Journal of Digital Imaging (2022) 35:68–76
https://doi.org/10.1007/s10278-021-00544-0

ORIGINAL PAPER

A Comparison of WebRTC and Conventional Videoconferencing for Synchronized Remote Medical Image Presentation

Vishal Patel[1] · Charles H. Li[2] · Van Rye[2] · Chia-Shang J. Liu[2] · Alexander Lerner[2] · Jay Acharya[2] · Anandh G. Rajamohan[2]

Received: 16 November 2020 / Revised: 18 October 2021 / Accepted: 11 November 2021 / Published online: 21 December 2021

Abstract
DICOM viewers must fulfill roles beyond primary diagnostic interpretation, including serving as presentation tools in teaching and multidisciplinary conferences, thereby enabling multiple individuals to review images collaboratively in real time. When in-person gathering is not possible, a variety of solutions have been deployed to maintain the ability for spatially separated users to view medical images simultaneously. These approaches differ in their backend architectures, utilization of application-specific optimizations, and ultimately in their end user satisfaction. In this work, we systematically compare the performance of conventional screensharing using a videoconferencing application with that of a custom, synchronized DICOM viewer linked using Web Real Time Communications (WebRTC) technology. We find superior performance for the WebRTC method with regard to image quality and latency across a range of simulated adverse network conditions, and we show how increasing the number of conference participants differentially affects the bandwidth requirements of the two viewing solutions. In addition, we compare these two approaches in a real-world teaching scenario and gather the feedback of trainee and faculty radiologists, who we found to favor the WebRTC method for its decreased latency, improved image quality, ease of setup, and overall experience. Ultimately, our results demonstrate the value of application-specific solutions for the remote synchronized viewing of medical imaging, which, given the recent increase in reliance on remote collaboration, may constitute a significant consideration for future enterprise viewer procurement decisions.

Introduction

A DICOM viewer constitutes an essential component of a digital radiology practice. The role of the image viewer is multifaceted and extends beyond its use by individual radiologists for primary diagnostic interpretation. Importantly, the viewer is often used to display images to multiple physicians simultaneously. In academic practices, this occurs routinely during trainee "readout," an educational process in which attending staff radiologists perform live review of imaging studies with a resident who has already preliminarily inspected the cases in order to verify the trainee's findings and interpretation. In addition to the trainee responsible for the initial review of the case, additional residents and staff may be recruited to the readout of particularly interesting or challenging cases, and medical students frequently also participate to gain practical exposure to the field of radiology. Another situation requiring the display of images to multiple physicians is that of multidisciplinary conferences. An instance of the DICOM viewer often plays a central role during these meetings in which several physicians from various specialties (e.g., surgery, oncology, radiation oncology, pathology, and radiology) all convene to review important diagnostic information about a patient's case and decide on an appropriate course of treatment.

Conventionally, this simultaneous viewing of imaging by multiple users has been achieved by physically convening the physicians around a single instance of the DICOM viewer, which may be displayed on a standard radiology workstation in the case of trainee readout or on a projection screen in multidisciplinary conferences. When physical gathering is impractical, however, some solution for synchronized remote viewing is required. The need for remote

✉ Vishal Patel
vishal.patel@med.usc.edu

[1] Mark and Mary Stevens Neuroimaging and Informatics Institute, Keck School of Medicine, University of Southern California, 2025 Zonal Ave, Los Angeles, CA 90033, USA

[2] Department of Radiology, Keck School of Medicine, University of Southern California, 1500 San Pablo Street, 2nd Floor, Los Angeles, CA 90033, USA

image viewer synchronization has been greatly magnified during the COVID-19 pandemic, with many radiology departments restructuring their educational models to allow for social distancing in their reading rooms and multidisciplinary conferences moving to entirely virtual formats [1–4].

There are several approaches to achieving the synchronized presentation of medical imaging among remote viewers. In one common approach, this scenario is understood as a specific instance of the more general problem of screen sharing for real-time collaboration. A wide range of screen sharing solutions are available, including dedicated enterprise videoconferencing systems and consumer-oriented group messaging applications. The underlying protocols are often proprietary, but may be derived from or similar to open technologies such as virtual network computing (VNC), NX, and Jingle [5–7]. Ultimately, these solutions, due to their general nature, rely on the capture and transmission of screen pixel data over the network link, with various caching and compression optimizations applied to reduce bandwidth utilization and latency. In contrast to general screen sharing software, some medical image viewers include built-in conferencing capabilities limited to image review. Though again, the exact details are often proprietary, because the scope of synchronization is more narrowly defined, an application-specific technique can be used: rather than handling arbitrary pixel data, these viewers can pre-fetch the studies at both endpoints and synchronize only the viewer state over the network. This is analogous to the approach used by some multiplayer online video games that load large assets such as character and environment models locally and transmit only game state updates to each client [8].

The transmission of the pixel data or viewer update instructions can take place over either a client–server architecture or a peer-to-peer (P2P) architecture (Fig. 1). A client–server model may reduce bandwidth requirements at the viewing endpoints and enable the inclusion of clients with limited network, computational, or power resources. However, this model concentrates the resource demands at the server, and the inclusion of a server between the viewing stations also results in increased latency. In addition, if the server is controlled by an outside entity, as is the case for many commercial videoconferencing systems, the client–server architecture may require the initiation of a business associate agreement prior to the transmission of protected health information [9]. The P2P model, on the other hand, directly connects all viewers without an intermediary, but increases the bandwidth demands on the endpoints as the number of viewers increases because each participant must establish its own direct connection with every other participant.

These disparate approaches to viewer synchronization and the different underlying network architectures can result in varying levels of performance and degradation of the viewing experience under certain conditions. Given the active deployment of these fundamentally different technologies and the current increase in the need for reliable, synchronized remote viewing of medical images, we set out to systematically compare the latency and image quality characteristics of a conventional videoconferencing system (CVS) transmitting pixel data via a central server and a tailored viewing application communicating only state updates over a P2P architecture. The latter application was developed using the Web Real Time Communications (WebRTC) application programming interface (API), a set of related standards for enabling real-time sharing of media streams and arbitrary data through a web browser [10]. In this work, we quantitatively compare these approaches using

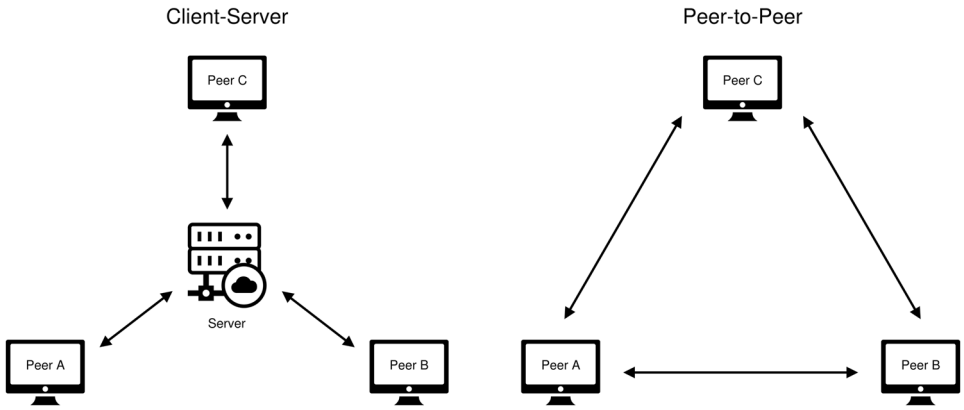

Fig. 1 Schematic comparison of client–server and peer-to-peer network architectures. In the client–server model, each peer establishes a connection only with a server. The server is an intermediary, and so data transmitted between peers must traverse at least two network hops (double-headed arrows). Increasing the total number of participants does not affect the individual peers, but it does increase the number of connections required by the server. Alternatively, in a peer-to-peer model, each peer establishes a direct connection with every other peer, and thus, data exchange requires only a single hop. However, increasing the number of participants results in increasing connection demands for every peer

controlled, simulated network conditions, and we also report radiologists' qualitative assessment of the methods following a typical readout scenario. We emphasize that our objective is not to implement a feature-complete alternative to a commercial CVS, nor a full-featured PACS thick client, but rather to investigate how an application-specific backend design affects overall system performance for the synchronized remote viewing of medical images.

Methods

WebRTC Image Viewer

We developed a WebRTC-enabled medical image viewer to test the hypotheses that this architecture would yield decreased latency and improved image quality compared to screen sharing with a CVS. While a number of open source, web-based DICOM viewers have been developed, we chose to construct the image display component of our web application around a customized version of Papaya, a JavaScript DICOM viewer distributed by the Research Imaging Institute at the University of Texas Health Science Center at San Antonio [11, 12]. We modified the Papaya source code to insert hooks to enable the registration of callback methods to be executed whenever the view was updated.

We assigned each user a random peer identifier upon application startup; communication of this identifier via an outside channel enables connection initiation, analogous to the meeting identifier used in videoconferencing applications that is commonly distributed via e-mail invitations. Between each pair of peers, we established an RTCPeerConnection per the WebRTC API. Establishing a connection between peers requires the initial negotiation of capabilities (supported codecs) over Session Description Protocol (SDP) and resolution of routing via Interactive Connectivity Establishment (ICE) [13, 14]. This process is referred to as "signaling" and is facilitated by an intermediary signaling server. We used the open source PeerJS library to carry out this exchange of metadata between peers [15]. Of note, only the metadata needed to initiate the connection is transmitted to the signaling server; all media streams and application data are transmitted via encrypted protocols directly between the two peers as described below.

Having established an RTC connection between two peers, we added a MediaStreamTrack to the connection to transmit an audio channel from the users' microphones, enabling voice communication with user-controlled mute capability. WebRTC communicates encrypted audio streams according to the Secure Realtime Protocol (SRTP) with encryption keys negotiated via Datagram Transport Layer Security (DTLS), transmitted over User Datagram Protocol (UDP) at the transport layer [16, 17]. To enable the unique approach we propose, we also attached an RTCDataChannel to the connection, enabling the bidirectional transfer of arbitrary data. WebRTC data messages are communicated through the Stream Control Transmission Protocol (SCTP) at the application layer, encrypted by DTLS, and transported over UDP [18]. SCTP enables the ordered, reliable delivery of messages over the conventionally unreliable UDP transport, while still limiting network overhead relative to Transmission Control Protocol (TCP). Figure 2 provides a schematic overview of the RTC connections established for our application.

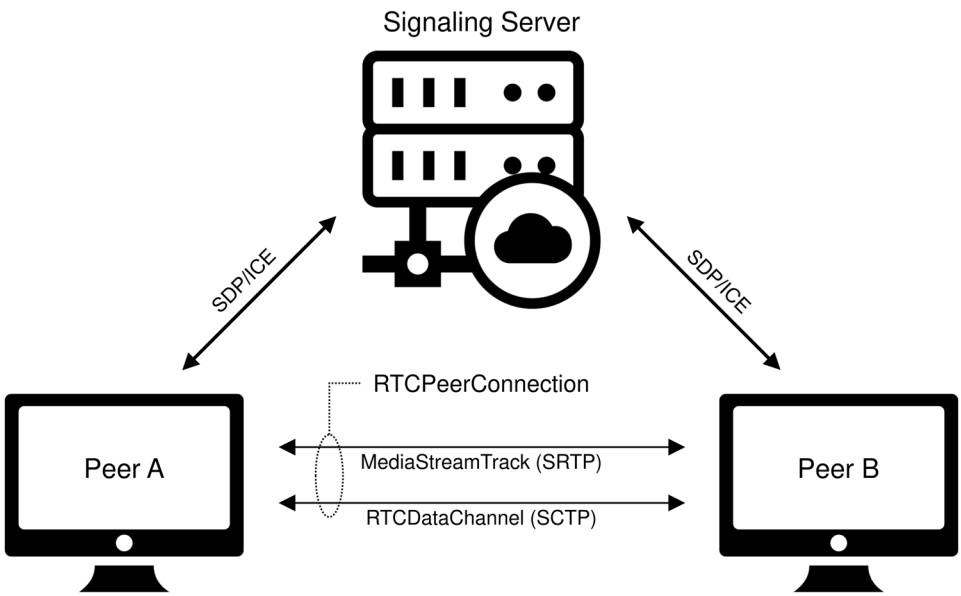

Fig. 2 Schematic overview of the WebRTC connections established for remote DICOM viewer synchronization. Initial connection negotiation is facilitated by a signaling server, but the continuous data updates are passed directly between peers over an RTCPeerConnection that encapsulates a MediaStreamTrack carrying microphone audio and an RTCDataChannel carrying viewer state updates

Our application specifies the following types of messages to be communicated over the data channel:

- Peer Roster: The list of known peers is broadcast whenever a new RTC connection is established, enabling the automated construction of a fully connected mesh network as new peers join.
- Viewer Control: This message type provides conflict resolution to ensure only a single peer can manipulate the viewer at a given time.
- Worklist Position: The application is organized around a worklist which includes multiple studies available for viewing. The peer in control of the viewer broadcasts the index of the study from the worklist that should currently be presented in the viewer. Whenever the index changes, the new image is downloaded into process memory and loaded into Papaya.
- Viewer State: This message type, broadcast continuously by the peer in control of the viewer and triggered by the aforementioned Papaya hooks, provides the essential parameters needed for other peers to synchronize their views: slice positions, window center and width, zoom and pan transformations, and crosshair visibility. Serialization of these parameters was performed per the MessagePack specification to reduce packet sizes [19].
- Chat Message: Textual messages may be posted to a chatroom by any peer. This functionality enables communication by peers without microphone capability, can be used to avoid interrupting ongoing spoken conversations, and facilitates the sharing of information such as hyperlinks.

To illustrate the capabilities of this technology, we deployed a demonstration site for an RTC-Enabled, P2P-Linked Imaging Conference Application (REPLICA) at https://radiogra.ph/rtc/.

Quantitative Assessment

In order to measure the performance characteristics of the WebRTC synchronization method relative to CVS screen-sharing across a variety of network conditions in a controlled environment, we deployed two identical Linux (kernel version 4.19) virtual machines (VMs) connected by a virtual network to each other and to the host. We refer to this pair of VMs as sender and recipient; the former indicates the VM in control of manipulating the image in the viewer, and the latter designates the VM that updates its display in response to information transmitted by the sender over the virtual network. Of note, this terminology does not consider the audio stream data, which is always transmitted bidirectionally in our tests. WebRTC synchronization was evaluated by loading the aforementioned application in Mozilla Firefox (version 68.12.0esr) inside both VMs and linking the viewers using an RTCPeerConnection as described. To evaluate the CVS, we loaded the same browser-based viewing application in the sending VM, but we programmatically disabled the RTC communication components. Instead, Zoom Video Communications' Cloud Meetings application (version 5.2.458699.0906) was launched on both VMs and used to share the browser window to the receiving VM. All quantitative tests were conducted with a computed tomography (CT) study of the head loaded into the Papaya viewer. We chose to use CT as the imaging modality for assessment because its display typically places high data rate demands on the viewer application (typical matrix size: 512×512 or greater, typical bit depth: 12 or greater, peak stack scroll rates: 15 slices per second or greater). In addition, audio from a recording of the US presidential debate from September 26, 1960, was continuously piped as input to virtual microphones on both VMs to produce a reproducible audio stream of simulated speech and a realistic network and computational burden [20].

We examined performance across a variety of network conditions by adjusting the traffic control queuing discipline of the Linux kernel in the sending VM to emulate networks with varying degrees of latency (0–512 ms) and packet loss (0–10%). We programmatically triggered Papaya to increment the displayed image slice in the sending VM once per second. For each set of network conditions, over the course of 100 displayed slice updates, the screens of both guest VMs were captured from the host at a sampling rate of 120 frames per second, and the output was saved in an uncompressed format. This recorded video stream was then analyzed using OpenCV to determine the time delays with which display updates on the receiving VM lagged those of the sending VM [21]. In addition, we measured the magnitude of image compression artifacts at the receiving VM by calculating the mean squared error (MSE) relative to the image displayed on the sending VM. The performance of the RTC and CVS synchronization methods as captured by both of these metrics was statistically compared using one-way analysis of covariance (ANCOVA) to isolate the effect of the synchronization technology from the network condition covariate.

Also, given the difference in network architectures between the RTC and CVS linking methods, we examined the changes in maximum bandwidth requirements as the number of connected peers was increased. For this test, we updated the slice displayed by Papaya on the sending VM at a rate of 30 Hz (i.e., perceptually continuously), and we cloned the receiving VM to simulate up to 32 connected peers. We recorded inbound and outbound network data rates on the sending VM and one of the receiving VMs over 100 one-second windows, and again compared the network utilization between the two synchronization technologies using one-way ANCOVA.

Qualitative Assessment

We also sought to gauge the practical impact of the RTC synchronization technology by gathering the opinions of eight trainee and five attending radiologists. To this end, we selected 10 head CT examinations from our institutional teaching file and divided them into 2 worklists containing 5 studies each. Radiology trainees were asked individually to examine all 10 cases and then contact an attending neuroradiologist for remote review. Cases from one worklist were reviewed by screensharing over a CVS (Zoom), and cases from the other worklist were reviewed using the WebRTC synchronization method. Immediately following each review session, the trainee and faculty member were asked to complete a survey containing a set of 5-point Likert items comparing the two synchronization methods for perceived image lag, image quality, audio lag, audio quality, ease of setup, and overall experience. All raters were provided with a definition of terms to encourage uniform interpretation: image lag referred to latency between image manipulation and screen update, image quality referred to degree of degradation by compression artifacts, audio lag referred to perceived latency between speech production and speaker driver response, audio quality referred to perceived clarity of speech, and ease of setup was to include time and complexity of actions required to establish a collaborative session. Because the number of trainees exceeded the number of faculty, some attendings participated in more than one review session; however, each subject completed the survey only once, following his or her initial experience. Survey results were statistically compared using the Mann Whitney test, and a Bonferroni correction was applied to control the false discovery rate in the setting of multiple hypothesis tests.

Results

The results from our quantitative analysis of RTC and CVS performance over simulated network conditions are summarized in Fig. 3. We found that the proposed RTC data channel synchronization technique resulted in decreased image

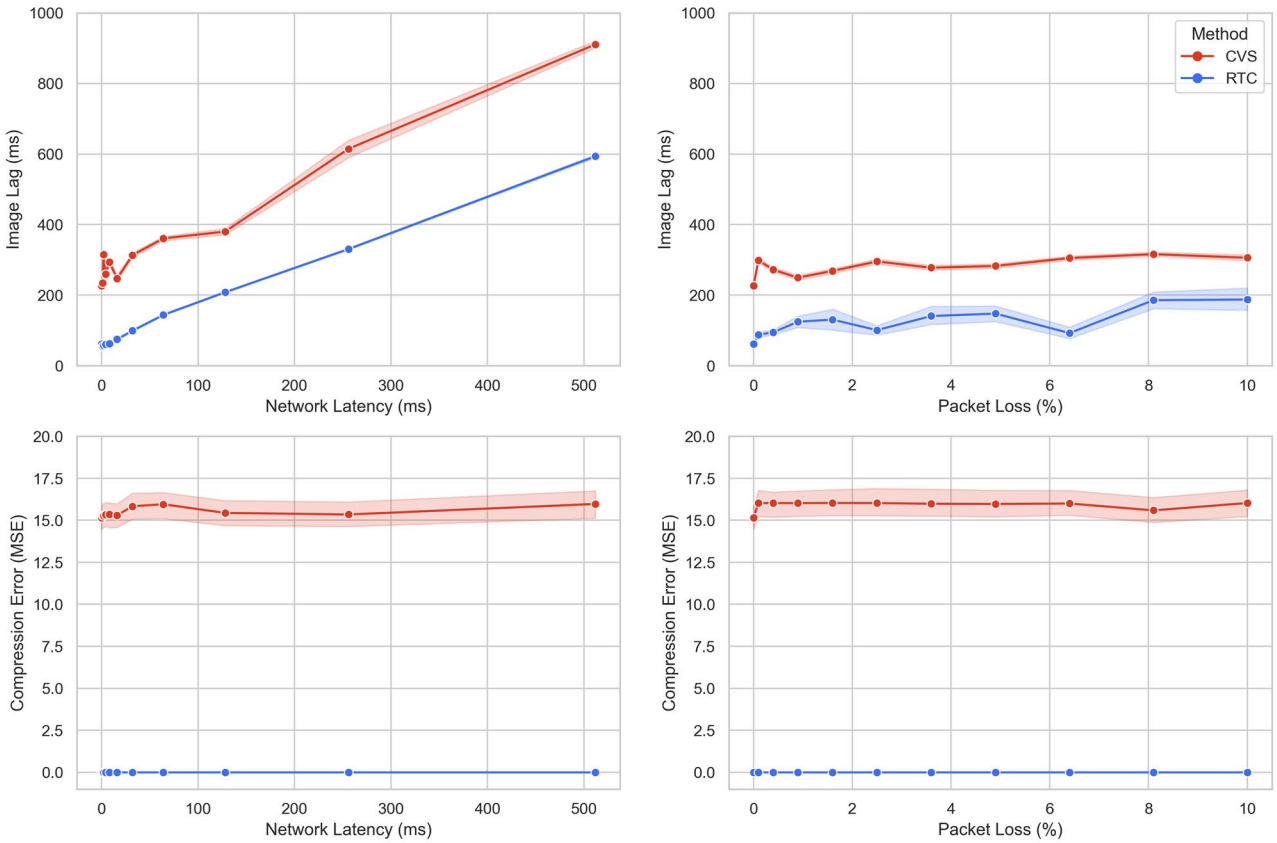

Fig. 3 Quantitative comparison of WebRTC and a CVS for synchronized viewing of medical images over adverse network conditions. Image update lag (top row) and error introduced by compression artifacts (bottom row) are shown as a function of increasing network latency (left column) and increasing packet loss (right column). Plotted points indicate means over 100 viewer updates and associated shaded regions span the standard error

lag relative to CVS screensharing at all levels of simulated network latency ($p < 0.001$). We also observed that the image lag was significantly dependent on the amount of latency ($p < 0.001$), increasing approximately linearly for both methods as network latency increased. Similarly, we found that RTC synchronization resulted in reduced lag relative to the CVS in the setting of simulated packet loss ($p < 0.001$). Again, increasing packet loss also resulted increased image lag for both methods ($p < 0.001$), though the magnitude of the effect was less pronounced than that for network latency over the range tested.

Regarding image compression artifacts, we observed no compression-related error in the RTC-synchronized viewer as per the design — full resolution images are loaded on each peer locally, and pixel data is not transmitted over the RTCPeerConnection. We did, however, detect compression related artifacts in the images transmitted over the CVS connection. A representative example of these artifacts are provided in Fig. 4, which shows that the CVS-compressed image exhibits attenuation of high frequency information like the boundaries between gray matter and white matter in cortical and subcortical areas, the edges of thin linear structures like narrow sulci and dural reflections, and the quantum mottle inherent to CT. The MSE associated with the compression artifacts in the CVS was statistically greater than that of RTC (i.e., greater than zero) for both the network latency ($p < 0.001$) and packet loss ($p < 0.001$) tests. The severity of compression-related artifacts appeared to be independent of the degree of network latency ($p = 0.502$) or packet loss ($p = 0.913$) over the ranges tested.

Our examination of network data rates during continuous image updates is summarized in Fig. 5. We found a significant dependence of network utilization on synchronization method ($p < 0.001$), and post hoc tests revealed that as the number of connected peers increased, the P2P architecture of the RTC synchronization method resulted in corresponding increases in inbound and outbound data transmission rates on both the sending and receiving endpoints ($p < 0.001$ in each case). In contrast, the centralized architecture of the CVS yielded approximately constant network utilization in the context of increasing peer count. For groups of 4 or fewer individuals (i.e., 3 or fewer connected peers), all measured data rates were less for the RTC method than for the CVS. Maximum RTC bandwidth requirements exceeded those of the CVS for the sender inbound and receiver outbound categories when group size exceeded 4, for the sender outbound category once group size surpassed 13, and for the receiver inbound category for groups larger than 27 members.

Our qualitative testing showed that subjects rated the RTC experience superior to the CVS for perceived image quality, image latency, ease of setup, and overall experience ($p < 0.001$ for all cases). There was no significant difference in the RTC and CVS ratings for perceived audio quality or audio latency. Stratifying responses by role (i.e., trainees or attendings) did not alter the results.

Discussion

We have studied the performance of WebRTC as a technology for linking remote DICOM viewers for use cases such as radiology education and multidisciplinary patient care. Our results indicate that the approach of loading image data locally and transmitting only viewer synchronization instructions over an RTCDataChannel provides a reduction in image latency relative to screensharing with conventional videoconferencing software, and it also eliminates image compression artifacts. These effects appear to have perceptual relevance as both trainee and faculty radiologists rated the RTC synchronization experience more favorably than CVS screensharing with regards to perceived image quality and latency, ease of setup, and overall experience.

Data transmission rates increased approximately linearly for the RTC synchronization method with the number of participants, as expected for a P2P architecture (Fig. 1), while that of the CVS we tested remained constant. Most of the linear increase in bandwidth requirements at the receivers is attributable to the always-active bidirectional audio

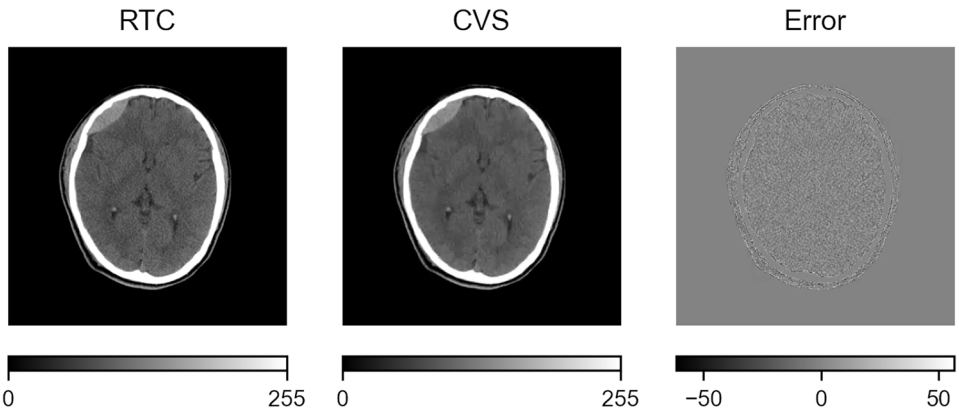

Fig. 4 Representative example of image compression artifacts typical of screen sharing with a CVS. The WebRTC synchronization scheme (left) shows the original image by design, while the data compression applied by the videoconferencing server degrades image quality, blurring subtle edges (center). The difference between the two techniques is more clearly visible in the subtraction image (right)

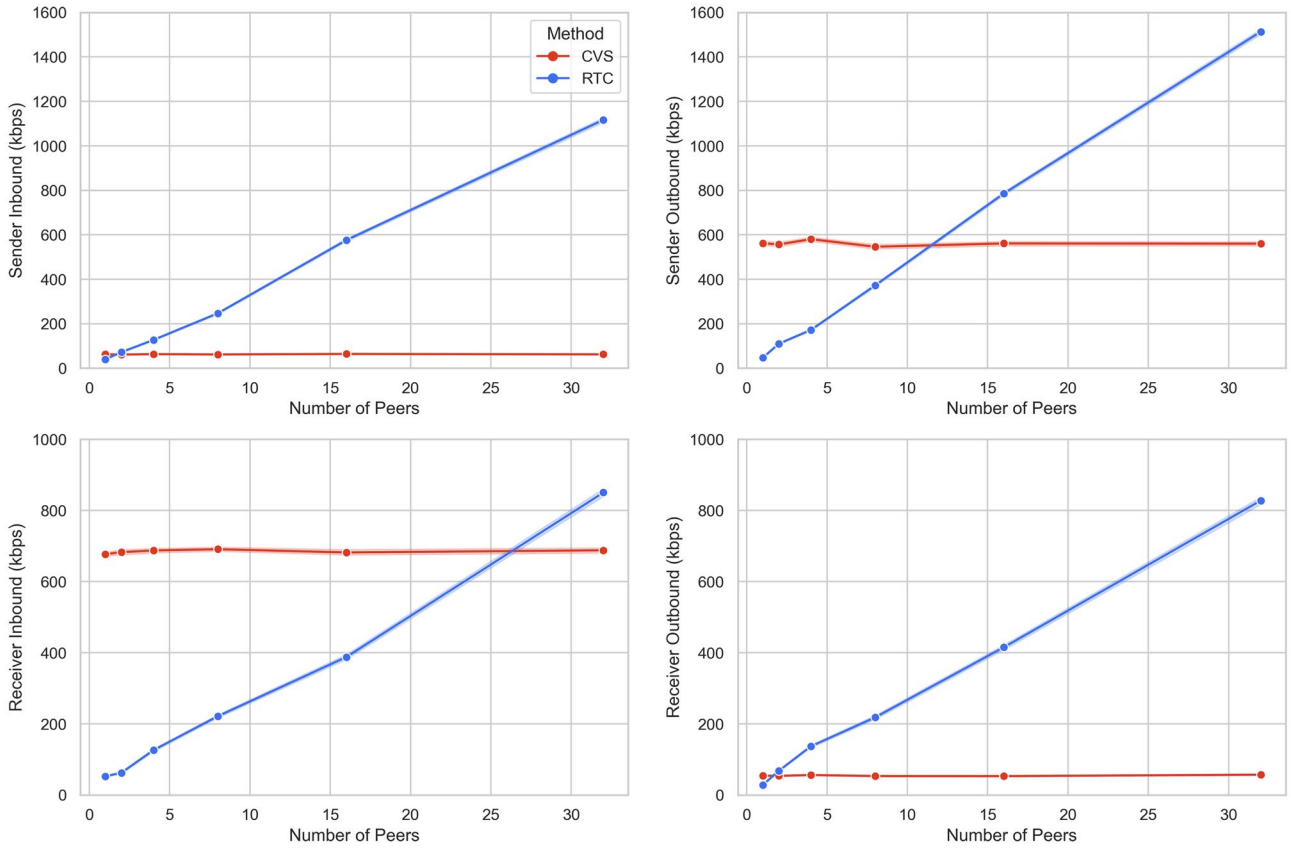

Fig. 5 Comparison of network resources required by the WebRTC and CVS image viewer synchronization methods. Bandwidth utilization is shown for the peer sending the image updates (top row) and a peer receiving the image updates (bottom row) as a function of the total number of meeting participants. Data rates are divided into inbound (left column) and outbound (right column) flows. Plotted points indicate means over 100 sampling windows and associated shaded regions span the standard error

stream established with each peer; on further examination of Firefox's internal RTC statistics (not shown), we discovered that the default Opus audio codec consumed approximately 26 kbps/peer in each direction, revealing that only about 22 kbps of bandwidth was devoted to receiving the viewer update stream [22]. The sender additionally bears the load of transmitting the continuous viewer updates to each peer and receives corresponding inbound traffic in the form of SCTP selective acknowledgements. Despite this linear dependence on the number of peers, for common use cases involving small groups (4 or fewer individuals), the bandwidth requirements of the RTC method remained less than that of the CVS we tested. As group size increased, the CVS approach provided lower data rates, albeit at the expense of increased latency and compression artifacts as shown. We note that even for groups including up to 32 viewers, the peak bandwidth required by the P2P architecture (approximately 1.1 Mbps down/1.5 Mbps up) falls well within the nominal limits of even previous generation local networking infrastructure such as 10BASE-T Ethernet and 802.11 g WiFi [23]. The requisite bandwidth also appears readily accessible over the WAN, given the Federal Communications Commission report that as of the end of 2018, 99.1% of the US population was covered by deployments of fixed terrestrial broadband with speeds of at least 25 Mbps downstream and 3 Mbps upstream or mobile long term evolution (LTE) connections with median speeds of 10 Mbps downstream and 3 Mbps upstream [24].

We emphasize that the approach presented here — communicating viewer state updates over a P2P network using a low-latency transmission protocol — can be implemented independently of WebRTC, which simply provides a convenient API for achieving this result in a web browser. Dedicated implementations could extend the minimal set of message types that we have presented to provide additional optimizations and functionality, such as synchronization of thick-client features like multiplanar reformation, intensity projection, and volume rendering. For the same reason, we have intentionally not addressed the mechanism by which the images or worklist are initially loaded at each peer. In a practical implementation, the communication of these viewer synchronization

instructions would ideally be included as a function of an enterprise PACS viewer itself, rather than in a third-party application. In that setting, the worklist and image prefetching can occur over the same virtual private network (VPN) or other secure tunneling solution that an enterprise has already established for remote image viewing by its thin clients. This arrangement also serves to internalize and thus protect the signaling server from outside threats.

Limitations of our work include the examination of a single commercial videoconferencing system as a comparison to the RTC approach; we studied the only HIPAA-compliant solution deployed at our institution. Different videoconferencing applications may have different latency and compression characteristics relative to that which we measured. Nevertheless, though the relative magnitudes may change with other videoconferencing applications, we expect the key findings of our work to hold since any CVS requiring data transit via a central server will experience additional latency related to this intermediate hop, and any system employing lossy compression to limit data rates will necessarily suffer from image degradation. We note that even those screen sharing applications with browser-based clients often employ a client–server architecture and are limited by the increased data rates required by the transmission of pixel data rather than viewer state. We have also assumed that all peers have access to an audio input device. While this assumption is likely to be satisfied for radiology workstations with speech recognition capabilities and most modern mobile computing devices, other workstations in physician offices and clinical areas, for example, may not have audio recording hardware. The ability to call into a conference over the public switched telephone network (PSTN) is offered by most CVS applications and is a particularly important feature to support for multidisciplinary conferences in which the majority of participants are often non-radiologists (and thus perhaps more likely to be connecting from workstations without an audio input device). Incorporating this interworking capability into the WebRTC scheme would require the additional overhead of running a private branch exchange (PBX) server.

Ultimately, the results presented here support the thesis that transmitting DICOM viewer update instructions over a low-latency connection enables a quantitatively and qualitatively superior collaborative image review experience for small groups, relative to conventional videoconferencing systems. This systematic analysis may be useful to imaging informatics professionals as they define vendor selection criteria for future PACS viewer procurements, given the increasing importance of remote conferencing capabilities in the modern era.

Acknowledgements The authors wish to acknowledge Michael Lee, MD; Ruskin Cua, MD; Ishita Desai, MD; and Sari Umekawa, MD, for their valuable feedback in testing the RTC technology.

Author Contribution VP and CL conceived the original idea. VP performed the data collection, processing, and statistical analysis, with support from CL and VR. VP and CL wrote the initial draft of the manuscript, with critical feedback and revisions by CL, AL, JA, and AR.

Statement of Data Access and Integrity The authors declare that they had full access to all of the data in this study, and the authors take complete responsibility for the integrity of the data and the accuracy of the data analysis.

Declarations

Conflict of Interest The authors declare no competing interests.

References

1. Li CH, Rajamohan AG, Acharya PT, Liu C-SJ, Patel V, Go JL, et al. Virtual Read-out: Radiology Education for the 21st century during the COVID-19 pandemic. Academic Radiology. 2020;27(6):872–881.
2. Matalon SA, Souza DAT, Gaviola GC, Silverman SG, Mayo-Smith WW, Lee LK. Trainee and Attending Perspectives on Remote Radiology Readouts in the Era of the COVID-19 Pandemic. Academic Radiology. 2020 Aug 1;27(8):1147–53.
3. Recht MP, Fefferman NR, Bittman ME, Dane B, Fritz J, Hoffmann JC, et al. Preserving Radiology Resident Education During the COVID-19 Pandemic: The Simulated Daily Readout. Academic Radiology. 2020 Aug 1;27(8):1154–61.
4. McRoy C, Patel L, Gaddam DS, Rothenberg S, Herring A, Hamm J, et al. Radiology Education in the Time of COVID-19: A Novel Distance Learning Workstation Experience for Residents. Academic Radiology. 2020 Oct 1;27(10):1467–74.
5. Kaplinsky C. TightVNC: VNC-Compatible Free Remote Control / Remote Desktop Software [Internet]. TightVNC Software. [cited 2020 Oct 13]. Available from: https://tightvnc.com/
6. X2Go [Internet]. X2Go Wiki. [cited 2020 Oct 13]. Available from: https://wiki.x2go.org/doku.php/start
7. Ludwig S, Beda J, Saint-Andre P, McQueen R, Egan S, Hildebrand J. XEP-0116: Jingle [Internet]. XMPP Specifications. XMPP Standards Foundation; 2018 [cited 2020 Oct 13]. Available from: https://xmpp.org/extensions/xep-0166.html
8. Glazer J, Madhav S. Multiplayer Game Programming: Architecting Networked Games. 1st Edition. Addison-Wesley Professional; 2015. 386 p.
9. Dresevic A, Mikel C. New HIPAA rules: a guide for radiology providers. Radiol Manage. 2013 Apr;35(2):34–9.
10. Jennings C, Boström H, Bruaroey J-I. WebRTC 1.0: Real-Time Communication Between Browsers [Internet]. World Wide Web Consortium (W3C). [cited 2020 Oct 14]. Available from: https://www.w3.org/TR/webrtc/
11. Wadali JS, Sood SP, Kaushish R, Syed-Abdul S, Khosla PK, Bhatia M. Evaluation of Free, Open-source, Web-based DICOM Viewers for the Indian National Telemedicine Service (eSanjeevani). J Digit Imaging [Internet] 2020 Jul 13 cited 2020 Oct 17 Available from: https://doi.org/10.1007/s10278-020-00368-4
12. Research Imaging Institute. Papaya [Internet]. Research Imaging Institute - Mango. [cited 2020 Sep 27]. Available from: http://mangoviewer.com/papaya.html
13. Handley M, Perkins C, Jacobson V. SDP: Session Description Protocol [Internet]. Internet Engineering Task Force. [cited 2020 Oct 17]. Available from: https://tools.ietf.org/html/rfc4566
14. Rosenberg J, Holmberg C. Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT)

Traversal [Internet]. Internet Engineering Task Force. [cited 2020 Oct 17]. Available from: https://tools.ietf.org/html/rfc8445
15. PeerJS - Simple peer-to-peer with WebRTC [Internet]. [cited 2020 Oct 17]. Available from: https://peerjs.com/
16. McGrew DA, Norrman K. The Secure Real-time Transport Protocol (SRTP) [Internet]. Internet Engineering Task Force. [cited 2020 Oct 17]. Available from: https://tools.ietf.org/html/rfc3711
17. Rescorla E, Modadugu N. Datagram Transport Layer Security Version 1.2 [Internet]. Internet Engineering Task Force. [cited 2020 Oct 17]. Available from: https://tools.ietf.org/html/rfc6347
18. Stewart R. Stream Control Transmission Protocol [Internet]. Internet Engineering Task Force. [cited 2020 Oct 17]. Available from: https://tools.ietf.org/html/rfc4960
19. Furuhashi S. MessagePack [Internet]. [cited 2020 Oct 16]. Available from: https://msgpack.org/
20. September 26, 1960: Debate with Richard Nixon in Chicago [Internet]. Miller Center. 2016 [cited 2020 Oct 14]. Available from: https://millercenter.org/the-presidency/presidential-speeches/september-26-1960-debate-richard-nixon-chicago
21. OpenCV Team. OpenCV [Internet]. [cited 2020 Oct 17]. Available from: https://opencv.org/
22. Terriberry T, Vos K. Definition of the Opus Audio Codec [Internet]. Internet Engineering Task Force. [cited 2020 Oct 17]. Available from: https://tools.ietf.org/html/rfc6716
23. Liu Y, Wang J. PACS and Digital Medicine: Essential Principles and Modern Practice. CRC Press; 2010. 370 p.
24. 2020 Broadband Deployment Report [Internet]. Federal Communications Commission. 2020 [cited 2020 Oct 14]. Available from: https://docs.fcc.gov/public/attachments/FCC-20-50A1.pdf