Space Connection: A New 3D Tele-Immersion Platform for Web-Based Gesture-Collaborative Games and Services

Chun-Han Lin
National Chengchi University
Taipei, Taiwan
Email: john.lin0420@gmail.com

Pei-Yu Sun
National Chengchi University
Taipei, Taiwan
Email: vipfifisun@gmail.com

Fang Yu
National Chengchi University
Taipei, Taiwan
Email: yuf@nccu.edu.tw

***Abstract*—The 3D tele-immersion technique has brought a revolutionary change to human interaction- physically apart users can interact naturally with each other through body gesture in a shared 3D virtual environment. The scheme of cloud- or web-based applications on the other hand facilitates global connections among players without the need to equip with additional devices. To realize web-based 3D immersion techniques, we propose *Space Connection* that integrates techniques for virtual collaboration and motion sensing techniques with the aim of pushing motion sensing a step forward to seamless collaboration among multiple users. Space Connection provides not only human-computer interaction but also enables instant human- to-human collaboration with body gestures beyond physical space boundary. Technically, to develop gesture-interactive applications, it requires parsing signals of motion sensing devices, passing network data transformation, and synchronizing states among multiple users. The challenge for developing web-based applications comes from that there is no native library for browser applications to access the application interfaces of motion sensing devices due to the security sandbox policy. We further develop a new socket transmission protocol that provides transparent APIs for bowsers and external devices. We develop an interactive pingpong game and a rehabilitation system as two example applications of the presented technique.**

I. Introduction

The news that Apple acquires PrimeSense brings 3D motion sensing techniques to a hot spot. PrimeSense is best known for developing gesture control for Microsoft's Kinect sensor, and has made strides towards bringing the technology mobile. As Apple conducted this investment due to the huge market potential of motion sensing, the market of motion sensors was estimated to reach $4.8 billion by 2016, driven by motion gaming and consumer electronic mainly, indicating the dramatically increased demand of the motion sensing software in the near future.

Kinect leads motion sensing techniques to an even more prosperous era. Researchers applied Kinect into fields other than commerce. Chang, Chen, and Huang [10] designed interactive games to help motivate people with motor impairments to participate in rehabilitation. The therapists can normally assist one client at a time, but with this system, the therapists can review and rehabilitate more individuals with higher efficiency. Zafrulla, Brashear, Starner, Hamilton, and Presti [30] tried to put Kinect into educational usage. They tried to develop a learning gaming system for deaf children to learn sign language. Kinect was also put into the usage of mapping technology [17] in which people held Kinect in their hands and moved indoors to map the environment.

While there are quite a few literatures on how motion sensing leads to innovative and intuitive human-computer interactions, comparably less discussions were done on connection of human to human interactions [4]. The concept of 3D tele-immersion aroused this direction. We integrate several techniques for virtual collaboration and motion sensing with the aim of pushing motion sensing a step forward to seamless collaboration among multiple users. The presented platform *Space Connection* facilitates instant human-to-human cooperation with body gestures beyond physical space boundary.

The objective of this work is to provide a virtual collaborating environment that makes seamless real-body motional interaction possible. To this aim, we propose a new socket transmission protocol that enables motion sensing devices and browser applications to communicate simultaneously via binary data formats among multiple users. Since socket transmission is supported by most of the mainstream rich web application techniques, the protocol provides a transmission tunnel between browser applications and the native motion sensing library, allowing browser applications to invoke native motion sensing library seamlessly. We further implemented Socket Natural Interaction (SocketNI) by wrapping up popular native motion sensing library OpenNI. SocketNI acts as a middle layer that enables web application development with functionalities of OpenNI. That is to say SocketNI SDK is compatible with OpenNI standard interfaces.

There are two aspects of contributions of SocketNI: to the users and to the developers. (1) To the users: Users can access to a virtual socio-spatial world with more convenience. All they need to do is type in the URL and browse to the motion sensing web application. They no longer need to download large executable files or purchase extensive hardware. Moreover, SocketNI provides a ubiquitous collaboration environment to the users. They can access applications without the limitation of space and interact with people elsewhere easily, changing the way how people social communicate with one another. (2) To the developers: Nowadays APIs for motion sensing applications are Microsoft Kinect for Windows SDK and OpenNI2 + NiTE2 (Prime Sense); however, none of them support web runtime environment. To convert their applications into web-based applications, developers have to establish connection, maintain sessions, synchronize application states, constitute protocol, and deal with data streaming. We provide SocketNI to address this concern. We wrapped the codes up

F. Yu is partially funded by the grant MOST-103-2221-E-004-006-MY3.

into a software development kit (SDK) that supports various web runtime environments such as Java applet, Unity3D and Silverlight. Also, SocketNI SDK has been implemented in multiple languages such as C#, VB.NET and Java, which are most popular languages for developing web applications.

II. Related work

We briefly review previous related work in three folds: Tele-immersion techniques, motion sensing techniques and browser plugins.

A. Telepresence and 3D Tele-immersion

Telepresence [7] is the use of technology to establish a sense of shared presence or shared space among geographically separated members of a group. Researchers [25] tried to establish a virtual space and capture motions of involved person precisely and convert them into avatars to interact with people from another place. Fuchs, State, and Bazin [16] introduce a dynamic and realistic 3D telepresence system. They found that when two depth camera sense the same object, they are very likely disrupted by one another causing holes in the outcome; therefore, they built algorithms which enabled hole-filling and smoothing, data merging between cameras, and 3D eye position tracking. Beck, Kunert, Kulik, & Froehlich [6] came up with establishing a depth correction table which can help calibrated data more precisely. In addition, they further extend their research to interaction of two or more groups. They not only provide tracked users their own perspective correct images but also tested on face-to-face, side-by-side and decoupled scenarios.

3D Tele-immersion systems introduce a new way of communication between people. Broader audiences enjoy collaborative interaction in a joint immersive environment with other users from remote places for business and entertainment usage. To extend Telepresence to provides 3D Tele-immersion, there remain many challenges such as real-time 3D video reconstruction and compression, Quality of Service (QoS) of synchronization of multiple 3D streams, multi-view displays of 3D contents. Communication of 3D Tele-immersion requires large computational power, storage and bandwidth. The data transmitted in 3D Tele-immersion contain visual and vocal data; in addition, to achieve seamless user experience of coexistence and collaboration in virtual space, we need to synchronize the data from multiple sources and resynchronize when the user views change.

Several previous researches have been proposed with innovative frameworks to assist the performance of 3D Tele-immersion. TEEVE (Tele-immersive Environment for EVEry-body) [29] presents a flexible and cost-effective end-to-end cross-layer architecture that incorporates all participating devices for efficient resource utilization and quality of service. TEEVE also follows end-to-end setup protocol that performs as a gateway between data transmission of different users and initiate tele-immersive environment. Graphic morphing [12] was used to connect between first and second frames. The technique perform special effect in motion picture that one frame can change to another frame smoothly. Fechteler et al. [14] proposed building a database containing pre-processed data like depth maps, skeleton information, alpha masks, so that highly realistic animations can be formed by warping and merging pre-captured database images. Unlike previous work, we develop a cross-platform socket interface to facilitate Web based interactive applications through motion sensing devices like Kinect.

B. Motion sensing devices and applications

Due to Wii's commercial success, the era of motion sensing started in November 2006 after Nintendo announced its release of Wii remote. With infrared camera tracker, vibration motor, and accelerometer, Wii remote can turn into guns, bats, swords, steering wheels [24] in games. One of the best-selling games of Wii, Rayman Raving Rabbids 2, made extensive and innovative attempts [8] of interactions between players and Wii. In Rayman Raving Rabbids 2 players have to shake, swing, hold still, press the buttons on the Wii Remotes, or put the Wii Remotes near their ear or mouth to pretend they are calling cellphones or drinking water. In addition to gaming business, Wii remote was implemented in other fields like rehabilitation [13][15] and education [23][5]. Later, gesture sensing technologies, e.g., Kinect, facilitate innovative commercial services, e.g., Dynamic dressing room [9], which allows people to utilize gesture to select clothes for fitting and screens the results on the display screen, reduced the time for clothe fitting. Customers can also easily visualize their look when they shop clothes on-line at home, cutting chances of buying unfit clothes.

Kinect lifts human computer interactions to another level by detaching wearable apparatus and introducing a brand new type of interaction. The equipment has a RGB camera, a depth senor and a multi-array microphone. It makes use of infrared (IR) lights to obtain depth data, and can track 48 points of our body for up to two players [21]. Users interact with their computers 3 dimensionally using gestures and vocal commands; in other words, users are not bounded by keyboards, mice or controllers and thus have intuitive experiences with digital contents. As soon as it is released, Kinect had been assisting optimization of business process. Kroeckel and Bodendorf [18] tried implementing Kinect into customer tracking and tracing at the point of sale to optimize and offer innovative services.

In addition to its wide applications, Kinect triggered new development of motion sensing devices. ASUS Company located in Taiwan released Xtion Pro and Xtion Pro Live to share the slice of the motion sensing industry. Leap Motion, a portable device with two IR cameras and three infrared LEDs, which can detect very fine movements of hand gestures, observe the objects to a distance of 1 meter with higher accuracy [27], was also produced. Last, Kinect 2.0, a new version of Kinect that enables advance gesture, facial, and voice recognition, was developed. It not only tracks skeletons of the users with higher precision but also scans the entire bodies of users vividly even in the dark. With the ability to trace people's heartbeats, muscle usage, and facial expression for up to six people simultaneously.

C. Browser Plugins Development

Plugins are external binaries that add new capabilities to a web browser and are loaded when content of the type they declare is embedded into a page. Google Chrome uses Netscape Plugin API (NPAPI) [2] as a way to access native OS. NPAPI is one of common cross-browser plugin frameworks to exchange data with the browser. It is not only implemented by Chrome, but also by Firefox and most other web browsers, excluding Microsoft Internet Explorer, which stopped supporting it in favor of ActiveX. Although Netscape Plugin API (NPAPI) is intended to be platform independent, in reality it is not fully so. It is a weak standard, and every browser implements

it somewhat differently. Therefore, Google developed Pepper Plugin API (PPAPI) to solve the legacy problem. Pepper Plugin API (PPAPI) was developed to minimize the changes from legacy NPAPI, hoping to ease adoption by browser vendors and plugin developers. Moreover, PPAPI also increases the portability and performance issues with NPAPI, particularly increases the capabilities such as generic 2D and 3D graphics and audio. Native Client (NaCl) is a platform-independent sandboxing technology which enables untrusted native code to run in a web browser safely. It allows real-time web applications that are compute-intensive and/or interactive (e.g. games, media, large data analysis, visualizations) to leverage the resources of a client's machine and avoid costly network access while running in a secure environment with restricted access to the host.Native Client (NaCl)started as a downloadable NPAPI plugin for multiple browsers, including Firefox, Safari, Opera and Chrome and was designed to transparently load and run other NPAPI plugins compiled as nexes and embedded into the page as a source file of an plugin element. Internet Explorer uses ActiveX controls [1] that adapts its earlier Component Object Model (COM) and Object Linking and Embedding (OLE) technologies for content downloaded from a network , particularly in the context of World Wide Web (WWW).

We aimed at developing a cross-platform framework for thin-client, all-web-usable 3D gesture-interactive applications. To achieve the goal, it requires seamless connection from motion sensing device to web application. Severe delay would put stop to the application; therefore, it is important to design a mechanism that can connect motion sensing devices to the web with efficiency. We first developed Socket Object Exchanging Framework (SOX) as a data exchanging framework to cross language data exchange and remote procedure invocation. Based on SOX, we developed Socket Natural Interaction (SocketNI), which is a library that allows web application to directly access native OpenNI library to communicate with motion sensing devices.

Finally, there is a trend that 3D sensors will be embedded into nowadays devices, like smart phones, televisions, and notebook. Listed below are some of the issues that can indicate this trend's happening. (1) Primesensen Inc. developed Capri, which is a motion sensing chip so small that you can hold it with your two fingers. (2) HP embed Leap Motion into notebook, HP ENVY17 Leap Motion, so that users can use their fingers to control the notebook without using keyboard. (3) Samsung released smart TV that doesn't need a controller to switch channel. Users use their body gestures to interact with the TV. And (4) Samsung Galaxy S4 has eye tracking camera embedded. We can foretell that the era of embedded 3D motion sensor is near; people can physically interact with remote peers without the limitation of space and affordability with just your smartphone and notebook using the presented techniques including the cross-platform technique SocketNI.

This work consists of two main impacts: For technical impacts, we expect that the platform and the public-available application interface of Space Connection can be leveraged by managers and programmers to develop new collaboration services with revolutionary motion sensing applications. For social impacts, we envision such collaboration through the new generation of the Internet creates a novel social network connecting people's space beyond physical boundary of real worlds.

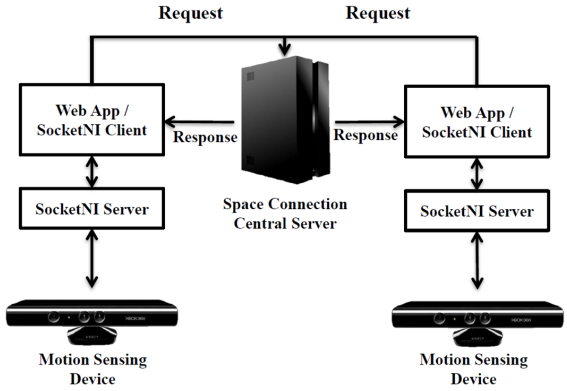

Fig. 1. Space Connection Architecture

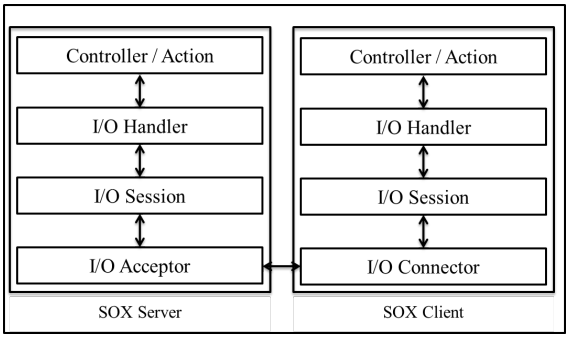

Fig. 2. Socket Object Exchange Framework (SOX)

III. Web-based Gesture Collaboration

Figure 1 shows the architecture of Space Connection. For each user who has their motion sensing device plugged into their computer, a local SocketNI server ,which is implemented base on SOX, will take control of the device and wait for any incoming command that send from web application client. Also, a web application which is implemented with our SocketNI SDK will communicate with the local SocketNI server so as to get the data which generated by motion sensing device. To synchronize the states such as skeleton positions or depth map among multiple web applications, we adopt a central server framework among players. A player is selected as the center to take control of communications. We evaluate the latency in Section V.

A. Socket Object Exchange Framework (SOX)

SOX's implementation allows different languages to exchange their objects and to invoke remote functions of each other through socket communication. It provides an interface that is implementable on different kinds of programming languages. Besides, SOX used a model-controller-liked foundation to invoke functions on each side. Current implementation of SOX can support JAVA and C#, and it can be implemented on more programming languages such as ActionScript or JavaScript. Figure 2 shows the components under SOX framework. An I/O Acceptor is used to establish a server listener which accepts clients' connection request. I/O Connector, in the other hand, is used to establish the connection with the server. After a connection is established, I/O Session component will maintain the session layer information and provide byte-based I/O function to read or write data to the remote end. I/O Filter component is the data codec of SOX framework, it is used to

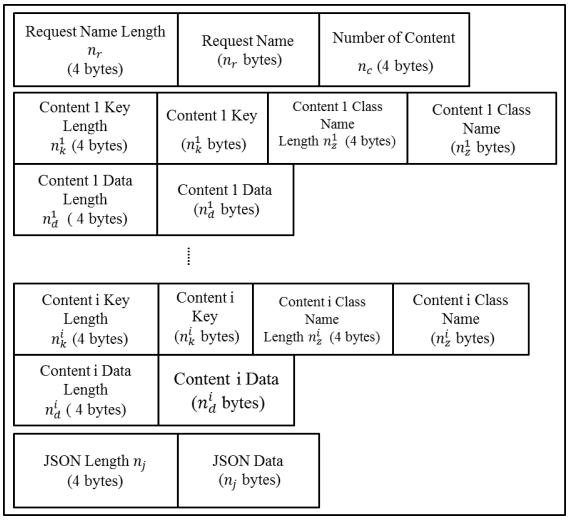

Fig. 3. Byte Level Data Layout of SOX

encode high-level object into binary or decode low-level binary data received by I/O Session component into high-level object.

Figure 3 shows the byte level data layout of which is encoded by I/O Filter. The first 4 bytes are used to describe an integer n_r which is this length of request name, and the fallowing n_r bytes are used to describe the request name string that encoded into UTF-8 format. The next 4 bytes are an integer n_c used to describe the amount of contents. Contents in SOX framework are elements that aren't suitable to be encoded into JSON format. JSON is a data-interchange format that is completely language independent, human-readable, and can be easily parsed by C, C++, C#, Java, Perl, and many others languages. It is built by key-value pairs using comma and brackets for segmentation. It changes data into UTF-8 character format and transmit among different programming language. The reason why some elements in transmitted data structure aren't suitable to be directly encoded into JSON is because JSON format describe object with UTF-8 context and makes data size much larger than they were. In most of the situations, serialize object directly into JSON could be feasible if the transmitted data do not contain a lot of large binary data; however, when it comes to BLOB (Binary Large Object), the size expansion could severely cause transmission delay and affect user experience. For example, a bitmap data contains a large byte array, if such array is described by UTF-8 string than the size of data will be over-expansed since an UTF-8 character takes 1 6 bytes to encode. If we would like to transmit a white pixel that contains 3 bytes which are (255,255,255) in RGB, under UTF-8 encoding, the three bytes will become a string "255,255,255" that expanse the size from 3 bytes to 11 bytes. Although this expansion isn't obvious when we send simple object but when it comes to BLOB such as bitmap the data can grow tremendously. SOX can distinguish which field should be encoded into raw bytes rather than JSON and then extract it out from its original data structure and become a so called content.

After the request has been sent to the remote end, I/O Filter will then decode the request and put the content back to its original data structure. For each content, we designed the fallowing data layout: The first 4 bytes of the i_{th} content is an integer n_k^i means the length of content key, and the fallowing n_k^i bytes is the content key. A content key is simply a hash code of the content. This content key is used to identify which field in the JSON data structure should the content put back after the whole data is sent to the remote end. An integer n_z^i that takes 4 bytes is then write into the buffer to describe the length of content class name and the fallowing n_z^i bytes is a string that represent the content's class. When SOX is implemented in object-oriented language, this class will define the way how the content be serialized and desterilized. After the class name is written, SOX then write a 4-byte integer n_d^i , which represents the length of content byte data, into the buffer, and the fallowing n_d^i bytes is the binary data that the content is serialized to. After all of the contents are written into buffer, we then write an integer n_j of 4 bytes that describe the length of JSON string, and the fallowing n_j bytes is the JSON string which is encoded into UTF-8 format. The JSON string here is derived from the serialization result of the transmitted object, however, the JSON string does not contain any BLOB since we've already extract them out as content, so the space utilization of this JSON context is quite efficient.

I/O Handler component acts as an invocation manager that determine which action and controller needs to be triggered after an object is received. Also, it controls the behavior when each life-cycle-related event happens and notifies the blinded events. Controller components are very critical part in SOX framework. They manage a set of actions (functions) that could be triggered from request of remote ends. Under SOX framework, there are two kinds of controller, one is Disposable Controller and the other is Resident Controller. Disposable Controllers are instantiated whenever a request triggers its action and will be released once the request finished. Resident Controller, however, is instantiated when a session opened and it won't be released until the session closed, so Resident Controller provides a persistence layer in a session thus can be used to manage the binding of object and its proxy object between server and client. For example, if a proxy object on client call a method, SOX will trigger the method of the real object on the server side which is bind with the proxy object, and its return value will be send back to the client synchronously or asynchronously on developer's demand. If the return value send back synchronously, then the code will look like a normal method call, but in fact the calls has traveled through the Internet and get its return value back, and all of the complex communication mechanisms are taken over by SOX.

B. Socket Natural Interaction (SocketNI)

OpenNI is an open source SDK which enable developers to easily create middleware libraries and applications for 3D sensors. NiTE is a robust 3D computer vision middleware program with multiplatform supported. It utilizes algorithms for the depth, color, IR and audio information received from the hardware device, which enable application to perform functions such as hand locating and tracking. Although OpenNI and NiTE library can be dynamically linked to the executable applications, it is un-permitted for a normal web application to access motion sensing device's data stream by these library due to security sandbox policy in the browser. Instead of developing plugins for each kinds of browser, for example, develop ActiveX plugin for Microsoft Internet Explorer, Native Client for Google Chrome and etc., we come up with a solution that is compatible for every browser, mainstream operating system and rich Internet application platform that support socket communication Socket Natural Interaction (SocketNI). SocketNI is a web-based implementation of OpenNI 2.0 interface and NiTE 2.0 middleware. Basically, SocketNI is developed on the

foundation of SOX framework, so it enables web application to seamlessly invoke the method of OpenNI and NiTE. SocketNI contains two parts, one is SocketNI server and the other is SocketNI client. SocketNI server is implemented in JAVA and has wrapped OpenNI, NiTE and some common motion sensor drivers with it. So it can serve under multiple platforms such as Windows, Mac OS, and Linux. SocketNI server act as a background service and once the device plugged into user's computer, SocketNI server will listen for the request from web application and invoke the corresponding function of OpenNI or NiTE thus control the device. SocketNI client is a library (SocketNI SDK) also developed under SOX framework and can be used to invoke OpenNI and NiTE function on SocketNI server. Since SocketNI client is a web-based implementation of OpenNI and NiTE, it provides all the APIs substitution of OpenNI and NiTE. By using SocketNI SDK, web application developers can use the APIs of SocketNI as if they are really using OpenNI and NiTE in their web application with almost no difference. Currently SocketNI client support JAVA and C# so it can be applied on rich Internet applications (RIAs) such as Java applet, Silverlight and Unity web application. The RIAs we discussed above usually will require a special hand-shake while a new socket connection is established due to the existence of sandbox policy. SocketNI server also implements the hand-shake mechanism for the above RIAs thus to simplify and fasten the developing progress for development that adopt SocketNI SDK. In addition, SocketNI inherits cross-device feature from OpenNI, so with SocketNI, users of different motion sensing devices can interact in same web application and complete their works or gaming activities collaboratively.

C. Application state synchronization of Space Connection

After SocketNI equips web applications with ability to access motion sensor's data. we show how Space Connection realizes collaborative interaction among multiple users on the Internet. In the Space Connection system, we build a server on the cloud that acts as a state arbiter among multiple users. To keep every user's states synchronized, one of the biggest difficulties that Space Connection has to overcome is the latency when data is transmitted among multiple users. The latency not only affects user experience but also cause abnormal program behavior. Space Connection used a centralized way to keep program states synchronized among peers. In other words, every change that could affect the collaboration result should be controlled under central server's arbitration before present on the screen of each peer. Figure 4 shows an example how Space Connection deals with the synchronization task between two players' skeleton position. When client A raises his left hand, instead of updating its avatar in the web application immediately, it sends the position changes to the central server first. After central server got the position changes, it commits the changes and broadcasts the changes to every peer. Clients' application can only update the states of program according to the changes that is approved by the central server.

IV. Applications

As mentioned, 3D Tele-immersion has been applied in different fields like gaming, education, tele-conferencing, re-habilitation, and choreography. It was widely used in gesture collaborative activities. Our service platform also has the potential to fulfill the needs. SocketNI provides a quick way for developers to build motion sensing web applications. They can easily transform applications existing using our SocketNI SDK.

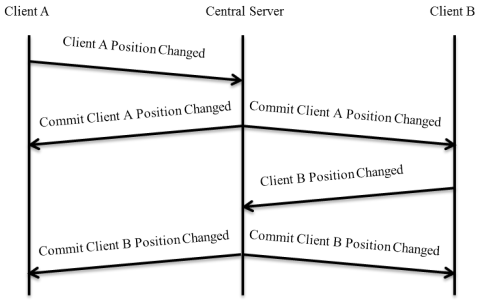

Fig. 4. Example of Space Connection State Synchronization

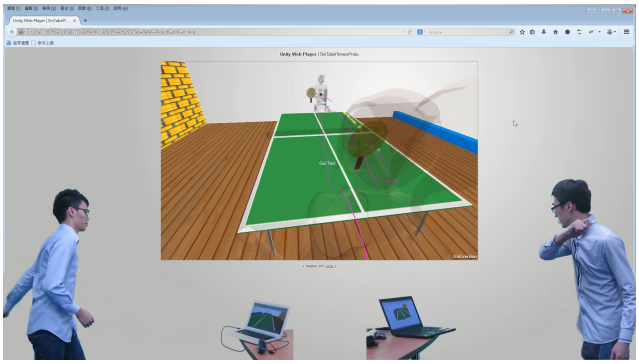

Fig. 5. A Table tennis game)

We aimed to develop a framework which enable users to use 3DTI technique on-line, using website browsers like Google Chrome, Firefox and Internet Explorer. With the development of new motion sensing technique with thin-client concept, we can record more detailed information and link people from different places and background seamlessly and easier. We direct in two folds our applications with the presented SocketNI below.

A. Interactive Gaming

Games [28][26] were among the first applications developed with 3D Tele-immersion. On-line games can have social bonding and collective physical exercise without the restriction of space. SocketNI challenges the way people interact in the past. It not only creates an invisible but conceptually presented space that is un-reproduce-able outside the 3D environment but offer an easy access to the users and helps developers to gain larger crowds of users.

Figure 5 is the picture of our table tennis game that is developed based on SocketNI SDK. Two players are equipped with difference devices: one PC with Kinect and one MAC with Xtion Pro. In the gaming scene, each translucent avatar stands for one player. They are playing at different places using different devices. For computers, we used one Mac, one SONY VAIO notebook, and for motion sensor, one is Microsoft Kinect, and the other is ASUS Xtion Pro. It can be tell from the operating system and the equipment we used that our SocketNI SDK support cross-platform and cross-device situations. The actual data streaming is fluent despite the fact that this is a 3D virtual environment. We scanned the skeleton of player and synchronize the 3D joint coordinate on both sides.

Also, we design two ways to minimize disturbance from background or other pedestrians.

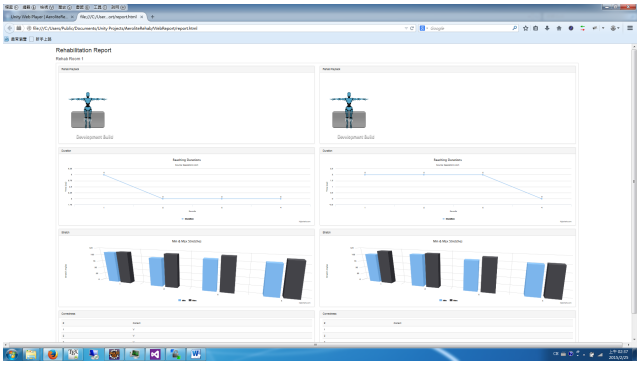

Fig. 6. Real-time Training Report (rehab motion playback, motion duration, stretching angles, memorization correctness table)

1) Every user has to pose a gesture of crossing their hands before he staring so that the program can understands that he is the exact player.
2) We can turn on and show the actual depth map on the screen to find out what disturbances (irrelevant people or other objects) are sensed thus to remove them from the realistic scene.

B. Interactive Training

SockenNI can also be used to implement real-time interactive training similar to sport training [19], rehabilitation [20][22][11]. The trainer and trainee can access instant accurate 3D data of their gesture.

It is well-motivated developing rehabilitation applications. According to World Health Organization, 15 million people worldwide suffer from stroke every year. Nearly six million [3] die and another five million are left permanently disabled. Rehabilitation is an effective way to better up the situation; however, it is hard for them to travel back and forth from home to hospital frequently due to disability. In the past, rehabilitation relied on personal therapy. Patients with low mobility have to travel far to reach the clinic. Other alternatives include personalized equipment which usually associated with high costs.

On-line interactive rehabilitation system enabled them to exercise with other patients under supervision of doctors. Full bodies of users are captured and shown in real-time, which cannot be acquired elsewhere. Doctors can utilize multiple displays to view patients' performance from all angles. At the same time, by allowing patients to interact with each others, it enhances the motivation for patients to do rehabilitation. We provide a memory game. There are eight rocks in the screen and a sequence of number will appear and disappear within a short time. Patients have to memorize the sequence and touch the rocks in sequence collaboratively. With this design, boring stretching rehabilitation and memorization exercise can be transformed into a cooperating game between multiple patients and can further stimulate the motivation for them to take such rehab session.

Figure 6 demonstrates a real-time doctor site analysis report that is formed after the training game. The result contains patients' motion video playback, stretching angle, endurance, and memorization result. This report provides a more concrete and precise data of patients. In the future, by accumulating the data, the database could become big data and can be apply to rehabilitation related research.

As the final remark, 3D Tele-immersion has created various innovation services: patients of healthcare rehabilitation can interact with their physical therapists at home, avoiding burdensome physical moving and traveling. Athletes can play sports with external players without space and weather's limitation. Dancers can choreograph with other experts instantly, which help refining dance movements and establishing relationships. Using the presented techniques, those tasks can be web-based applications connected with a computer and a motion sensing device.

V. Performance Evaluation

Motion data are transmitted from local SocketNI server via SOX and then are passed to the other peers via the Internet. We design an experiment measuring the latency of SOX in this section. In each frame, SOX transmits data including depth map, user map, 3D joint coordinates and orientation quaternions of users' skeleton. A depth map is an array that records the depth of each pixel where the value ranges from 0 to 10000; therefore, each pixel is encoded by two bytes. The size of a depth map is 320×240 pixels with total length 153600 bytes $(320 \times 240 \times 2 = 153600)$. User map is quite similar to depth map instead that user map is used to map pixels to users' identifiers. Programmers can distinguish the owner of each pixel using user map. User map has the same size as depth map does. Besides those two maps, SOX also transmits for each user the positions and orientations of 15 different joints in 3D coordinates and quaternions. As more users stand in front of a single motion sensing device, the length of depth map and the length user map remain the same, but data of coordinates and orientation quaternions of joints increase linear to the number of users. Table I shows how average latency of SOX increases by adding users.

TABLE I. SOX Transmission Latency

Users	Avg. Delay(ms)
1	15
2	32
3	46

After all the joint coordinates and orientation quaternions are received by the browser application and are applied to the corresponding joints of an avatar model, the avatar's states are then synchronized in the scene of each peer. Note that the participated peers are within campus WIFI to access the Internet in our experiment (might be faster than general cases). To accomplish state synchronization, the web application keeps observing every joint coordinate and orientation quaternion to track changes of those values. Every time when a change of value happens, the web application sends the new value to the central server who broadcasts the new value to other peers. A maximum sending rate in this experiment is restricted to 30 requests per second. We use UDP as the transmission protocol. The reason to trade reliability with performance is because the lost packets could soon be recovered by next ones.

Table II shows the average transmission latency of each joint's data in millisecond. As the number of peers increase, latency increases faster than the linear increase of the data size on average. It takes 24.04 milliseconds to transmit a frame data from one peer to another with 2 participants; the cost increases to 105.7 milliseconds t from one to another with 3 participants. The performance could be improved by adopting peer-to-peer framework rather than the presented centralized framework.

TABLE II. STATE SYNCHRONIZATION LATENCY

Joints	2 Peers	3 Peers
Left Shoulder	28.5	102.9
Right Shoulder	22.6	87.9
Left Arm	23.7	98.9
Right Arm	22.5	84.9
Left Hand	25.0	93.3
Right Hand	22.0	112.9
Spine	27.5	105.9
Neck	23.3	91
Head	22.9	116
Left Hip	24.5	129
Right Hip	23.6	111
Left Leg	24.3	108
Right Leg	23.3	121.9
Left Foot	24.0	104
Right Foot	22.9	117.9
Avg. Network Delay	24.04	105.7
Bytes Sent Per Second	9057	16143
Bytes Received Per Second	8237	13149

VI. CONCLUSION

We present an interoperability infrastructure for creating 3D applications using 3D motion sensor data within Web browsers. The approach uses a network protocol called SOX to connect web applications with a 3D motion sensor via an OpenNI 2 style interface. The use of SOX is justified by the need to send data in binary format to prevent large amounts of data expansion that occurs when using JSON. A table tennis game and a rehabilitation game are created as proofs of concept of the technique.

REFERENCES

[1] Introduction to activex controls. https://msdn.microsoft.com/en-us/library/aa751972%28v=vs.85%29.aspx.

[2] Native client. http://www.chromium.org/nativeclient.

[3] Stroke statistics | internet stroke center. http://www.strokecenter.org/patients/about-stroke/stroke-statistics/.

[4] D. S. Alexiadis, P. Kelly, P. Daras, N. E. O'Connor, T. Boubekeur, and M. B. Moussa. Evaluating a dancer's performance using kinect-based skeleton tracking. In *Proceedings of the 19th ACM International Conference on Multimedia*, MM '11, pages 659–662, New York, NY, USA, 2011. ACM.

[5] H. Andreas, S. Selver, S. Christian, E. Martin, D. Matja, and H. Bo. Nintendo wii remote controller in higher education: Development and evaluation of a demonstrator kit for e-teaching. *Computing and Informatics*, 29(4):601–615, 2010.

[6] S. Beck, A. Kunert, A. Kulik, and B. Froehlich. Immersive Group-to-Group Telepresence. *IEEE Transactions on Visualization and Computer Graphics*, 19(4):616–625, Apr. 2013.

[7] W. Buxton. Telepresence: Integrating Shared Task and Person Spaces. 92.:123–129, 1992.

[8] M. Casamassina. Pre-e3 2007: Hands-on rayman raving rabbids 2 - IGN. http://www.ign.com/articles/2007/07/09/pre-e3-2007-hands-on-rayman-raving-rabbids-2, July 2007.

[9] H.-T. Chang, Y.-W. Li, H.-T. Chen, S.-Y. Feng, and T.-T. Chien. A dynamic fitting room based on microsoft kinect and augmented reality technologies. In M. Kurosu, editor, *Human-Computer Interaction. Interaction Modalities and Techniques*, number 8007 in Lecture Notes in Computer Science, pages 177–185. Springer Berlin Heidelberg, Jan. 2013.

[10] Y.-J. Chang, S.-F. Chen, and J.-D. Huang. A kinect-based system for physical rehabilitation: A pilot study for young adults with motor disabilities. *Research in Developmental Disabilities*, 32(6):2566–2570, Nov. 2011.

[11] C.-H. Chen, J. Favre, G. Kurillo, T. Andriacchi, R. Bajcsy, and R. Chellappa. Camera networks for healthcare, teleimmersion, and surveillance. *Computer*, 47(5):26–36, May 2014.

[12] S. Chen and K. Nahrstedt. Activity-based synthesized frame generation in 3dti video. In *2013 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6, July 2013.

[13] J. E. Deutsch, M. Borbely, J. Filler, K. Huhn, and P. Guarrera-Bowlby. Use of a low-cost, commercially available gaming console (wii) for rehabilitation of an adolescent with cerebral palsy. *Physical Therapy*, 88(10):1196–1207, Oct. 2008.

[14] P. Fechteler, A. Hilsmann, P. Eisert, S. V. Broeck, C. Stevens, J. Wall, M. Sanna, D. A. Mauro, F. Kuijk, R. Mekuria, P. Cesar, D. Monaghan, N. E. O'Connor, P. Daras, D. Alexiadis, and T. Zahariadis. A framework for realistic 3d tele-immersion. In *Proceedings of the 6th International Conference on Computer Vision / Computer Graphics Collaboration Techniques and Applications*, MIRAGE '13, pages 12:1–12:8, New York, NY, USA, 2013. ACM.

[15] A. Fraser, M. Annett, and W. F. Bischof. Lean on wii: Physical rehabilitation with virtua reality wii peripherals. 154(IOS Press Ebooks):229 – 234, 2010.

[16] H. Fuchs, A. State, and J.-C. Bazin. Immersive 3d Telepresence. *Computer*, 47(7):46–52, July 2014.

[17] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon. Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. In *In Proc. UIST*, pages 559–568, 2011.

[18] J. Kroeckel and F. Bodendorf. Visual customer behavior analysis at the point of sale. *International Journal On Advances in Systems and Measurements*, 5(3 and 4):178–187, Dec. 2012.

[19] G. Kurillo, R. Bajcsy, K. Nahrsted, and O. Kreylos. Immersive 3d environment for remote collaboration and training of physical activities. In *IEEE Virtual Reality Conference, 2008. VR '08*, pages 269–270, Mar. 2008.

[20] G. Kurillo, T. Koritnik, T. Bajd, and R. Bajcsy. Real-time 3d avatars for tele-rehabilitation in virtual reality. *Studies in Health Technology and Informatics*, 163:290–296, 2011.

[21] T. Leyvand, C. Meekhof, Y.-C. Wei, J. Sun, and B. Guo. Kinect identity: Technology and experience. *Computer*, 44(4):94–96, Apr. 2011.

[22] S. Obdrzalek, G. Kurillo, J. Han, T. Abresch, and R. Bajcsy. Real-time human pose detection and tracking for tele-rehabilitation in virtual reality. *Studies in Health Technology and Informatics*, 173:320–324, 2012.

[23] E. Pearson and C. Bailey. Evaluating the potential of the nintendo wii to support disabled students in education. - version details - trove. 2007.

[24] A. Shirai, E. Geslin, and S. Richir. WiiMedia: Motion analysis methods and applications using a consumer video game controller. In *Proceedings of the 2007 ACM SIGGRAPH Symposium on Video Games*, Sandbox '07, pages 133–140, New York, NY, USA, 2007. ACM.

[25] W. Steptoe, J. Normand, O. Oyekoya, F. Pece, E. Giannopoulos, F. Tecchia, A. Steed, T. Weyrich, J. Kautz, and M. Slater. Acting Rehearsal in Collaborative Multimodal Mixed Reality Environments. *Presence*, 21(4):406–422, Nov. 2012.

[26] M. Tamai, W. Wu, K. Nahrstedt, and K. Yasumoto. View control interface for 3d tele-immersive environments. In *2008 IEEE International Conference on Multimedia and Expo*, pages 1101–1104, June 2008.

[27] F. Weichert, D. Bachmann, B. Rudak, and D. Risseler. Analysis of the accuracy and robustness of the leap motion controller. *Sensors*, 13(5):6380–6393, 2013.

[28] W. Wu, A. Arefin, Z. Huang, P. Agarwal, S. Shi, R. Rivas, and K. Nahrstedt. "i'm the jedi!" - a case study of user experience in 3d tele-immersive gaming. In *2010 IEEE International Symposium on Multimedia (ISM)*, pages 220–227, Dec. 2010.

[29] Z. Yang, K. Nahrstedt, Y. Cui, B. Yu, J. Liang, S.-H. Jung, and R. Bajscy. TEEVE: the next generation architecture for tele-immersive environments. In *Seventh IEEE International Symposium on Multimedia*, pages 8 pp.–, Dec. 2005.

[30] Z. Zafrulla, H. Brashear, T. Starner, H. Hamilton, and P. Presti. American sign language recognition with the kinect. In *Proceedings of the 13th International Conference on Multimodal Interfaces*, ICMI '11, pages 279–286, New York, NY, USA, 2011. ACM.