3D TELE-IMMERSION PLATFORM FOR INTERACTIVE IMMERSIVE EXPERIENCES BETWEEN REMOTE USERS

Nikolaos Zioulis, *Dimitrios Alexiadis*, *Alexandros Doumanoglou*, *Georgios Louizis*,
Konstantinos Apostolakis, *Dimitrios Zarpalas*, *and Petros Daras*, *Senior Member, IEEE*
Centre for Research and Technology Hellas, Information Technologies Institute
6th Km Charilaou-Thermi Road, Thessaloniki, Greece

ABSTRACT

Tele-immersion (TI) related technologies can change the way people interact and bridge the gap between the physical and digital worlds. However, while the technology itself advances, most developed platforms have complex setups and require large investments. In this work, a low-cost platform is introduced, integrating multiple TI-related advances. Focusing on ease of use and rapid deployment, a fast and fully automatic calibration method is proposed. The platform enables real-time 3D reconstruction of users and their placement into a pre-authored 3D environment. Moreover, interaction is achieved through the user's body posture, removing the need for additional equipment and enabling natural control while immersed. Developing a real-time TI platform requires the efficient integration of several multidisciplinary elements. An elegant, minimal solution to these challenges is proposed and validated in a prototype TI multiplayer game, *SpaceWars*.

Index Terms— Tele-Immersion, 3D human reconstruction, Kinect, calibration, real-time, multi-user, platform

1. INTRODUCTION

Tele-immersion (TI) [1, 2] is an emerging technology, pivotal to future interactive 3D applications, emplacing people at different locations into shared virtual environments and facilitating real-time interaction between participants in a realistic way. Through this new medium, the barrier of physical presence is removed, creating new pathways in the industries of entertainment [3, 4], knowledge transfer [2], health-care [5, 6] and collaboration [7]. Developing an efficient TI platform is a demanding task due to its multidisciplinary nature: Expertise in various fields is required, including computer vision, compression/information theory, networking, high performance computing and computer graphics, as well as the integration of technologies from these fields [8].

One way to achieve tele-immersion is through synthetic avatar representations, with the user's motion captured and transferred to them in real time [4]. However, with such representations, the presence level lacks an important aspect, the actual user's appearance. By utilizing the recent advances in real-time 3D reconstruction, realistic user representations can be created that embed interaction-vital information, like facial expressions, hair movements and body deformations. This results into an increased awareness level, making the whole experience unconstrained, thus, increasing users' immersion.

TI platforms that focus on realistic users' representations employ multi-camera capturing systems. They can be classified with respect to the user data representation [9]: On the one end, with image-based representations, intermediate views are interpolated from the captured camera views. On the other end, with full-3D geometry-based representations, 3D reconstruction techniques are applied to generate both the geometry and texture information of the captured user, in the form of a textured 3D mesh. Relevant state-of-the-art, real-time approaches, can be classified into those employing i) dense stereo multi-camera systems and ii) multiple active range sensors. In [2], an efficient depth (disparity)-from-stereo method is proposed, where the extracted depth-maps are then used to generate multiple separate meshes, which are combined at the rendering stage to synthesize intermediate views for given viewpoints. Multiple Kinect sensors are used in [10] and [11] to produce 3D mesh representations by triangulating the depth maps acquired from each sensor and generating 3D meshes, with the color information encoded as a color-per-vertex attribute. In a more recent work [12], the depth images are fused via a volumetric method, producing a watertight 3D mesh. A weighted texture blending technique is utilized in order to generate a texture map from the different viewpoints' color images.

Aiming for remote and real-time interaction, future TI platforms should focus on real-time 3D reconstruction and transmission, where each participant's full 3D geometry and appearance is generated, enabling on-demand TI experiences. Such representations allow for i) seamless integration with most rendering engines , ii) collision detection, and iii) can also be leveraged for motion capturing, discarding the need of additional equipment. However, such systems require usually expensive capturing equipment and complicated installation configurations, unsuitable for rapid deployment. In [13] a TI testbed is described with two remote stations, one with 6 3D cameras and another with 12, while in [2] 36 grayscale and 12 color cameras are used for stereo 3D capturing. A portable

This work was supported by the EU funded project PATHway under contract 643491.

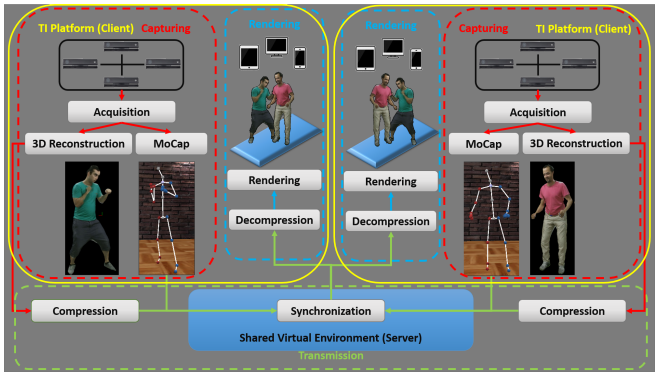

Fig. 1. The overall implemented architecture.

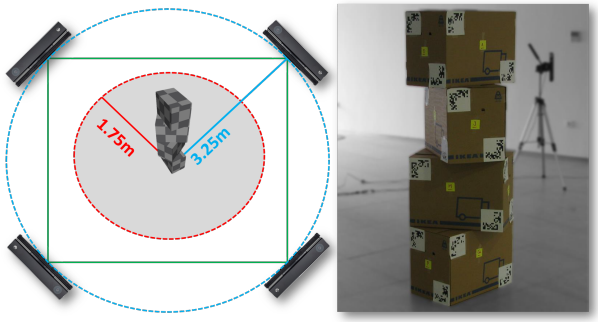

Fig. 2. Capturing setup and calibration structure.

one is also presented that only captures a partial view of the user with four color cameras. Another portable TI platform is described in [14], where a stereo camera positioned next to a projected screen is used along with a portable curtain-draped frame to segment and remove the background.

In this work, a portable and easily deployed platform for indoor spaces, operated as a local TI capturing station, is described. The proposed platform uses low-cost commercially available RGB-D sensors, and utilizes an easy-to-use, multi-sensor calibration scheme, which allows for coarse sensor positioning around a pre-defined capturing space, without further restrictions or custom configurations. By integrating technologies from multiple fields, the user's full 3D appearance and motion are captured in real-time and transmitted inside remote virtual environments, where interaction between multiple immersed participants is facilitated. The resulting 4D media can be consumed by a number of devices like personal computers, tablets, virtual reality headsets or 3DTVs. The platform was verified and demonstrated via a prototype TI game, *SpaceWars*.

2. PLATFORM OVERVIEW

The architecture of a TI platform reflects its multidisciplinary nature and is composed of three tiers [9]: i) capturing and reconstruction, ii) data compression and transmission and iii) rendering. The hardware components always include a multi-sensor integrated capturing station and presentation units (displays). The three important choices that define each implementation reside in the selection of the sensors, the data representation used and the network architecture. This section will structurally present the aforementioned choices by following the data processing chain. An overview of the proposed platform's architecture is given in Figure 1. Each remote client (yellow), corresponding to one user, captures and reconstructs the user's appearance in real-time using a multi-Kinect setup (red). Additionally, the user's motion is tracked, in the form of a skeleton structure, to enable interaction of the user with the virtual environment. The heavyweight 3D data are then compressed and transmitted (green) together with the lightweight skeleton data to the server. After synchronization, the global state is calculated and sent back to all receivers, where the 3D data are decompressed and rendered, reproducing the global state.

2.1. Multi-sensor capturing

Hardware: Although some previous works [2] highlighted the inherent problems of using multiple first-generation Kinect sensors (e.g. interference, limited depth range, etc.), other relevant works [10, 15, 12] employed this type of sensor, focusing on low-cost solutions. However, the increased quality offered by the new Kinect version 2 (accurate depth, HD color camera and minor interference issues) naturally led to its adoption in the proposed TI platform.

The capture site of the proposed TI system has a distributed capturing and centralized processing nature: it comprises of K Kinect V2 sensors and $K+1$ PCs, with K client PCs (one for each sensor) and one central PC collecting and processing the data (reconstruction, etc). Each client streams data to the central PC at real-time rates ($>$15fps) through a local Ethernet interface. The reason for using several computers is that the current Kinect V2 SDK does not support installation of multiple Kinects onto the same computer.

In order to achieve high exchange rates, the locally captured data at the client are encoded before transmission, following an intra-compression scheme to minimize transfer latency. The HD color data are compressed using standard JPEG compression [16], while a lossless algorithm is used [17] for the depth data.

Synchronization: Since the Kinect sensor does not come with an off-the-shelf hardware triggering solution, a software-based synchronization solution was opted for. Each sensor acquires frames asynchronously and only transmits them to the central PC once signaled. More specifically, a protocol based on a "*wait-for-all*" principle was implemented: a) the clients grab frames asynchronously; b) the central PC broadcasts a "*send now*" message; and c) each client transmits the last grabbed RGB-D frame. Disregarding varying sensor frame rates, this approach guarantees synchronization of the acquired frames up to camera shutter times (worst case of 33 ms for the Kinect sensor). In practice, with this signaled acquisition method, synchronization issues were observed only during very fast user's motions.

Spatial configuration: The proposed platform utilizes K=4 Kinect sensors, placed on a circle of radius >3m, at 90° angle intervals, and all pointing to the center of the captured scene. This configuration, shown in Fig. 2, can provide full coverage of the user in a circular region of radius >1.5m.

External Calibration: An "one-shot" calibration method, incorporating an easy-to-build structure, was developed. Four commercially available (standardized IKEA) packing boxes, of known dimensions, with QR code markers placed on them, are needed to build the calibration structure, which is shown at the right of Fig. 2. A detailed building manual is available at `http://vcl.iti.gr/spacewars/`. Additionally, the exact virtual counterpart of the calibration structure, a CAD 3D model (available at the same url address), was also designed. Let the 3D CAD model be denoted as $\mathbb{M} = \{\mathbb{V}, \mathbb{W}, U\}$, where $\mathbb{V} = \{\mathbf{V}_1, \mathbf{V}_2, \ldots, \mathbf{V}_m\}$ are the model vertices' 3D positions, $U(\mathbf{x})$ is the accompanying unwrapped model's texture and $\mathbb{W} = \{\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_m\}$ are the texture coordinates corresponding to each vertex.

Unlike standard checkerboard calibration approaches [2], in the proposed method the calibration structure acts as a registration anchor and needs only to be positioned at the center of the capturing space, in order to be captured once by the sensors. Given that each Kinect $k = 1, ..., K$ acquires a pair of color and depth frames $[I_k(\mathbf{x}), D_k(\mathbf{x})], \mathbf{x}^T = (u, v)$, these are aligned to the virtual CAD model, employing the SIFT transform [18]: SIFT correspondences $\mathbb{C} = \{c^i_{\mathrm{M}}, c^i_k\}, i = 1, \ldots, N$ are established between the model's unwrapped texture U and each viewpoint's k color image frame I_k. Each texture coordinate $c^i_{\mathrm{M}} \in \mathbb{W}$ of the virtual model corresponds to a vertex $V^i_{\mathrm{M}} \in \mathbb{V}$. Additionally, using each sensor's color-to-depth mapping function and the corresponding depth camera's intrinsic parameters, the 2D coordinates c^i_k are mapped into the 3D points $\mathbf{v}^i_k$. Other non-patented feature alternatives (e.g. A-KAZE [19]) can also be used to establish correspondences. Given the 3D position correspondences $\mathbf{V}^i_{\mathrm{M}} \leftrightarrow \mathbf{v}^i_k$, each sensor is aligned with the structure's global coordinate system through Procrustes analysis [20], resulting into the transformation matrices $\mathtt{RT}_k$. With the above described quick and fully automated method, the platform can be externally re-calibrated fast and effortlessly in the case of sensor dislocations, with minimal human interventions.

2.2. Real-time 3D reconstruction

A full-3D geometry-based representation of users offers some distinct advantages when placed inside virtual worlds, such as seamless integration with most rendering/game engines (lighting, shadows, collision detection). Consequently, the introduced platform focuses on real-time full-3D geometry reconstruction of the moving users. The methods were implemented using CUDA, to exploit the parallel processing capabilities of the GPU and perform near real-time.

Each sensor produces a stream of spatially and temporally aligned color and depth frames $[I_k(\mathbf{x}), D_k(\mathbf{x})]$, which are collected by the central processing computer to reconstruct the user's geometry and appearance, on a per-frame basis. We note that user foreground-background segmentation is performed at each client, which is straightforward using the depth data and is available also in the Kinect SDK.

Reconstruction of geometry: For each "foreground" pixel in each depth image $D_k\mathbf{x}$, the corresponding 3D position $\mathbf{X}_k(\mathbf{x}) = \mathtt{RT}_k^{-1} \cdot \Pi_k^{-1}\{\mathbf{x}, D_k(\mathbf{x})\}$ is generated, where Π_k^{-1} denotes the projection operator given the intrinsic depth camera's parameters, and $\mathtt{RT}_k$ is the extrinsics matrix of the sensor. Additionally, after employing an organized triangulation scheme (SDCT) [11] on the depth-image plane, the corresponding vertex normals $\mathbf{N}_k(\mathbf{x})$ are also calculated.

Given the vertex positions $\mathbf{X}_k$, a bounding box (BB) of the user is extracted. Considering the extracted BB, a Fourier Transform-based volumetric reconstruction method [21] is then employed to produce a manifold and watertight mesh. For details the reader is referred to [21] and [12]. In a nutshell, the vertex normals $\mathbf{N}_k(\mathbf{x})$ are "splatted" onto a voxel grid inside the user's bounding box, to produce the volumetric gradient-field $V(\boldsymbol{q})$ ($\boldsymbol{q}$ denotes the voxel index). The gradient-field $\mathbf{V}(\boldsymbol{q})$ is then "integrated", by multiplying with the integration filter $\hat{\mathbf{F}}(\omega) = j\omega/\|\omega\|, j = \sqrt{-1}$ in the frequency domain. The final volumetric function $A(\boldsymbol{q})$ is generated by addition of the X, Y, Z components of the integrated gradient field. The final triangulated mesh is extracted as the iso-surface $A(\boldsymbol{q}) = L$ using the marching cubes algorithm [22], with L the average value of $A(\boldsymbol{q})$ at the input sample points.

Texture mapping: A multi-texturing approach is used to embed appearance to the generated geometry. While recent approaches [23] offer high quality results, due to our real-time processing constraint a faster approach was opted for. Multi-texture mapping is applied to the generated 3D mesh, where each mesh triangle is assigned two textures from the multiple Kinects. These are then blended in a weighted manner, following the approach that is detailed in [24]. According to that method, the texture mapping weights depend on: a) the "viewing" angle of the captured surface, i.e. on the angle between the line-of-sight vector and the vertex normal, and b) the 2D distance of the 2D projection pixel from the foreground human's 2D silhouette.

2.3. Compression and networking

In the whole from-capturing-to-rendering TI chain, a strong interrelation exists between the 3D data format choice, the compression scheme, the transmission layer and the networking architecture. In the proposed multi-party TI framework, a server-based networking scheme is preferred over a peer-to-peer network, to offer scalability and centralized simulation capabilities. Furthermore, connections are not only limited to participants, as "spectators" can connect and observe the immersive environment scene along with the interacting participants. Each local TI station transmits the data to the server, where synchronization takes place and then the global state is transmitted to connected users.

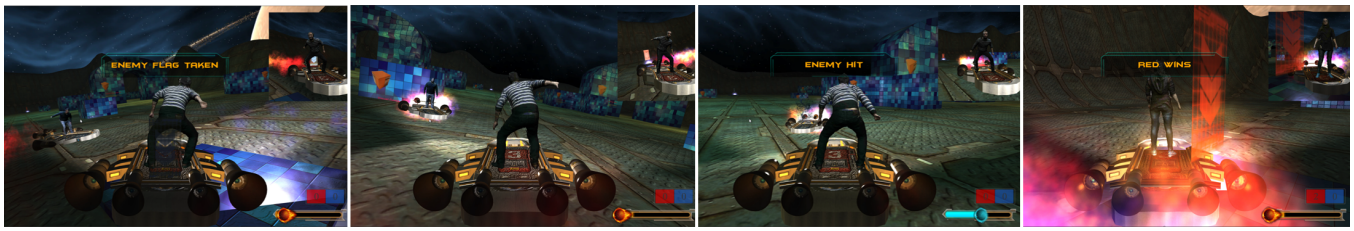

Fig. 3. Screenshots with the *SpaceWars* players in action. The dynamic 3D reconstructed representations of the players appear in the virtual environment, where they can interact with the environment and with each other.

With respect to the compression of heavy 3D reconstruction data, the geometry information (vertex positions, normals and attributes, as well as connectivity) is compressed via OpenCTM [25], as in [26]. This choice allows the platform to be scalable to network conditions [27]. A intra-frame static mesh codec, such as CTM, was selected since the reconstructed 3D meshes are "time-varying" meshes (i.e. with variable number of vertices and connectivity along frames), but only recently a few immature approaches for real-time inter-frame time-varying mesh compression have been developed [28]. Standard JPEG compression [16] is employed for textures, due to its simplicity and very-fast performance.

2.4. Immersion

Immersion in our platform is achieved through users' placement in, and interaction with, the shared virtual environment, as well as among the users themselves. The textured mesh representation is integrated seamlessly in most 3D graphics engines using standard graphics pipeline. Interactions are guided by the users' activity detection and analysis, offering a natural user interface, driven by the sensor's inherent skeletal tracking. The skeleton data structures are synchronized on the server side and are embedded in the platform's state message. Gesture recognition and posture analysis can then be implemented to trigger events and enable 3D navigation respectively. Through this natural body controlled interface, the immersive environment is augmented with realistic representations of interacting users, and presented in any device capable of standard graphics rendering.

3. SPACEWARS: A 3DTI APPLICATION

To demonstrate the proposed platform's capability in enabling remote interaction of real-time 3D reconstructed users, the platform was used to drive the development of a proof-of-concept application, an immersive multi-player sci-fi symmetrically designed 3D game titled *SpaceWars*: players ride on hoverboards, and compete against each other in a capture-the-flag setting, during which they can see their own and their opponent's appearance in real-time and full 3D inside the arena, as depicted in Fig. 3. A video can be found at `http://vcl.iti.gr/spacewars/` . Players navigate the arena using their body posture: a) bending the knees and flexing the torso (i.e. taking a ski posture) increases the hovering speed; b) leaning left or right, changes the hovering direction respectively. Additionally, players interact with each other: a) performing a throwing action, they can fire their weapon at their opponent; b) navigate away from their opponents fired projectiles; c) get destroyed when hit by their opponent's projectile. But also with the environment: a) hover over the opponent's flag to pick it up; b) navigate in the environment and use it to block the opponent's projectiles and view; c) return the opponent's flag to their own base to score a point.

The game was implemented in the Unity3D game engine. The RabbitMQ framework was used as the messaging layer, with each remote station sending its messages to the game server. The latter, synchronizes the game state and in turn sends a state message to each client. As in most internet-based multiplayer games, client side prediction was implemented to account for transfer latency. The employed centralized architecture allows for non-participant users to also connect to the platform as spectators. This "spectator mode" was realized in a two-fold implementation to showcase the transmedia nature of TI platforms and enhance the attractiveness of the game. Besides the traditional desktop application, in which the user can navigate the arena by using a standard keyboard, an Augmented Reality approach was implemented on a tablet. In this version of the spectator, the 3D virtual scene is registered to a target image featuring a distinctive pattern. Users are able to freely move around a physical copy of the target (printed on cardboard) and navigate the arena using the device's back camera as a physical viewing source. Therefore, the mixed-reality TI content can be observed virtually everywhere, from the tabletop to the palm of your hands. The entire platform was successfully demonstrated twice, during an open-day event at the authors' premises and a school hosting event, where non-expert users of all ages enjoyed the game.

4. CONCLUSION

A portable TI platform, integrating 3D reconstruction, motion capturing, activity detection and analysis, 3D compression and networking technologies to facilitate immersive interactions across remote sites, was introduced. Besides the intrinsic difficulty in reconstructing the appearance of users in real-time, high quality and in lightweight, transfer-friendly data formats, the additional requirement of ensuring interaction among the users and the virtual environment, results in complex and expensive solutions. This work addressed most of these challenges, while keeping the complexity and overall cost low and has been verified through a prototype TI game.

5. REFERENCES

[1] T. DeFanti, D. Sandin, M. Brown, D. Pape, J. Anstey, M. Bogucki, G. Dawe, A. Johnson, and T. Huang, "Technologies for virtual reality/tele-immersion applications: issues of research in image display and global networking," in *Frontiers of Human-Centered Comp., Online Communities and Virt. Environ.* 2001.

[2] R. Vasudevan, G. Kurillo, E. Lobaton, T. Bernardin, O. Kreylos, R. Bajcsy, and K. Nahrstedt, "High-quality visualization for geographically distributed 3-D teleimmersive applications," *IEEE Trans. on Multimedia*, vol. 13, no. 3, pp. 573–584, 2011.

[3] 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 *IEEE ISM*, 2010, pp. 220–227.

[4] Ch. Lin, P.Y. Sun, and F. Yu, "Space connection: a new 3D tele-immersion platform for web-based gesture-collaborative games and services," in *IEEE/ACM 4th Int. Workshop on Games and Soft. Engin.*, 2015.

[5] K. Nahrstedt, "3D tele-immersion for remote injury assessment," in *Proc. of Int. Workshop on Socially-aware multimedia*, 2012.

[6] G. Kurillo, R. Bajcsy, O. Kreylos, and R. Rodriguez, "Tele-immersive environment for remote medical collaboration," in *Medicine meets virtual reality*, 2009, vol. 17, pp. 148–150.

[7] Z. Yang, B. Yu, W. Wu, R. Diankov, and R. Bajscy, "Collaborative dancing in tele-immersive environment," in *Proc. of the 14th annual ACMMM*, 2006.

[8] J. Smith and F. Weingarten, *Research challenges for the next generation internet*, Comp. Res. Association, 1997.

[9] A. Smolic, "3D video and free viewpoint videofrom capture to display," *Pattern recognition*, vol. 44, 2011.

[10] A. Maimone and H. Fuchs, "Encumbrance-free telepresence system with real-time 3D capture and display using commodity depth cameras," in *10th IEEE Int. Symposium on Mixed and Augmented Reality*, 2011.

[11] D. Alexiadis, D. Zarpalas, and P. Daras, "Real-time, full 3-D reconstruction of moving foreground objects from multiple consumer depth cameras," *IEEE Trans. on Multimedia*, vol. 15, no. 2, pp. 339–358, 2013.

[12] D. Alexiadis, D. Zarpalas, and P. Daras, "Fast and smooth 3D reconstruction using multiple RGB-Depth sensors," in *IEEE VCIP*, 2014.

[13] Z. Yang, K. Nahrstedt, Y. Cui, B. Yu, J. Liang, S. Jung, and R. Bajcsy, "TEEVE: The next generation architecture for tele-immersive environments," in *7th IEEE ISM*, 2005.

[14] W. Wu, R. Rivas, A. Arefin, S. Shi, R. Sheppard, B. Bui, and K. Nahrstedt, "MobileTI: a portable tele-immersive system," in *Proc. of the 17th ACMMM*, 2009.

[15] A. Maimone and H. Fuchs, "Real-time volumetric 3D capture of room-sized scenes for telepresence," in *IEEE 3DTV-Conference*, 2012.

[16] G. Wallace, "The JPEG still picture compression standard," *Comm. of the ACM*, vol. 34, pp. 30–44, 1991.

[17] Y. Collet, "Lz4: Extremely fast compression algorithm," *code.google.com*, 2013.

[18] D. Lowe, "Object recognition from local scale-invariant features," in *Proc. of the 7th IEEE ICCV*, 1999, vol. 2.

[19] P. F. Alcantarilla, J. Nuevo, and A. Bartoli, "Fast explicit diffusion for accelerated features in nonlinear scale spaces," in *BMVC*, 2013.

[20] D. Kendall, "A survey of the statistical theory of shape," *Statistical Science*, pp. 87–99, 1989.

[21] M. Kazhdan, "Reconstruction of solid models from oriented point sets," in *3rd Eurographics SGP*, 2005.

[22] W. Lorensen and H. Cline, "Marching cubes: A high resolution 3D surface construction algorithm," in *ACM SIGGRAPH*, 1987, vol. 21, pp. 163–169.

[23] R. Pagés, D. Berjón, F. Morán, and N. García, "Seamless, static multitexturing of 3d meshes," *Computer Graphics Forum*, vol. 34, pp. 228–238, 2015.

[24] D. Alexiadis, D. Zarpalas, and P. Daras, "Real-time, realistic full-body 3D reconstruction and texture mapping from multiple kinects," in *Proc. IEEE IVMSP*, 2013.

[25] M. Geelnard, "OpenCTM, the open compressed triangle mesh file format," 2010.

[26] D. Alexiadis, A. Doumanoglou, D. Zarpalas, and P. Daras, "A case study for tele-immersion communication applications: From 3D capturing to rendering," in *IEEE VCIP*, 2014.

[27] S. Crowle, A. Doumanoglou, B. Poussard, M. Boniface, D. Zarpalas, and P. Daras, "Dynamic adaptive mesh streaming for real-time 3d teleimmersion," in *Proc. of the 20th Int. Conf. on 3D Web Technology*, 2015.

[28] A. Doumanoglou, D. Alexiadis, D. Zarpalas, and P. Daras, "Towards real-time and efficient compression of human time-varying-meshes," *IEEE Trans. on Circuits and Systems for Video Technology*, 2014.