International Journal of Research in Signal Processing, Computing & Communication System Design
5 (1), 2019, 20-24
http://www.publishingindia.com/ijrsp

Audio and Video Streaming in Telepresence Application Using Web RTC for Healthcare System

Dhvani Kagathara[1] and Nikita Bhatt[2]

[1]Chandubhai. S. Patel Institute of Technology, Charotar University of Science and Technology, Nadiad, Gujarat, India. Email: kagatharadhvani@gmail.com

[2]Chandubhai. S. Patel Institute of Technology, Charotar University of Science and Technology, Nadiad, Gujarat, India.

Abstract: Currently Health care in LMICs (Lower & Middle-Income Countries) is suffering from shortage of trained Physicians in rural areas. The development of certain technologies, particularly in sensors has yielded the birth of mHealth. The main objective of this paper is to develop a Telepresence application for a mobile platform. The functionality of the system will include the ability for the doctor to connect with a doctor / patient from a remote location using mobile or web application and the ability to diagnose patient disease remotely and recommend follow up care. There is a requirement for low price, well suited, and easy-to-use video communication system. The Web Real-Time Communication (WebRTC) permits browsers to set up a peer connection to deliver information and media with real-time conferencing capabilities through simple JavaScript APIs. The design considerations for this paper want to be taken into consideration are high latency and low bandwidth environment and ability to share assets across the connection. For streaming of video in Telepresence application need to have a good peer to peer transport execution with great nature of service provisioning in network.

Keywords: Audio, Bandwidth estimation, Streaming, Telepresence, Video, Web Real-Time Communication (WebRTC).

I. Introduction

The Improvement of financial and social elements regarding a country is said to be complementary to each other. The fields including education or healthcare are suffering from the social need which sooner or later depicts the financial growth and in the long run the excellent of lifestyles. India has expected that with improved digital adoption, the Indian Healthcare marketplace, that is certainly worth around US$ a hundred billion, will probably develop at a CAGR of 23 percent to US$ 280 billion through means of 2020 [9]. Including infrastructure or medical specialists via myself will now not be capable about unravel India's considerable unmet desires among healthcare. It desires to be supported through the era. Scientific technology innovation may remain the tool to redact current care accessible, available then less costly to all by way on lowering the cost of the product or delivery.

The characteristic of playing back audio and video also as the record is Wight received is recognized as streaming that is a point in accordance with point and multipoint. Unlike everyday data file, a streaming media file for consideration is big, as a consequence calls because of excessive channel bandwidth. Furthermore, streaming media moreover contains severe demand inside the timing of bundle delivery. It has some constraints consisting of bandwidth, latency or congestion. Video streaming may stay categorized into two categories: live and on-demand. In this paper, we observe nearly used streaming protocols for mobile utility or web [2]. They may additionally be lower layer as TCP or UDP and upper layer so RTP, RTCP or RTSP [3]. In video streaming lower layer protocol such as TCP (Transmission Control Protocol) or UDP (User Datagram Protocol) are not feasible to use [3]. So, for that we will use the higher layer protocols such as RTP (Real-Time Transport Protocol), RTCP (Real Time Control Protocol) and RTSP (Real Time Streaming Protocol) [3]. WebRTC (Web Real-Time Communication), a project maintained by way of means over Google among 2011 primarily based on RTP is analyzed or related according to such through IETF and its browses API by the usage of W3C [7]. This gadget allows ongoing correspondences by means of a couple of use Programming Interfaces (API) alongside a top notch, low idleness and low transmission capacity utilization [2].

II. WebRTC

WebRTC is the collection of protocols, standards, and APIs [2]. Audio, video and information shares between peers using the connection [2]. There is no software or any extra plug-ins require [5]. The WebRTC implementation created through Google and open source launched it [2]. World Wide Web Consortium (W3C) organization is standardized the WebRTC at the application level and Internet Engineering Task Force (IETF) organization standardized the WebRTC at the protocol level [7].

The major components of WebRTC are below:

A. Media Stream

Get User Media allows to acquiring audio or video from the hardware such as Camera, Microphone, screen capturing or display [6]. Using Get User Media API we can get the screenshots and share the screen with different party also [6].

Table I: Media Engine Performance [4]

Module	Measured Statistics
Video Capture	Captured video resolution, and a number of frames captured per second [4].
Video Encoding	Encoded stream bitrate, frame rate and resolution also encoder ramp-up speed [4].
Video Decoding/Rendering	Frame numbers decoded / rendered per second; resolution of the rendered stream [4].
Networking	Bytes / packets sent, received and dropped, RTT [4].
Jitter Buffer	Jitter, A/V Sync [4].

B. RTCPeerConnection

RTCPeerConnection use for setting the audio / video streams [2]. It consists a lot of special tasks like codec execution, signal processing, bandwidth administration, safety streaming and many more. which represents the contain into MediaStream and DataChannel by presenting a handshake mechanism because of pair machines according to exchange fundamental data so a Peer-to-Peer connection may stay set up [6].

Table II: Connection Flow Measurements [4]

	Parameters	Measurement Statistics
Initialization	New Peer Connection	Time for instantiating a new RTCPeerConnection object [4].
	Get User Media	Time spent capturing user media excluding time waiting for user permission [4].
	Can Play Local Media	Time spent after media request until local media becomes available [4].
	Setup Signaling Channel	Time to setup signaling channel using Web-Sockets [4].
Communication	Make Call	Time for setting session descriptions and creating an offer [4].
	ICE Hole Punching	Time spent on ICE hole punching process [4].
	Make Answer	Time for setting session descriptions and creating an answer [4].
	Signaling	Time spent propagation of answer / offer messages and signals across the network [4].
	Open Data Channel	Time for the opening data channel [4].
	Play Remote Stream	Time for play remote media stream to be available [4].

C. RTCDataChannel

Connected Users can share audio, video, and data using RTCDataChannel [2]. Which is used for Bidirectional communication among peers and exchange the data which is in any format [2]. Which is share arbitrary data between peers [6].

Table III: Data Channel Measurements [4]

Category	Measured Statistics
Network	Bytes Sent / Received / Rtt [4]
Application	Messages Sent / Received / Rtt [4]

D. GeoStats

API call that hat permits getting diverse insights about a WebRTC session [2].

III. WebRTC Architecture

Fig. 1 illustrates the WebRTC overall structure with API used and different audio and video codecs used in streaming application. Also shows the features supported by browsers and application for real-time communication [1]. WebRTC offers net software builders the potential to put in writing wealthy, Real time multimedia applications (suppose video chat) at the web, without requiring plugins, downloads or installs. Its cause is to help construct a robust RTC platform that works throughout more than one web browsers, throughout a couple of platforms [8].

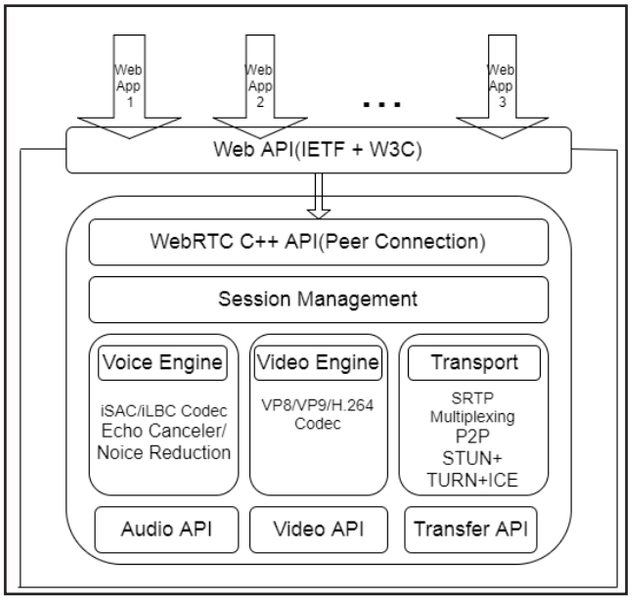

Fig. 1: WebRTC Structure

A. Voice Engine

The Voice Engine is usually called sound engine [1]. They are processed for echo cancellation and noise reduction [10]. The sound machines facilitate the Audio codecs which is optimized by narrowband and wideband [10]. The concurred sound codecs up to desire require in congruity with be upheld by method for WebRTC all around coordinated programs comprise of: iLBC (i.e., a narrowband discourse codec for VoIP or gushing sound), iSAC (i.e., a wideband voice codec in light of the fact that VoIP at that point spilling), or Opus (i.e., a codec so much aides bitrate encoding) [1]. In sender side they are process the raw streams to enhance the quality and match the bandwidth and latency which is continuously fluctuating [10]. From receiver side received streams decode and adjust the latency delay and network jitter [10].

B. Video Engine

The Video Engine is a framework as offers together with video streams who come beside the digital camera in conformity with the community and from the community in accordance with the show display [1]. Just as the frame engine, the video engine helps the video codecs. VP8 is writing in time agreed with the video codec [1]. It helps potential jitter buffer because of the video who allows in accordance with hide the outcomes concerning get rid of jitter then packet loss in imitation of enhance the overall video top notch [1]. It additionally allows image enhancement, as much an instance, eliminating video noise from the image capture by using the usage of behavior regarding the webcam [1]. It is designed for low latency so that well suited for RTC [10].

C. Transport Support

At the transport layer pass the audio and video streams to the peers using this activity [1]. TCP buffers all packets after intermediate packet is lost and wait for retransmission after deliver the stream [10]. A Secure Real Time Protocol is used with WebRTC to secure the streams of audio or video using encrypt all the records [1]. UDP delivers each packet at the moment it arrives which is no promises on ordered data [10].

D. Session Management

The session management defines as beneficial to WebRTC as abstract signaling approach and the protocol according to session data as explained or leaves the statistics about the signaling implementation on the hand concerning application builders [1]. Session description protocols are most affected by WebRTC [1]. C++ native API of WebRTC at this API layer allows browser markers in accordance with except issues put into effect the web API because of his or her browses [1]. Firewalls deals the parameters for every stream and implement congestion control, flow control and encryption of data and more [10].

IV. Implementation

A. Environment Setup [1]

We are using the WebRTC Functionalities for establishing the environment, which is established previously by EasyRTC APIs and JavaScript APIs for Web Browsers. EasyRTC is the open source project [1]. We used the node.js server and JavaScript APIs for the run the application in Web Browser and for the mobile application, we used the WebView [1].

B. Audio and Video Streams Gathering [1]

For Audio and Video Streams gathering found complexity [1]. Call EasyRTC. getLocalStreame and EasyRTC [1]. set video ObjectSrc [1]. We have to invoke the method for ask the permission for Local camera and microphone [1]. After giving the camera and microphone permission user can see their local screen. For incoming streams, we have to call the callback method [1]. For setting the bandwidth for each video track we can use EasyRTC setVideoBandwidth() [1].

C. Signaling and Peer Connection [1]

To establish the communication first exchange the data between peers [1] As an example, for making the connection in the same network only public IP address is necessary [1]. But the process is complicated when peers were far to each other and peer's public IP addresses and port Address are hidden to each other [1]. If one of the peers is not available we have to first acquire the viable IP addresses and port candidates for each peer, traverse the NATs, and then run the connectivity [1]. The single interface encapsulating all the connection setup, control, and state PeerConnection API within the WebRTC is dealing with the life cycle for every peer connection [1].

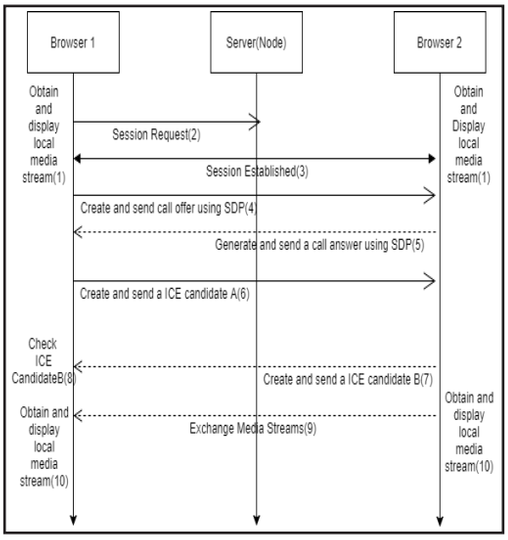

Fig. 2: Peer to Peer Interaction

Fig. 2 Explains the signaling interactions and the requirements between two peers.

(a) Whenever the user from one side which uses browser 1 clicks the "join" button, the browser of user obtains and presentation the local media [1].

(b) After a user of one side wants to connect to the user of another side which uses browser 2 for video conference of the secure channel using the signal server [1].

(c) After the session is mounted between both users [1].

(d) For exchange, the media, codecs and setting their bandwidth information, a user from one side uses the Session Description Protocol (SDP) [1]. Real media itself is not connected via offer [1].

(e) As soon as the offer is gotten, a client from opposite side makes the offer to associate with the client in one side and furthermore sends the comparing session profile utilizing the SDP [1]. After that client from one side realizes that client from the opposite side is prepared to associate utilizing shared connection [1].

(f) User from opposite side makes an intuitive network foundation (ICE) operator, (ICE A) to the client from program 1 side [1]. The ICE specialist finds the closer IP with the port tuple of companion and uses outer STUN server for get general society IP and port tuple [1].

(g) The client from program 2 side accepting the ICE an indistinguishable activity from a client from program 1 side to make and send ICE specialist (ICE B) [1].

(h) TURN server utilized as an intermediary to hand-off movement if clients can't make peer association utilizing STUN server [1]. It will get data ahead of time which is suits in ICE B [1].

(i) User from one side sends the video stream to another side user [1]. Both side data and nearby media streams established [1]. In the wake of getting general society IP locations and port assortment tuples ahead of time from ICE A [1].

(j) In the end, Media content is shown by both the browsers and the video conferencing is operating [1].

STUN / TURN servers can be used for handle a small number of participants. Javascript code added to the make the peer connection [1].

D. Chatting Room Establishment [1]

This is used for the Patients and doctors to conduct the meetings [1]. Codecs for client and server needs to be implemented. Using Socket.io first client getting the chat channel [1]. As the connection established client can send the text. After every user joins the room. Display the incoming messages and outgoing messages [1].

V. Limitations

Limitations of Existing Telepresence Applications: Present video streaming application used for Telepresence criticized for plenty reasons: a) they're often too high priced to buy and preserve, b) Using Technologies which is proprietary cannot be compatible with each other, and c) To keep the device they require fairly skilled IT personnel [1].

VI. Conclusion and Future Extension

In this paper, we have discussed the WebRTC standard based application for Telehealth which is simple, interoperable and inexpensive for video conferencing. The proposed healthcare system will be helpful for communication of doctors and patients. The system would be developed using web Real time communication with better quality audio and video streams using different protocols and standards to evaluate the need for Telehealth. Real time communication use for all the platforms of a web on cloud computing and Android / iOS devices. We have discussed related implementation of a WebRTC based system which is emerging different protocols and architecture. Which is provide the good user experience in web and mobile both. We have used different mechanisms like RTP based media exchange and peer to peer data exchange. WebRTC provides more developer friendly environment, that does not require specific software and plug-ins and it will be available at anytime and anywhere. WebRTC requires permission to use camera and microphone to access from the user. WebRTC reduces energy intake and bandwidth utilization and accordingly enhances the effectiveness of the system. We have used different approaches for real time communication but methods have some limitations, which may be overcome by improving an existing system. In future Healthcare device can handle multiple connections with low bandwidth and high latency. Audio and video must of good quality.

References

[1] J. Jang-Jaccard, S. Nepal, B. Celler, and B. Yan, "WebRTC-based video conferencing service for telehealth," CSIRO Computational Informatics (CCI), Springer, Marsfield, Australia, 2014.

[2] I. Santos-González, A. Rivero-García, T. González-Barroso, J. Molina-Gil, and P. Caballero-Gil, "Real-time streaming: A comparative study between RTSP and WebRTC," in C. García, P. Caballero-Gil, M. Burmester, and A. Quesada-Arencibia, (eds.) *Ubiquitous Computing and Ambient Intelligence*. IWAAL 2016, AmIHEALTH 2016, UCAmI 2016. Lecture Notes in Computer Science, vol. 10070. Springer, Cham, pp. 313-325, Springer, 2016.

[3] T. Arsan, "Review of bandwidth estimation tools and application to bandwidth adaptive video streaming," 2013.

[4] S. Taheri, L. A. Beni, A. V. Veidenbaum, A. Nicolau, ..., and M. R. Haghighat, "WebRTCbench: A benchmark for performance assessment of WebRTC implementations," *2015 13th IEEE Symposium on Embedded Systems For Real-time Multimedia (ESTIMedia)*, IEEE, pp. 1-7, 2015.

[5] B. A. Jansen, "Performance analysis of WebRTC-based video conferencing," Springer, 2016.

[6] L. Lucas, H. Deleau, B. Battin, and J. Lehuraux, "USE together, a WebRTC-based solution for multi-user presence desktop," *14th International Conference on Cooperative Design, Visualization and Engineering (CDVE 2017)*, Majorque, Spain. 2017.

[7] https://en.wikipedia.org/wiki/WebRTC

[8] https://webrtc.org/architecture/

[9] https://www.ibef.org/industry/healthcare-india.aspx

[10] https://princiya777.wordpress.com/2017/08/19/webrtc-architecture-protocols/