Network Support for Social 3-D Immersive Tele-Presence with Highly Realistic Natural and Synthetic Avatar Users

Rufael Mekuria
Centrum Wiskunde & Informatica
Sciencepark 123
1098 XG Amsterdam, NL
+31(0)20 592 4020
Rufael.mekuria@cwi.nl

Antonella Frisiello, Marco Pasin
Istituto Superiore Mario Boella
Via P.C. Boggio 61
10138 Turin, Italy
0039 011 2276201
frisiello@ismb.it, pasin@ismb.it

Pablo Cesar
Centrum Wiskunde & Informatica
Sciencepark 123
1098 XG Amsterdam, NL
+31 (0)20 592 4332
P.S.Cesar@cwi.nl

ABSTRACT

The next generation in 3D tele-presence is based on modular systems that combine live captured object based 3D video and synthetically authored 3D graphics content. This paper presents the design, implementation and evaluation of a network solution for multi-party real-time communication of these types of content. This prototype includes a UDP/TCP multi-streaming kernel that includes media synchronization support, packet scheduling, loss resilient real-time transmission and an easy to use blocking and non-blocking API. To compress the live reconstructed 3D data streams that represent the natural user, two categories of 3D mesh codecs were integrated: a highly adaptive real-time geometry driven mesh codec and a fast single rate codec that provides better performance at high resolutions. Subjective tests with 16 subjects indicate that only modest perceptual degradation of the highly realistic 3D natural user is introduced, especially when the users in the virtual world are at a distance. We developed a session management protocol for setting up streams based on the specific 3DTI capabilities allowing device scalability from light (render only) to heavy clients (rendering and 3D Capturing). Additionally, a distributed messaging system via web-sockets and cloud infrastructures based on publish and subscribe was integrated for real-time delivery of avatar and other AI data.

Categories and Subject Descriptors

H.4.3 [**Information Systems Applications**]: 3D Tele-Immersion, Media Synchronization, Mesh Compression, Tele-Presence, Virtual Worlds, Media Streaming

General Terms

Performance, Design, Experimentation, Human Factors

Keywords

3D Tele-presence, 3D Mesh Compression, 3D Humans

1. INTRODUCTION

http://dx.doi.org/10.1145/2723695.2728605

Recent work has shown that it is possible to obtain photo-realistic full 3D animated mesh objects in real-time with inexpensive consumer grade depth cameras (such as Microsoft's Kinect [1] [2]). In such a case, the representation is a segmented 3D object (i.e. 3D mesh or 3D point cloud) defined in the 3D space that can be compositely rendered in the virtual world with other synthetic assets or specific 3D rendering modules. In Figure 1 we show an example of a live reconstructed mesh rendered in a 3D scene with synthetic 3D avatars. This mesh was reconstructed in real-time from streams of the first generation of Microsoft's Kinect based on the work in [2]. To obtain a real-time distributed shared experience that combines such natural and synthetic contents, the development of appropriate framework for real-time data transmission and compression is a key challenge.

Figure 1 A natural user rendered in a virtual scene with synthetic users

The **contributions** of this paper are the implementation and evaluation of a complete framework to support networked multi-site 3D tele-presence with highly realistic natural users generated from depth cameras with 3-D reconstruction techniques combining synthetic avatar based users. We introduce the multi-purpose UDP/TCP **streaming kernel** to address the streaming needs. This includes error resilient real-time transmission and support for basic **media synchronization** between heterogeneous streams. We briefly discuss the implemented **session management** protocol for signaling custom data types that can be deployed on top of the XMPP presence protocol. For natural users, this framework integrates novel **3D Mesh compression**. We included a subjective study of the perceived quality of the decoded and original natural human in the 3D virtual room with 16 subjects and two different contexts (i.e. near camera, far camera) to allow subjective optimization of the transmission and compression settings. For avatar based users and messaging data, this framework integrates and evaluates different **real-time messaging** solutions based on available publish and subscribe protocols. Lastly, we integrated this solution in a larger 3D tele-immersive test bed that includes 3D audio, avatar users, natural users and rendering in a virtual 3D room. We evaluate the overall **streaming performance** in a 3-way scenario with heterogeneous sites.

The structure of the paper is as follows, in section 2 we present the structure of the envisioned 3D tele-immersive framework, the real-time data streaming framework (data plane) and the session management protocol (control plane). In section 3 we present the integration of 3D mesh compression in the framework and their evaluation based on subjective testing. In Section 4 we further evaluate the networked performance for the avatar and natural user transmission in a realistic 3-way scenario.

MMVE'15, March 18-20, 2015, Portland, OR, USA

http://dx.doi.org/10.1145/2723695.2728605

2. SYSTEM ARCHITECTURE AND IMPLEMENTATION

In this section we detail the design and implementation of our system. The full framework architecture is presented in section 2.1. The real-time TCP/UDP streaming engine is presented in section 2.2. Last, the session management scheme is discussed in section 2.3. Together they provide the networked infrastructure for social immersive 3D communication with natural and synthetic users.

2.1 Modular 3D Immersive Communication Framework Architecture

Figure 2 shows a simple outline of the modular 3D tele-immersive system architecture. Any user runs a main application that loads modules for specific render and capture capabilities based on its local configuration (which are stored in an xml format for convenience). For example, there exists a module for natural 3D reconstruction via depth cameras, one for animating pre-recorded data via skeleton tracking, for real-time complex human shape pose modeling, 3D audio rendering and capturing, 3D computer controlled avatar characters and for navigation and representation of the world. For example application A loads natural user data (module A), audio capture (Capture B) and has a module to render incoming natural user streams (i.e. Render A). The information about the loaded modules is shared at login to the session management system which provides authentication and user login and setup of streams. When user B, that can only render natural users (it is a passive user), logs in it can request the natural user stream from user A, but user A will not request any such streams from user B. We deploy this modular architecture to allow terminal scalability from passive, very light users to users with powerful capture and render capabilities. To allow stream setup between corresponding modules, we keep track of module and payload types and their correspondence in a centrally administered table specific to the framework.

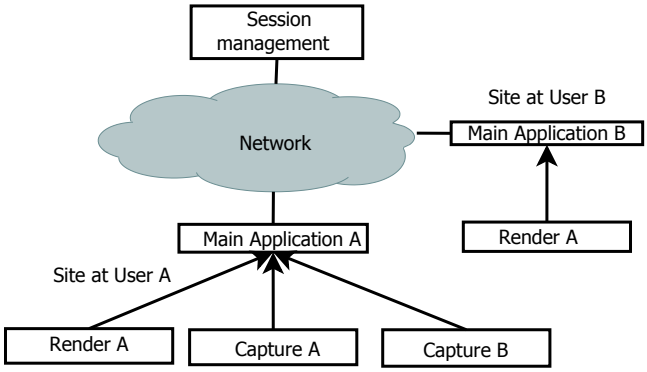

Figure 2 Modular 3D immersive Architecture with Terminal Scalability

2.2 Real-Time Streaming Framework

The streaming framework for real-time data is illustrated in Figure 3. We highlight that it is needed to develop such specific streaming framework based on custom transport protocols, as existing standard protocols and popular media streaming frameworks do not support the data types that we aim to transmit. They work with registered standardized data types as registered by internet assigned naming authority (IANA) (e.g. mostly audio and video defined on a rectangular grid [3]). For our needs, the framework in Figure 3 can handle different types of incoming and outgoing media streams (UDP/TCP) representing 3D data and real-time messaging. Critical for real-time communication of natural users are pure TCP and UDP with real-time FEC as described in [4]. For command and avatar messaging we use the Publish and Subscribe paradigm based on web-socket, UDP and XMPP. The media synchronization module operates independently of the incoming/outgoing streams and their specific implementation. It provides media synchronization services to the modules by keeping track of end-to-end latencies of the relevant streams. This allows modules to request target inter-sender and target inter-media skews and do synchronization based on a local play-out buffer on a best effort basis. To facilitate this, the streaming framework includes a virtual clock for global time synchronization based on a PTP like synchronization protocol and a function that allows receiver modules to report stream processing latency (i.e. decoding time).

To control admission of incoming and outgoing streams, the allowed streams table is managed by the session management protocol discussed in the next section. The monitor component keeps track of all the processing and network latencies that streams experience (i.e. compression, FEC, rendering, capturing, network delay etc.) which allows the system to detect anomalies and problems in the pipeline. The UDP Src and Sink Components handle the socket based network communication. For UDP streams based on [4], per packet decoding (progressive decoding) of the incoming packets needs to be performed, to achieve this we implemented an efficient multi-threaded process in UDPSrc to simultaneously receive and decode packets. The API to the 3D tele-immersive framework (pull/push frame API) includes both blocking and non-blocking methods for sending and receiving data, serving the specific needs of the module. The instance of the streaming framework resides in the main application, and can be used by the different render/capture modules. This allows modules that are otherwise unaware of each other to perform the synchronization of media streams and share a network service. The streams all follow an RTP like format where the payload_type specifies the specific administered payload type for the 3D module. The sequence number is only used by UDP and signals the ordering of the packet in the overall frame. The ssrc is a unique randomly generated identifier for the stream; frame_id is the frame count number, timestamp the globally synchronized capture time of the frame (or an approximation of this). Source Id uniquely determines the sending host, The NC_header_size signals the existence of a forward error correction that contains additional information for packet FEC decoding. Session Id and Routing are reserved for distinguishing session and overlay routing.

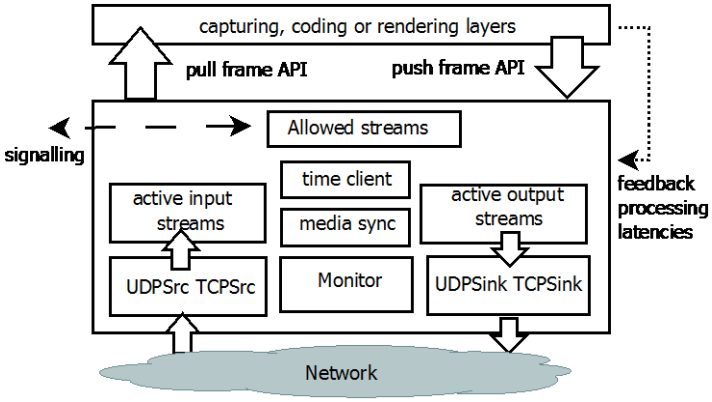

Figure 3 Data Streaming Framework Implementation

2.3 Basic Session Management Protocol and Implementation

We chose the open source XMPP presence protocol for user presence and signaling. We chose the XMPP 0030 Device Discovery extension to exchange capabilities of the 3D tele-immersive network entity (i.e. configuration as described in section 2.1.). Clients that log in provide their capture and render modules as discovery items which are triplets of the form <name,type,category>, where we use the category field to signal the module name and the type field for possible additional parameters. Additionally, we defined XMPP messages MEDIA_STREAMS, for requesting an overview of the available media streams, REQUEST_STREAMS to request a specific stream from a 3DTI terminal and ACK_REQUEST_STREAMS to acknowledge or decline such requests. In Figure 4 we show the sequence diagram with two users. First TLS/SASL based authentication happens. Once this is successful XMPP messages can be exchanged. The user's module configuration is exchanged to the server via discovery items. When another user logs in, XMPP signals presence and the server updates stream information in the MEDIA_STREAMS. This message lists the available 3D streams, (aggregated from the discovery information provided at login times by each user). In this case the second client requests the stream from client 1 first (this updates the allowed streams table in Figure 3). When client 1 receives this request it checks if it can send the stream, and sends an ACK_REQUEST_MESSAGE which indicates if the stream request was acknowledged or not. If it is, the media stream is transmitted via UDP or optionally TCP.

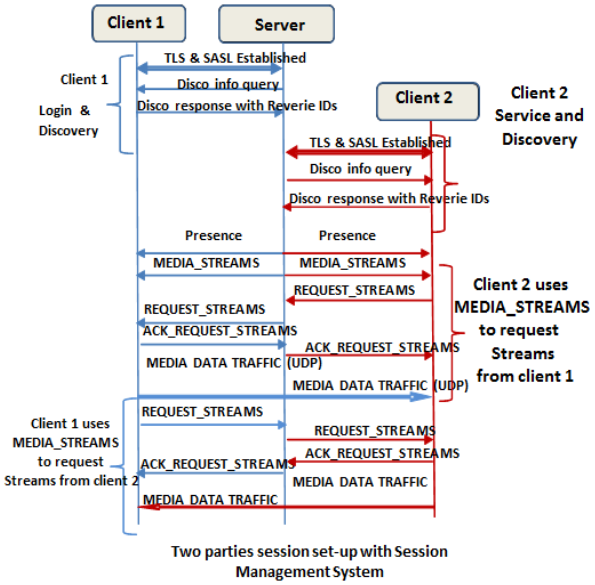

Figure 4 3DTI Session Management based on XMPP

Table 1 Common Header for TCP/UDP based streams

payload_type	Sequence number
ssrc	Frame_id
Time_stamp	
Packet count	Routing id
Source id	Session id
Nc_header_size	xxxxxxxxxxxxxx

3. NATURAL USER COMPRESSION

The 3D Mesh data representing the natural human is large (~15MB) and requires lossy compression, in the next section we describe the integrated codecs and in section 3.2. we do a subjective evaluation of the resulting decoded quality.

3.1 Natural User 3D Mesh Compression Objective Results

To reduce the size of the natural user that is reconstructed as a mesh from kinect streams, we integrated two lossy real-time 3D mesh codecs: imtpc from [5] and geometry driven from [6]. The original and mesh decoded with the first codec are shown in Figure 5 which achieves compression ratio of 12-14 x without vertex decimation. The decoded results with [6] are shown in Figure 6 with a setting (0.008 used throughout the paper) that gave 115 x compression rate and decimated around 9 in 10 of the vertices in the original mesh. The 5 Kinect Scans are reconstructed with the same system as in [7].

Figure 5 Original reconstructed natural user (left) and the imtpc decoded version with [5] (right) (upper body only)

Figure 6 Decoded mesh based on [6] with and without enhancement layers (geometry driven 0.008)

3.2 Subjective Results

To evaluate further the compression performance, we look at the human perception. It is commonly known that the objective assessment methods of video quality might not reflect the human judgment. For that reason subjective assessments have been carried in related works such as [8] , [9] and [10]. How do the used compression methods influence the end-user perception? Which aspects are more affected by the different compression methods? Does the best compression suit both for the network performance and the user experience? As we used reconstructions based on Kinect 1st generation, what is the quality compared to the original reconstructed mesh?

A laboratory test has been performed according the procedures suggested in ITU Standards [11], [12]. 16 volunteer representative users have been recruited. The core criterion for inclusion in the experiment was the attitude toward technology, assessed according the TAM [13]. The participant sample population was

gender balanced, the age range from 25 to 38 years old (mean 30,8), with a high educational level (degree, master or PhD), mainly in engineering branches. The users provided their informed consent to participate to the experiment when invited.

A within test has been designed: in order to estimate how individual behaviour changes when the stimuli and variables change, each subject has been asked to assess all the configurations to be tested [14]. Two iterations have been performed with different subjects. The stimuli to evaluate were a sequence of randomized videos, presenting the same subject, a 3D human moving at 8 fps in a virtual environment created with the full 3DTI framework that includes a high quality 3D rendering and degradation introduced by the compression method. The independent variables observed were:

- The compression methods: original, imtpc [5], geometry driven [6] (parameter 0.008, basic and enhanced versions)
- The virtual distance of the 3D replicant (far vs close to the virtual camera in the 3D space)

The subjects have been welcomed in the laboratory, and before starting the test received the introduction to the experimental goals, highlighting the focus on the 3D human and the context of the study. The possibility to interrupt the test if needed was provided. Before evaluating the videos, each subject had the possibility to interact with the real system to become more familiar with the context of realistic users in a 3D virtual room. Then, they were asked to watch pre-recorded stimuli videos one at a time (in a randomized order). After each video the subjects expressed an evaluation of the perceived video quality of the 3D reconstructed human. Additional rating dimension of the perceived realism into the scene (explained as the integration of the 3D human render with the virtual environment) was given.

Each dimension has been evaluated by using the MOS – Mean Opinion Score [11], allowing to collect the average of the opinions ("votes") on the single given conditions. The rating scale (on 5 points: Excellent 5, Good 4, Fair 3, Poor 2 , Bad 1) was printed and left on the table as reference for helping the subject to reply. The test protocol applied allows collection of the subjective evaluations on the single video just after it is watched, on the basis of the immediate impression. For each test variable, the mean value and the standard deviation of the statistical distribution of the assessment grades has been calculated.

Table 2 shows modest differences between original and decoded meshes. Pairwise comparison of each codec setting with the original mesh via paired samples t-tests, showed that only for the codec with voxel simplification (0.008) [6] the mean difference was significant ($p<0.05$). Next, we investigate the effect of nearby and far away camera views in the 3D world shown in Table 3 and Figure 7. For geometry driven both near and far were tested significantly lower compared to the original mesh ($p<0.05$). On the other hand, differences of the original with imtpc were not found significant.

Table 2 – MOS on the perceived quality

	MOS and Compression method							
	Original [2]		**IMTPC** [5]		**VoxBasic** [6]		**Vox enhanced** [6]	
	mean	st.dev	mean	st.dev	mean	st.dev	mean	st.dev
quality	**3,2**	0,9	3,0	1,0	2,7	0,9	2,9	1,0
realism	**3,3**	0,8	2,9	0,9	3,0	0,9	2,9	1,0

Table 3 MOS at different virtual distances

	MOS and Virtual distance			
	Near		**Far**	
	mean	st.dev	mean	st.dev
quality	2,8	0,9	3,1	1,0
realism	2,9	0,9	3,2	0,9

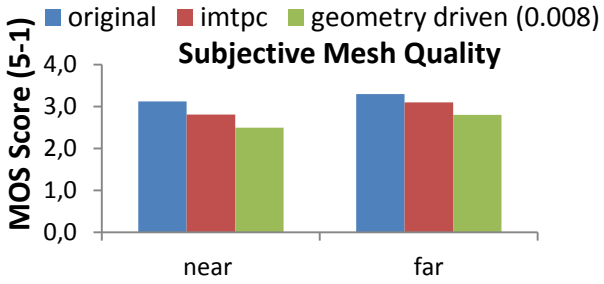

Figure 7 Subjective results evaluation natural user quality (original, imtpc, voxel based (geometry driven))

4. FRAMEWORK EVALUATION

4.1 AI and Avatar Commands messaging

We experimented with 3 implementations of publish and subscribe messaging for avatar animation and commands, based on XMPP XEP 0060 via open fire server, WebSockets (via a real-time cloud service) and native UDP. Our preference went to the web-socket implementation as we experienced less connectivity problems (i.e. firewalls, NAT), modest delay, no occasional losses as in UDP, and we could use the massive real-time distribution services offered by the real-time elastic cloud (real-time.co) for massive distribution.

We tested the performance with a test module sending data via the main application outgoing message queue over the web socket to the cloud. The message is then again received in another module on another host. We send both small messages (20 bytes) and large messages (500 bytes) at different frequencies from 10 Hz to 200 Hz. We utilize the clock synchronization service of the Real-Time Streaming module to measure the latency of the message dissemination. The module runs in realistic conditions were other network traffic and system processes are also running. All delays were tested below 50 ms on average upto 100Hz, only at 200 Hz end-to-end delay becomes over 100 ms upto uncontrollable. As we do not want this in our framework, we limit the outgoing message sending rate to a maximum of 100 messages per second which is sufficient to support the avatar and AI based user modules.

4.2 Natural User Transmission

For the multi-site transmission with 3 or more users, we prefer the geometry driven codec [6] that gives smaller frame sizes resulting in better frame rates. The experimental setup was deployed in a LAN network with 3 machines connected via a switch as shown in Figure 8. PC1 is a natural user reconstructed from 5 Kinects, PC2 is a natural user reconstructed from 1 Kinect and PC3 runs an avatar based synthetic user. The Machine and setup specifications are given in Table 4, users receive all streams. We use the network emulator for Windows [15] to emulate losses, delays (10-100 ms) and jitter (10-50 ms, 50 % of the delay each time) on incoming packets on each site. Additionally the incoming bandwidth is limited to 100 Mbps. The PC 3 avatar user site is a relatively modest laptop computer, while PC1 and PC2 are

stronger machines. We deploy both UDP based plus fec based on [4] and TCP. For TCP we apply rate control on the data by skipping late frames from the sender queue, (i.e. a last in first out policy towards the TCP socket, LIFO). For UDP transmission we fixed the sending frame rates. We measure delays from capture time to render time including codec delays.

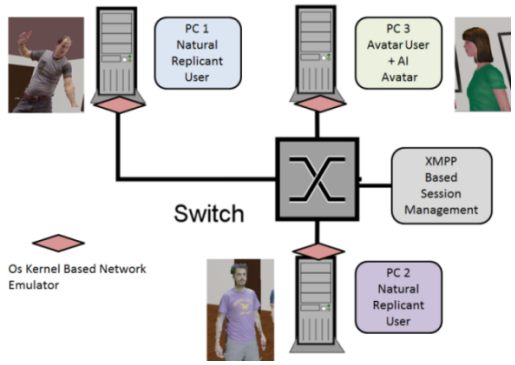

Figure 8 Experiment setup, 3 sites are connected via a switch and network impairments are introduced via a network emulator on incoming data at each site

Table 4 Machine and component specs

Component	Specs
PC 1	Desktop Intel i7 2,8 GhZ , 8 GB Ram, Win 7 64-bit, NVIDIA GeForce GTX 760
PC 2	Desktop Intel i7, 3,4 GhZ, 16 GB, Win 7 64 Bit, NVIDIA Geforce GTX 470
PC 3	Laptop Intel i7 , 1,6 GhZ, 4GB, Win7 64 Bit, AMD Radeon
Switch	NetGear GS 105 Gigabit Switch
Network Emulator	Network Emulator for Windows [15]

We have performed measurements across each site, but we only show the results for PC3 that is a modest laptop receiving both of the natural user streams (results were consistent across sites). The UDP based transmission limits the end-to-end delay to the computational and network latencies that could even be further reduced by implementation. On the other hand, transmission with TCP introduces delay in network conditions with loss and delay. In the current implementation both the 5 Kinect and 1 Kinect streams are received within 300 ms bounds for UDP as shown in Figure 9 and Figure 10. Also, the frame rates achieved for the received streams are shown in Figure 10. For UDP the achieved frame rates are stable in varying network conditions and at heterogeneous sites (contrary to TCP which was not shown)

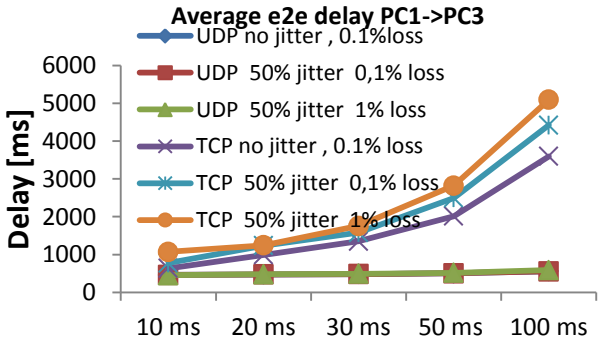

Figure 9 End to End delay of PC1 stream received at PC3

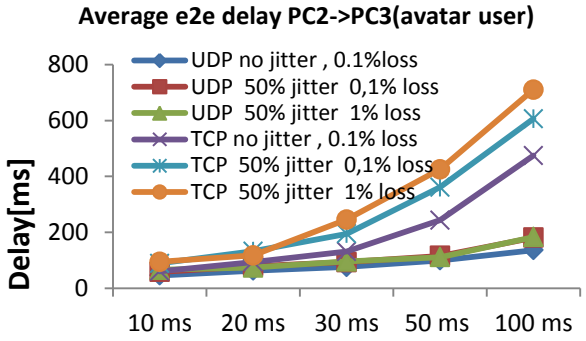

Figure 10 End-to-End delay of PC2 stream received at PC 3

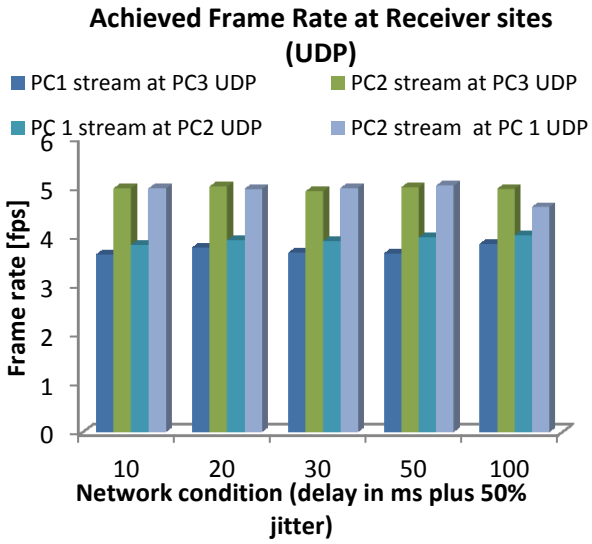

Figure 11 Achieved frame rates for each stream at each site

4.3 Frame Skew and Media Synchronization

Currently, support for media synchronization at the rendering tier is provided via play-out buffering. The basic support procedure is listed in Table 5. The streaming module keeps track of all the streams and their latency based on the synchronized timestamps from the virtual clock in the payload headers. As some incoming streams experience extra computational latencies before rendering (i.e mesh decoding), modules report back these times to the real-time streaming framework via a function (such that they can be accounted for). In step 3, modules can request the target sync latency, which is the extra time the module needs to wait before rendering frames to achieve the required sync with the other streams (as long as it is within real-time bounds 300 ms). Such waiting can be done using a play-out buffer implemented in the modules. We found the design of the play-out buffer for 3D audio trivial, as based on the audio sample rate and frame buffer-size, we could calculate the amount of frames needed to buffer and achieve synchronization as such. For buffering meshes with possibly varying rates, we made continuous running estimates of the instantaneous frame--rate and buffer K mesh frames based on these estimates. We implemented a play-out buffer in a 3D Audio module that captures and transmits monophonic 44.1 Khz 16 bits PCM samples over TCP such that it can synchronize with the mesh streams. In Table 6 we detail the achieved average media skews and deviations between audio and mesh data at PC1 and PC 2, and between incoming meshes from PC1 and PC2 at PC3. The

measurement is done with PC1 sending 5 Kinect meshes at 6 fps, PC2 Sending meshes with 8 fps, 25 ms delay, 10 ms jitter and 0.1% packet loss. We report both the mean skew and the standard deviation to better assess the acquired synchronization quality. The implemented play-out buffers reduces the skew to negligible proportions.

Table 5 Media Synchronization Support

1.	MediaSync in Real-Time Streaming Framework Tracks all network stream delays
2.	Modules provide their computational processing latency (decoder time etc..)
3.	Modules request target latency (either per sender)
4.	Modules based playout buffer provides mean synchronization

Table 6 Achieved Skews with Play-out Buffers

		Sync control Off		Sync control On	
skew type	Loc.	mean [ms]	Stddv	Mean [ms]	Stddv
PC2-Mesh – PC2 3D Audio	PC1	32	28	2	16
PC1 5-K-Mesh PC1 3D Audio	PC2	150	35	10	26
PC1 5-K-Mesh - PC2 1 K -Mesh	PC3	178	25	33	18

5. DISCUSSION AND CONCLUSION

We described the design, implementation and evaluation of a practical network support system for modular social 3D tele-immersive interaction with natural and synthetic users. The most challenging aspect was the multi-site real-time transmission and compression of natural user data represented as 3D Meshes. We deployed subjective evaluations, as there are no good objective ways to measure the quality of this type of content and objective network experiments. The session management system was deployed on top of XMPP, and user credentials from a social network portal can be easily imported, linking 3D immersive communication to the social network. In our future work we will focus on improvements for mesh compression and more efficient multi-site transmission strategies.

ACKNOWLEDGEMENTS

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement no. ICT-2011-7-287723 (REVERIE project). We thank all the partners in REVERIE project for their useful contributions to the project. We thank prof. Klara Nahrstedt from the University of Illinois for her very useful advice and valuable discussions.

REFERENCES

[1] C. Zhang, Q. Cai, P. A. Chou, Z. Zhang, R. Martin-Brualla., "Viewport: A Distributed, Immersive Teleconferencing System with Infrared Dot Pattern," *IEEE Multimedia Magazine*, vol. 20, no. 1, pp. 17-27, Jan-March 2013.

[2] D. Alexiadis, D. Zarpalas, and P. Daras, "Real-Time, full 3-D reconstruction of moving foreground objects from multiple consumer depth cameras," *IEEE Transactions on Multimedia*, vol. 15, pp. 339-358, 2013.

[3] IANA. Session Description Protocol (SDP) Parameters. [Online]. http://www.iana.org/assignments/sdp-parameters/sdp-parameters.xhtml

[4] R. Mekuria, M. Sanna, E. Izquierdo, D. Bulterman, and P. Cesar, "Enabling 3D Tele-Immersion with Live Reconstructed Mesh Geometry with Fast Mesh Compression and Linear Rateless Coding," *Multimedia, IEEE Transactions on* , vol. PP, no. 99, 2014.

[5] R. Mekuria , P. Cesar , D. Bulterman., "Low complexity connectivity driven dynamic geometry compression for 3D Tele-Immersion," in *IEEE ICASSP*, Florence, Italy, 2014, pp. 6162 - 6166.

[6] R. Mekuria , P. Cesar., ., "A Basic Geometry Driven Mesh Coding Scheme with Surface Simplification for 3DTI," *IEEE Communication Society: Multimedia Communications Technical Committee E-letter*, vol. 9, no. 3, pp. 6-8, May 2014.

[7] A. Doumanoglou, D. Alexiadis, D. Zarpalas, P. Daras., "Towards Real-Time and Efficient Compression of Human Time-Varying-Meshes," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 24, no. 12, December 2014.

[8] Z. Wang and C. Bovik, "Mean Squared Error: love it or leave it? A new look at signal fidelity measures," *IEEE Signal Processing Magazine*, vol. 26, no. 1, pp. 98-117.

[9] A. Ciancio, J.F.L. Oliveira, F.M.L Ribeiro, E. ABda Silva., A. Said.,., "Quality perception in 3D interactive environments," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2013.

[10] W. Wu, A. Arefin, G. Kurillo, P. Agarwal, K. Nahrstedt, R. Bajcsy., "Color-plus-Depth Level-of-Detail in 3D Tele-immersive Video: A Psychophysical Approach," in *Proceedings of the 19th ACM international conference on Multimedia (MM '11)*, pp. 13-22.

[11] ITU-T , "Methods for objective and P.800.2 subjective aassessment of speech quality. Mean opinion score interpretation and reporting," ITU Recommendation P.800.2, 2013.

[12] ITU-T, "P.910 Audiovisual quality in multimedia services. Subjective video quality assesment methods for multimedia appilcations," P.910,.

[13] V. Venkatesh V., Morris, M.G., Davis, G.B. Davis, F.D.., "User Acceptance of Information Technology: Toward A Unified View," *MIS Quarterly*, vol. 27, no. 3, pp. 425-478, 2003.

[14] G. Charness, U Greezy, U., Kuhnc, M.A.., "Experimental methods: Between-subject and within-subject design," *Journal of Economic Behaviour & Organization*, pp. 1-8, Aug. 2012.

[15] Microsoft Asia, "NEWT Network Emulator for Windows," Microsoft Software, 2013.