless web applications. Efficient techniques to transfer input events from web browser clients to server applications are still limited. An HTML REST-style interface in which the browser calls a specific URL using a separate connection for each event works well if the client has a fast connection to the server (eg, >50 Mbps). Otherwise, long delays occur, resulting in a noticeable lag in visualization modification on the server and subsequently on the client. Using websockets, input event transfer is fast with significant improvement of the system's usability.[31] Websockets are already supported by some HTML5 web browsers and will very likely become more disseminated in the future.

Experimental set-up

We tested CoWebViz interactively streaming from our on-campus server across the internet to four different network

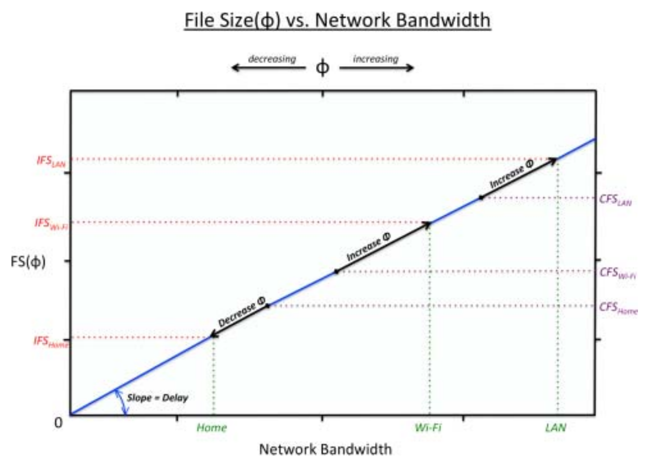

Figure 5 This concept graph illustrates optimal file size, FS(ϕ), in comparison to the network bandwidth. Each network results in a specific ideal file size (IFS). A modified image results in a current file size, which is always adjusted to the IFS, based on optimization of streaming image quality, ϕ.

endpoints to assess real-world usage and very low bandwidth usage: a high performing LAN (~90 Mbps), a fast WI-FI connection (~10 Mbps), a residential HOME network (~3 Mbps) and a LIMITED network constrained to very low bandwidth (~1 Mbps), as in table 1. To ensure comparability among optimization tests, we ran CoWebViz multiple times with a single modifying user at different times of the day for 3 min in a strict procedure with alternating 20 s modification and idle phases directed by a timer. In this way, all tests were performed as fully integrated user tests of actual end-to-end systems including mouse movement by the user (eg, at HOME). This ensures these results represent the user experience, not just machine-generated experiments. The data generated include a time-stamped row of data from each of the server and client for each processed image including JPEG image quality, file size, width and height. Results show averages over all applicable tests. The modification phases run the system at peak performance streaming as many frames as possible by continuous user mouse movement (maximizing events per second on the server side). Therefore, user variability in precise movement performed during modification should not substantially affect the results because network bandwidth is the limiting performance factor even on the fastest networks.

RESULTS

Performance measurements

CoWebViz's optimization algorithms are compared in the following section. The behavior of the network data throughput, JPEG quality and resolution is illustrated in table 1 and figure 6 among four different streaming configurations and four internet endpoints. Image resolution is a percentage defined as the ratio of the number of pixels of a sent image divided by the number of pixels of a baseline image, both with the same aspect ratio. For these tests, 1024×768 is the baseline resolution that is defined as 100%.

Baseline configuration (configuration 1)

All image frames are sent with a fixed ϕ (resolution: 1024×768, JPEG quality: 80).

Effect of sending only modified images

Comparison of sending all images (configuration 1) with sending only modified images that share the same fixed ϕ (configuration 2):

- No frames are sent when there are no modifications to the visualization.
- With modifications, the frame rate appropriately decreases considerably to 17 fps (−78%) and 16 fps (−54%) for LAN and WI-FI, respectively, which are the frame rates given by the base visualization application.
- Due to the a priori low bandwidth availability, the frame rate stays at approximately 8 fps for HOME and 3 fps for the LIMITED network.

Effect of the automatic quality adjustment

Comparison of sending only modified images with fixed ϕ (configuration 2) with sending only modified images using the automatic ϕ calculation (configuration 3):

- Without modifications, ϕ is set to its maximal values.
- With modifications and the automatic quality calculation, the JPEG quality increases by 5% and 7%, while the frame rate stays almost identical (±1 fps) on WI-FI and LAN.
- Using the same configuration to the HOME and LIMITED networks results in a higher, interactive frame

Table 1 Performance comparison of CoWebViz optimization algorithms

Configuration	Network type	Frames per second	Throughput in Mbps	Client CPU usage in %	Server CPU usage in %	JPEG quality	Image resolution
1. Send all images with fixed resolution (1024×768) and fixed JPEG quality of 80	LAN	76.2	25.2	86.7	51.7	80	100
	WI-FI	33.5	11.4	85.6	23.6	80	100
	HOME	7.7	2.6	46.1	13.0	80	100
	LIMITED	2.9	0.9	15.9	7.7	80	100
2. Send only modified images with fixed resolution (1024×768) and fixed JPEG quality of 80	LAN	16.7	5.6	54.1	18.5	80	100
	WI-FI	15.5	5.2	61.8	16.5	80	100
	HOME	8.2	2.6	44.6	13.6	80	100
	LIMITED	2.7	0.9	7.9	8.0	80	100
3. Send only modified images with AUTO resolution and AUTO JPEG quality	LAN	17.3	6.8	52.5	17.6	85.7	100
	WI-FI	15.5	5.4	63.7	21.7	84.1	97.6
	HOME	10.2	2.2	55.6	26.9	70.1	68.4
	LIMITED	10.1	0.3	14.6	21.2	23.7	7.6
4. Configuration 3 streaming with two eye perspective views (data shows 1 view)	LAN	16.1	5.1	77.3	73.9	79.7	89.3
	WI-FI	13.9	2.9	79.7	59.3	66.7	63.0

The four configurations show progressively increasing optimization as measured during modification (peak performance).
CPU, central processing unit.

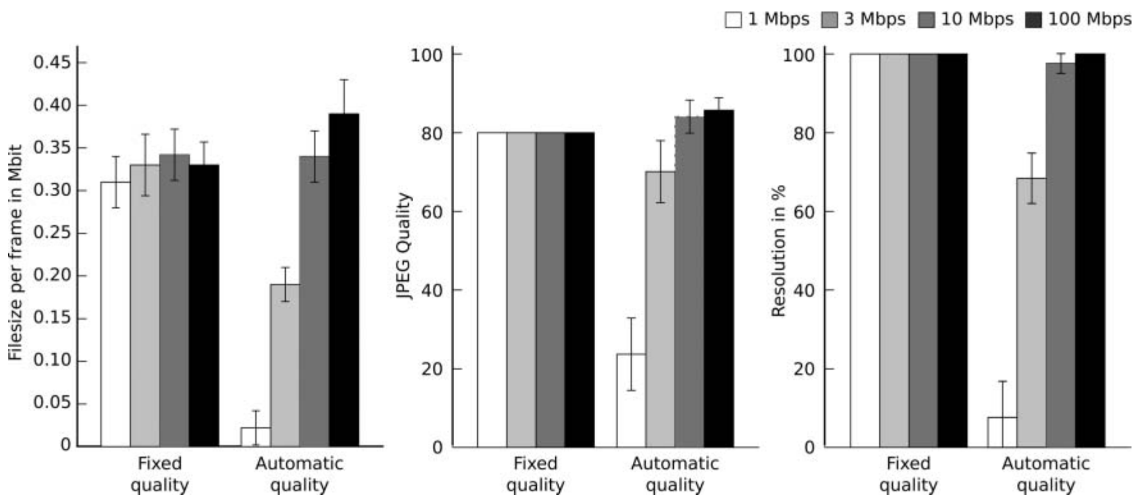

Figure 6 The bar graphs show the file size per frame, JPEG quality and resolution changes between fixed and automatic quality adjustments to four internet endpoints.

rate of 10 fps. This frame rate increase is associated with a 12% lower JPEG quality, a 17% lower resolution and a 25% higher client central processing unit (CPU) usage on the HOME machine.

- ▸ As shown in figure 6, this algorithm retains the performance and optimizes the JPEG quality towards the frame rate.

Stereoscopic modes

- ▸ On the client side, half-wide side-by-side input streams for 3D HDTV behave equally well as monoscopic streams despite larger resolution (1920×1080) required for the displays.
- ▸ The two-view stereoscopic visualization results in two similar streams. Each stream's JPEG quality values are slightly lower on LAN (frames: −1.2 fps, JPEG: −7%, resolution: −5%) and lower on WI-FI (frames: −1.6 fps, JPEG: −21%, resolution: −20%), compared to single-view streaming.

Scalability

Figure 7 demonstrates the performance of CoWebViz's streaming visualization simultaneously on each of one to six separate client machines each constrained to 10 Mbps or 3 Mbps. In each case, the ϕ value, shown as JPEG quality, the frame rate, and the throughput remain high and stable due to the parallel architecture of the code (threading). The same holds for mixed bandwidth scenarios. Of course, without server load balancing, the server CPU usage and the summarized throughput of all client connections increase with an increasing number of clients.

User experience

CoWebViz requires at most three steps to stream stereoscopic content: (1) instantiation of CoWebViz's server-side application (which could be web controlled); (2) loading of the visualization server's URL in the client-side web browser; and (3) for stereoscopic content, arranging the two browser windows according to the stereoscopic system's requirements, for example, typically

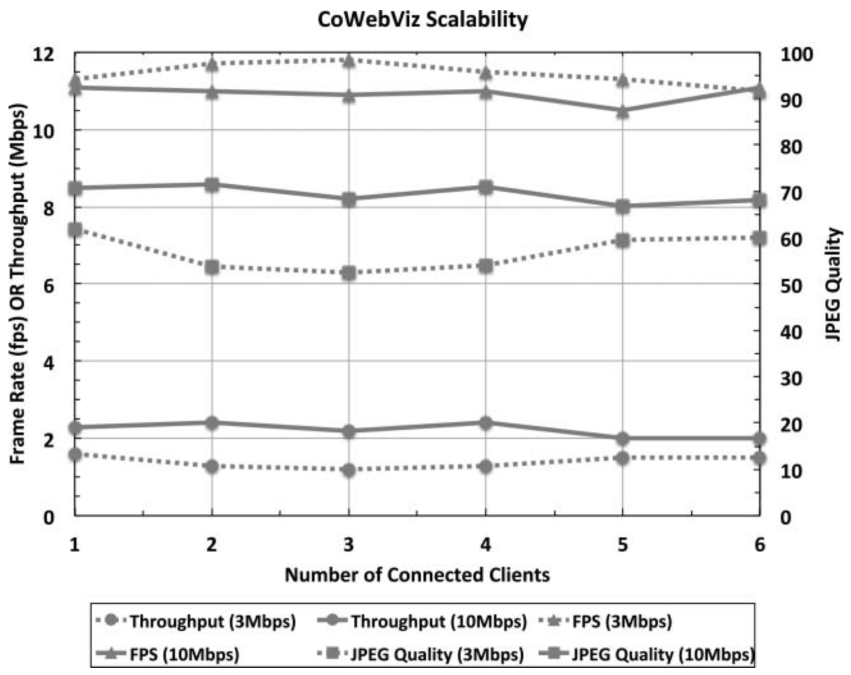

Figure 7 CoWebViz scalability is demonstrated by 12 experiments. Each of one to six separate client machines constrained to 10 Mbps or 3 Mbps bandwidth are simultaneously connected during visualization manipulation (peak performance). With an increasing number of clients, the frame rate, throughput and quality remain stable.

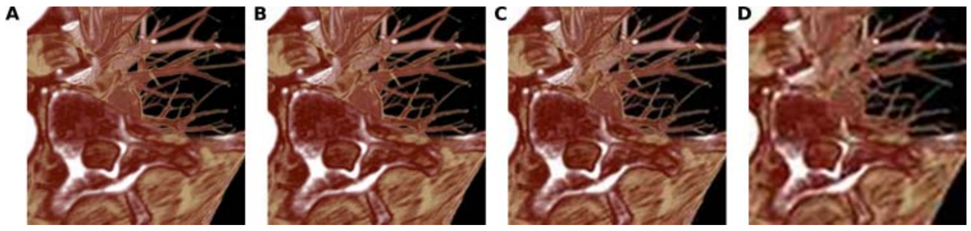

Figure 8 These images are example visualizations as streamed to users during a typical CoWebViz session. (A) Without mouse movement (no image modification), JPEG quality and resolution can reach 100 and 100%. During modification, JPEG quality and resolution vary depending upon bandwidth limits, (B) 70 and 64%, (C) 60 and 49%, (D) 35 and 18%.

a side-by-side arrangement for streaming left and right eye views for current generation 3D HDTV. In contrast to a native application, CoWebViz does not require installation of special software, thereby skipping special intervention by organizational IT staff required by restricted user permissions common to security policy on corporate computers.

Because the interactive frame rate is automatically maintained above 10 fps in real-world usage, the user experience is best depicted by visually inspecting the effect of manipulating CoWebViz's parameters from an example session. Without mouse movement, the result is maximum quality and no network usage. Modifications result in specific ϕ calculated for each connection (table 1 and figure 6). Figure 8 includes example images with the corresponding JPEG quality and resolution in a static phase (figure 8A) and modification phases (figure 8B–D), such as rotating or windowing a stereoscopic volume rendering.

DISCUSSION

Striking the optimal balance between quality and performance is an important design factor when streaming video when intra and inter-frame compression is commonly used. Using a web browser without the installation of additional third-party plug-ins restricts the available options for finding that balance. HTML5 video streaming can be used, but it results in buffering delays that prevent interactive usage. Sending single JPEG or PNG images consecutively is therefore the only workable solution; both formats are supported on all web browsers. We decided to use MJPEG as it is supported by many web browsers and has the best performance of all current options. In contrast to pulling images via long polling from the client, MJPEG pushes them from the server (comparable to video streaming), which leads to higher frame rates, key to the maximizing the user experience.

The enhancement that results in CoWebViz's best usability is the combination of the automatic quality adjustment and delay algorithms. With low bandwidth connections (eg, HOME), the algorithms retain performance, that is frame rate, with a drop in the streaming image quality. On high bandwidth connections, bandwidth is conserved while frame rate and streaming image quality are maximized. The enhancement that results in the best optimization in terms of bandwidth usage sends only the images that have been modified. Using non-inter-frame compression leads to high bandwidth usage during image modification (frame rate is maintained). In contrast to the described peak performance data, clinical users and educators usually modify medical visualizations with careful deliberation. The percentage of time that medical visualizations are dynamically modified in practice thus tends to be very small (as distinguished from time when they are static). This results in fewer modifications relative to our testing procedures leading to fewer modified frames per second, culminating in substantially lower overall bandwidth usage.

Empirically observed, the delay and automatic quality algorithms maintain interactivity for fluent modification without the need for special configurations or user adjustments for each client. This leads to fewer interaction events on the client and subsequently fewer modifications on the server, maintaining usability.

The scalability of a server-side rendering relies on the network and the server hardware. With CoWebViz installed on a desktop-class server, it is possible to provide two-view stereoscopy for up to four remotely collaborating participants. Certain system changes (eg, providing quality classes, instead of individual quality) and network configurations (eg, multicast) would be necessary to scale to a larger number of participants.

CONCLUSION

A key element for the robust usage of advanced medical visualization techniques and technologies is ease of user access. Although many projects have shown the benefits of high-performance computing for medical image analysis, ease of access was rarely a focal point. We suspect this is a reason for limited use.

CoWebViz enables a collaborative stereoscopic visualization hub that combines high performance and high-quality visualization with lightweight clients. A client-side web browser with no added software allows for cross-platform access to interactive, collaborative, stereoscopic medical visualizations on major operating systems (Linux, Unix, Windows, Mac OS, iOS) and processor architectures, including mobile devices. In addition to the ubiquity enabled by only requiring available web browsers, CoWebViz's user experience is optimized, without the need for interface controls, by automatically maintaining the best image quality and frame rate for the available network bandwidth at each location, and seamlessly shared control among collaborators.

CoWebViz's approach to sharing both interactive and stereoscopic visualization can be seamlessly integrated into existing web applications and could therefore coexist with current web-based hospital information systems. The approach also enhances usability for the end-user because there are no prerequisite skills or software installations required.

CoWebViz's feasibility and advantages facilitated the use of virtual stereoscopic volume rendering technologies in pre-med and medical education for undergraduate biology and medical students at the University of Chicago. A needs assessment has identified applications for clinical use, particularly surgical applications.

Acknowledgements The servers were generously housed and supported by the Computation Institute of the University of Chicago in collaboration with the Department of Surgery.

Contributors All three authors have made substantial contributions to each of: conception and design, acquisition of data, analysis and interpretation of data, drafting and revising the article critically for important intellectual content, and final approval of the version to be published.

Funding MK was supported in part by a fellowship within the PhD Program of the German Academic Exchange Service (DAAD). NMP was supported in part by University of Chicago Institute of Translational Medicine (NIH/CTSA UL1 RR024999). All authors were supported in part by NorthShore University Health System.

Competing interests None.

Provenance and peer review Not commissioned; externally peer reviewed.

Data sharing statement No unpublished data are available.

REFERENCES

1 Muller MA, Marincek B, Frauenfelder T. State of the art 3D imaging of abdominal organs. *JBR-BTR* 2007;90:467–74.
2 Klein J, Bartz D, Friman O, *et al*. Advanced algorithms in medical computer graphics. State of the Art Reports, Aire-la-Ville, Switzerland: Eurographics 2008.
3 Huang HK. From PACS to web-based ePR system with image distribution for enterprise-level filmless healthcare delivery. *Radiol Phys Technol* 2011;4:91–108.
4 Kaspar M, Dech F, Parsad NM, *et al*. Web-based stereoscopic visualization for the global anatomy classroom. *Stud Health Technol Inform* 2011;264–70.
5 Sielhorst T, Bichlmeier C, Heining S, *et al*. Depth perception-a major issue in medical AR: evaluation study by twenty surgeons. *Med Image Comput Comput Assist Interv-MICCAI 2006* 2006;9:364–72.
6 Fraser JF, Allen B, Anand VK, *et al*. Three-dimensional neurostereoendoscopy: subjective and objective comparison to 2D. *Minim Invasive Neurosurg* 2009;52:25–31.
7 Arguiñarena EJC, Macchi JE, Escobar PP, *et al*. Dcm-ar: a fast flash-based Web-PACS viewer for displaying large DICOM images. *Conf Proc IEEE Eng Med Biol Soc* 2010;3463–6.
8 Welter P, Hocken C, Deserno TM, *et al*. Workflow management of content-based image retrieval for CAD support in PACS environments based on IHE. *Int J Comput Assist Radiol Surg* 2010;5:393–400.
9 Hunter J, Henderson M, Khan I. Collaborative annotation of 3D crystallographic models. *J Chem Inf Model* 2007;47:2475–84.
10 Tongleamnak S, Covavisaruch N, Vatanawood W. Programmable web-based volume visualization. The 1st ECTI Annual Conference; 13–14 May 2004, Thailand. ECTI Association 2004.
11 Temkin B, Acosta E, Hatfield P, *et al*. Web-based three-dimensional virtual body structures: W3D-VBS. *J Am Med Inform Assoc* 2002;9:425–36.
12 John NW. The impact of Web3D technologies on medical education and training. *Comput Ed* 2007;49:19–31.
13 Melzer K, Lipinski HG, Grönemeyer DHW. X3D-technologies for medical image visualization (abstract). *ACM SIGGRAPH*;8–12 August 2004, Los Angeles.
14 Lamberti F, Sanna A. A streaming-based solution for remote visualization of 3D graphics on mobile devices. *IEEE Trans Vis Comput Graph* 2007;13:247–60.
15 Engelmann U, Munch H, Schroter A, *et al*. A teleradiology concept for entire Greenland. *Int J Comput Assist Radiol Surg* 2006;1:121.
16 Suwelack S, Maier S, Dillmann R. Web-based interactive volume rendering. *Stud Health Technol Inform* 2011;163:635–7.
17 Lamberti F, Sanna A, Henao Ramirez EA. Web-based 3d visualization for intelligent street lighting. Proceedings of the 16th International Conference on 3D Web Technology; 20–22 June 2011 Paris. ACM, 2011:151–4.
18 Richardson T, Stafford-Fraser Q, Wood KR, *et al*. Virtual network computing. *IEEE Internet Computing* 1998;2:33–8.
19 Stegmaier S, Diepstraten J, Weiler M, *et al*. Widening the remote visualization bottleneck. Proceedings of the 3rd International Symposium on Image and Signal Processing and Analysis; 18–20 September 2003, Rome. ACM, 2003;1:174–9.
20 The VirtualGL Project. VirtualGL: 3D without boundaries. http://www.virtualgl.org (accessed 29 Feb 2012).
21 Lamberti F, Sanna A. Extensible GUIs for remote application control on mobile devices. *IEEE Comput Graph Appl* 2008;28:50–7.
22 Noumeir R, Pambrun J. Using JPEG 2000 interactive protocol to stream a large image or a large image set. *J Digit Imaging* 2011;24:833–43.
23 Hewage C, Karim H, Worrall S, *et al*. Comparison of stereo video coding support in MPEG-4 MAC, H. 264/AVC and H. 264/SVC. Proceedings of IET Visual Information Engineering-VIE07; 25–27 July 2007, London. Royal Statistical Society, 2007.
24 Kunihiko N, Toru H, Kenji O, *et al*. Volume rendering using grid computing for large-scale volume data. *11th IEEE International Conference on Computer-Aided Design and Computer Graphics*, 19–21 August 2009, Huangshan, China. IEEE 2009.
25 Hastings S, Oster S, Langella S, *et al*. A grid-based image archival and analysis system. *J Am Med Inform Assoc* 2005;12:286–95.
26 Bohne-Lang A, Groch W, Ranzinger R. AISMIG—an interactive server-side molecule image generator. *Nucleic Acids Res* 2005;33:W705–9.
27 Jourdain S, Ayachit U, Geveci B. ParaViewWeb, a web framework for 3d visualization and data processing. *IJCISIM* 2011;3:870–7.
28 Kaspar M, Parsad NM, Silverstein JC. CoWebViz—interactive collaborative sharing of 3D stereoscopic visualization among browsers with no added software. 1st ACM International Health Informatics Symposium; 11–12 November 2010, ACM Arlington, VA. ACM, 2010.
29 Silverstein JC, Walsh C, Dech F, *et al*. Multi-parallel open technology to enable collaborative volume visualization: how to create global immersive virtual anatomy classrooms. *Stud Health Technol Inform* 2008;132:463–8.
30 Silverstein JC, Walsh C, Dech F, *et al*. Immersive virtual anatomy course using a cluster of volume visualization machines and passive stereo. *Stud Health Technol Inform* 2007;125:439–44.
31 Fette I, Melnikov A. The WebSocket Protocol. RFC 6455 (Proposed Standard), IETF Dec 2011.