RGB-D Camera Network Calibration and Streaming for 3D Telepresence in Large Environment

Po-Chang Su
Center for Visualization and Virtual Environment
University of Kentucky
Lexington, Kentucky 40506
Email: psu223@g.uky.edu

Ju Shen
Interactive Visual Media (IVDIA) Lab
Department of Computer Science
University of Dayton
Dayton, Ohio 45469
Email: jshen1@udayton.edu

Muhammad Usman Rafique
Center for Visualization and Virtual Environments
University of Kentucky
Lexington, Kentucky 40506
Email: usman.rafique@uky.edu

***Abstract*—With the advance of multimedia technology, traditional remote communication applications (e.g. video chat) may not satisfy the increasing demand for effective interactivity due to the lack of realistic experience of being physically present. Telepresence is a new form of communication that aims to strengthen the connection between people by creating more immersive and interactive environments. However existing telepresence systems often suffer from limited viewing space or failing to render realistic 3D contents on the display. In this paper, we present a novel system that can improve the interactive experience among remote users with dynamically generated virtual 3D environments. To support wide view rendering, an automated calibration system is developed for RGB-D camera network that is scalable to capture and reconstruct large environments. Our proposed client-server architecture is capable of transmitting and processing large volume of RGB-D data across the the network in real-time. Virtual 3D views are automatically synthesized by seamlessly fusing multiple video streams acquired by different cameras. Our experimental results demonstrate better visual effects that can be applied to existing telepresence systems for more motivating and engaging remote applications.**

***Keywords*—RGB-D Camera Networks, 3D Telepresence, Immersive Environment, Real-Time Rendering, Remote Communication**

I. Introduction

Current multimedia technologies are transforming the way we communicate and interact in our daily routines. Video chat is a typical example that allows people to have remote conversations and be able to see each other together with the surrounding environments simultaneously. While 2D video is efficient for remote conversation with the basic communication requirements, it may not be sufficient for specific activities which demand high engagement and interaction, such as virtual classroom [1], immersive gaming [2]. As an emerging technique, telepresence aims to integrate spatial immersion and virtual realty to create a perception of being physically present in a shared non-physical environment. To date, many telepresence systems are proposed to offer immersive and realistic experiences to end-to-end users. For example, a telepresence robot is designed to enable teachers to deliver motivating lectures remotely [3]; dynamic scene generation based on perspective views can make the co-space experience more engaging and absorbing [4]. To enhance the awareness of time and the real world, audio-visual data and other stimuli are often employed. In this paper, we focus on creating the visual perception that dynamically synthesizes realistic virtual environment from the first-person view in real-time.

Among existing telepresence systems, high-speed video transmission and real-time data visualization are crucial factors to provide immediate feedback [5]–[7]. In order to deliver rich information to users, depth sensors are often employed together with color cameras for real world scene acquisition, which provides fertile source for the 3D environment generation. However, existing depth cameras often suffer from limited field of view and low resolution issues. To mitigate this limitation, Simultaneous Localization and Mapping (SLAM) techniques [8]–[11] have been utilized by moving the camera around a capture space to fuse sufficient data. This solution works well when it is used to scan a static object, such as a chair, or a table. However, for a highly dynamic scene, SLAM seems inadequate to fulfill this task. As such, a network of multiple cameras can be employed to record the capture space simultaneously and broaden the capture views. There are already a large body of works using color camera networks for various types of vision processing [12]–[14]. With the recent success of low-cost commodity RGB-D cameras, such as Kinect, the integration of color and depth information provide a new way for scene acquisition and bring many interesting applications, such as virtual mirror rendering, body pose tracking and understanding [15]–[17], Among these works, different types of RGB-D camera networks are employed, which provide richer information and a larger capturing space.

In this paper, we present a real-time scene acquisition and reconstruction system that enables people to communicate and interact in a more motivating and engaging environment. Our major technical contribution is the availability to acquire and render wide-viewed real world with dynamic perspectives to end users. A network of RGB-D cameras is automatically calibrated that allows multiple sensed streams to be simultaneously aligned and fused into a global space. Our system can bring multiple users from remote places into a shared environment. A series of experiments are performed to verify the proposed system and demonstrate the potential benefits to existing telepresence systems with better more motivating and

978-1-5090-6549-3/17 $31.00 © 2017 IEEE
DOI 10.1109/BigMM.2017.83

engaging effects.

The remaining of the paper is organized as follows: Section II reviews recent literature on camera network calibration and telepresence system. In Section III, we describe our proposed framework in details, including camera network calibration with a spherical object and stream fusion based on the proposed client-server architecture for data processing. The implementation details and its evaluation of calibration and rendering results can be found in Section IV. The conclusion can be found in Section V.

II. Related Work

Extrinsics estimation is a fundamental step for building a camera network that captures large environments and merges multiple camera views into the same coordinate. To establish correspondences across the camera views, a reference object visible to different cameras is often used to perform camera network calibration. While there are a number of calibration objects have been used to calibrate multiple cameras, most of them are not suitable for the case that cameras are sparsely placed [18]–[22]. The authors in [23] waved a laser pointer surrounded by multiple cameras can easily identify the correspondences. However, for depth sensor, it is difficult to obtain the accurate depth value on LED tip due to uncertainty of the depth measurement on the depth discontinuity region. Kim et al. used a customized 3D reference object which consists of 18 2-D planes to cope with widespread camera network structure [24]. Nevertheless, this method needs at least 2 planes visible from the cameras, resulting in the constraint on arbitrarily positioned cameras. Thus it can be seen that small-overlapping regions occur when multiple cameras are widely spaced. They significantly limit the use of view transformation for alignment and simple calibration objects.

In fact, the topic of calibration of wide-area networks of sparsely-spaced cameras has been intensively studied [25]–[27]. Ly et al. utilized image of lines to improve the calibration results for multiple cameras with only partially overlapping fields of view [25]. In [26], an active self-calibration of a multi-camera system scheme was proposed to solve the problem of non-overlapping views and occlusion by automatically rotating and zooming each camera. A probabilistic model was used to find the appropriate relative pose during extrinsic calibration. Moral et al. used large planar scenes such as the floor or ceiling to calibrate camera with disparate views [27]. The use of pan-tilt-zoom cameras or special scene features limits the types of applications where these techniques can be deployed. In [28], the authors used a spherical object and estimated the location of the sphere center for extrinsic calibration of multiple RGB-D cameras. However, the sphere detection was not very robust due to the manual effort required in removing outliers. Furthermore, as the cameras were not time-synchronized, the technique was labor intensive as the sphere needed to be physically moved and affixed to different locations in order to capture enough data for calibration.

While current telepresence systems have given users with improved 3D interaction experiences, there is still room to improve with better visual effects that deliver a more realistic sense of face-to-face interaction. The works in [29]–[31] only provided a 3D image with the fixed viewpoint on 2D display without the ability of arbitrarily changing the perspective view. Such constrain loses the flexibility of fully interacting with partners on the other side. On the other hand, some of the telepresence works only put the users in virtual environments instead of the scenes of where they are present in real-world [1], [32]–[34]. As a result, it decreases authenticity of interaction due to unreal surrounding environments. To implement a telepresence system that can render real 3D scenes, Maimone et al. built a RGB-D camera network for capturing and reconstructing 3D data [4]. The rendering on the 3D environment changes based on the viewer's perspective. However, their system only reconstructed the partial region of the environment and users who were at different places can not be put into the same environment. To achieve high-quality 3D telepresence, the works in [31], [35], [36] either used a cluster of cameras or 3D projectors for their environmental setup. However, such hardware installment is expensive and not feasible for general users.

III. Proposed Method

Our proposed framework for real-time 3D telepresence is shown in Figure 1. We first calibrate a local camera network to provide users with large reconstructed environments. To provide scalability, each RGB-D camera is connected to a computer served as a client to collect calibrated RGB-D images and detect correspondences for extrinsic calibration. The camera view captured from a remote place is then merged into the local scene based on 3D geometry. Next, the server receives the data from the clients and then renders 3D scenes. Each client in the camera network is time synchronized with the server using Network Time Protocol (NTP) [37]. We will describe the details of each step in the following sections.

A. Local Camera Network Calibration

We configure a camera network that can cover the wide area in an indoor environment. As shown in Figure 2, cameras are sparsely placed with different orientations. The extrinsics of each camera are estimated during the calibration stage and then applied to transform 3D points from the local coordinate to a global coordinate. Our goal is to estimate m extrinsic matrices $\{\boldsymbol{P}_1, \boldsymbol{P}_2, .., \boldsymbol{P}_j, .., \boldsymbol{P}_m\}$ for a network of m RGB-D cameras $\{C_1, C_2, .., C_j, .., C_m\}$. Here the bold format is used for vectors and matrics. The coordinate of an arbitrarily-chosen reference camera C_1 is treated as the world coordinate system. The extrinsic matrix $\boldsymbol{P}_j$, which represents the extrinsics of the j^{th} camera C_j in the camera network, is determined by a translation vector $\mathbf{T}_j = [t_x^{(j)} \quad t_y^{(j)} \quad t_z^{(j)}]^T$ between the reference camera and an arbitrarily-chosen camera and a rotation matrix $\mathbf{R}_j$ parameterized by the three rotation angles $\theta_x^{(j)}, \theta_y^{(j)}$ and $\theta_z^{(j)}$:

$$\boldsymbol{P}_j = \begin{bmatrix} \boldsymbol{R}_j & \boldsymbol{T}_j \\ 0 & 1 \end{bmatrix} \tag{1}$$

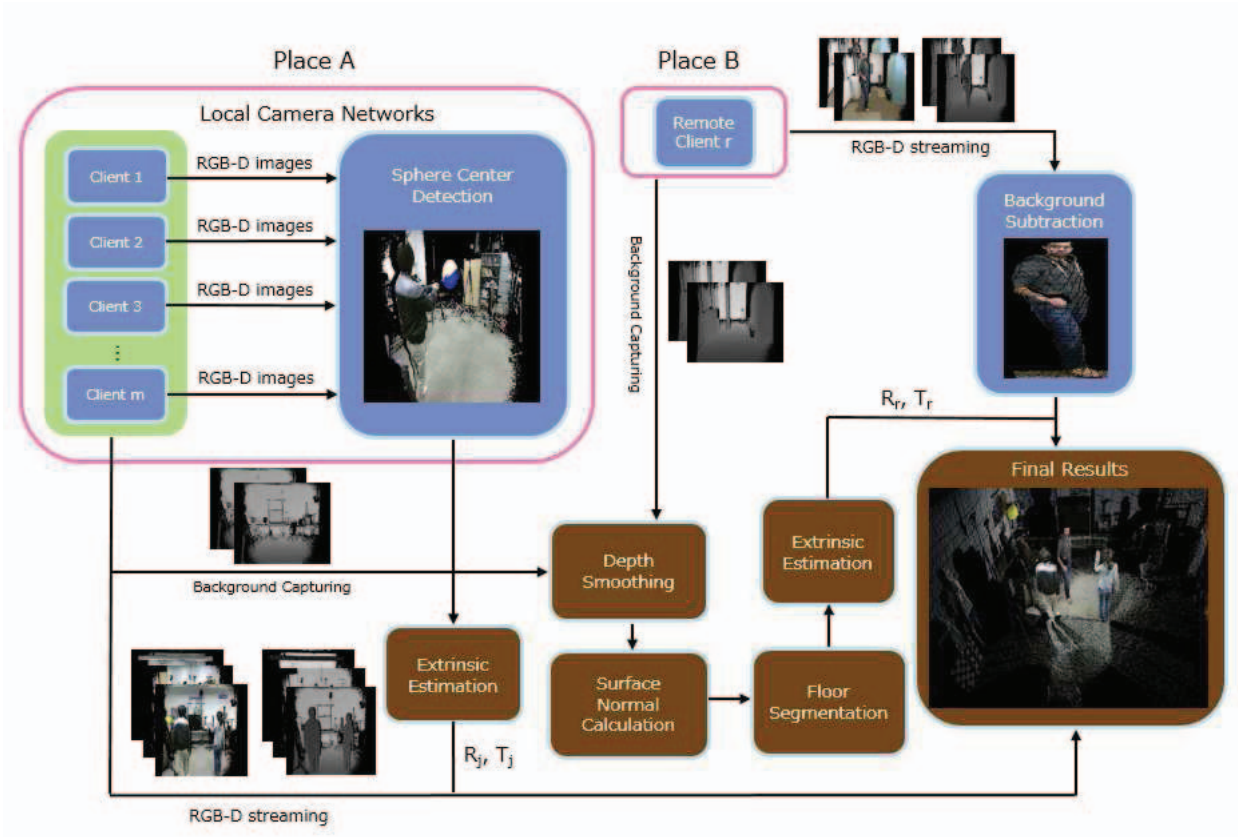


Fig. 1. Overview of our 3D telepresence framework by the client-server architecture. The task in the brown box is performed on server side.

To overcome searching common features in wide viewing angle between cameras, our calibration method is based on [38]. A sphere ball, which is well-suited for our camera network placement, is used other than a traditional reference object like checkerboard. The sphere center can be inferred and put into the correspondence list as long as the small overlapping region is available between a camera pair. Given a sequence of captured RGB-D images, a simple background substraction method initially extracts out foreground region by checking depth differences with a pre-built background mask. To further detect the silhouette of a sphere, the sphere ball is painted in yellow so as to be easily distinguished in the histogram of the distribution in HSV color space. The qualified pixels within the sphere region are then transformed to 3D points that serve as inputs for sphere fitting algorithm [39]. As such, the estimated sphere center of each RGB-D frame is aggregated into a set of sphere centers for geometrical alignment between cameras.

After sending the detected sphere centers from each client to the server, pairwise camera calibration is performed to find the geometrical relation between the reference C_1 and all other cameras based on location of sphere center. A sequence of the detected sphere centers are denoted as $\{\boldsymbol{c}_1, \boldsymbol{c}_2, .., \boldsymbol{c}_i, .., \boldsymbol{c}_n\}$, where $\boldsymbol{c}_i$ represents the 3D coordinate of an estimated sphere center at the i^{th} frame in the RGB-D image sequence: $\{\boldsymbol{I}_{c_1}, \boldsymbol{I}_{c_2}, .., \boldsymbol{I}_{c_i}, .., \boldsymbol{I}_{c_n}, \ \boldsymbol{I}_{d_1}, \boldsymbol{I}_{d_2}, .., \boldsymbol{I}_{d_i}, .., \boldsymbol{I}_{d_n}\}$. Then, a system of linear equations are constructed to provide the initial guess of the transformation matrix $\boldsymbol{P}_j$ between each camera pair [40]. If there is no overlapping region between C_1 and C_j, the camera C_j will first search the rest of the cameras that share partial field of view. The camera that has the lowest error distance with C_j's correspondences will be selected. Therefore, the estimation on indirect transformation between C_1 and C_j can be acquired. Finally, to optimize extrinsics for each camera, bundle adjustment algorithm [41] is utilized to minimize the sum of distance errors from the established correspondences pairs in the camera network. As our goal is to refine camera pose in 3D space, the minimization formula for bundle adjustment is as follows:

$$\min_{\boldsymbol{P}_j^{-1}, \widetilde{\boldsymbol{C}}} \sum_{j=1}^{m} v_j d(f(\boldsymbol{P}_j^{-1}, \widetilde{\boldsymbol{C}}), \boldsymbol{C}_j)^2 \tag{2}$$

where variable $\widetilde{\boldsymbol{C}}$ is a $4 \times n$ matrix that stores the 3D coordinate of the sphere center trajectory in the world coordinate, d is the Euclidean distance $||f(\boldsymbol{P}_j^{-1}, \widetilde{\boldsymbol{C}}) - \boldsymbol{C}_j||^2$ between estimated 3D points and measured 3D points $\boldsymbol{C}_j$. The variable $v_j \in \{0, 1\}$ indicates whether the point is visible by camera C_j. As an initial input, we simply transfer all the extracted 3D sphere

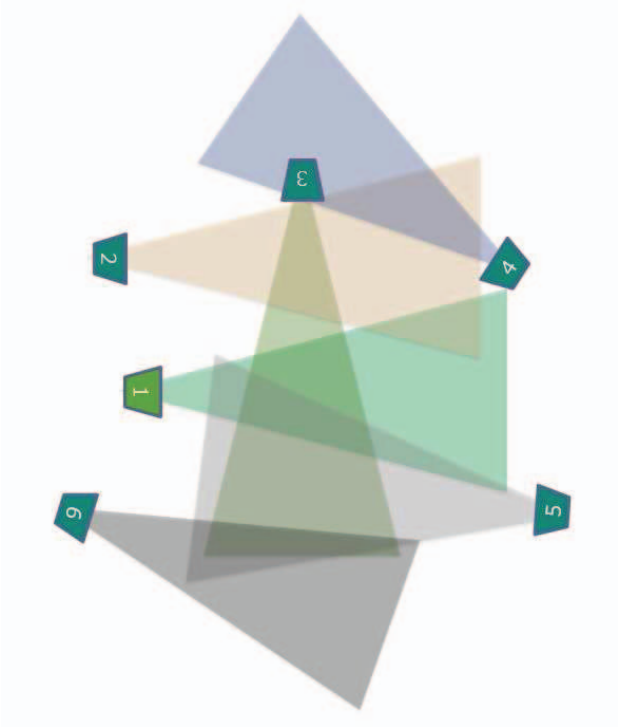


Fig. 2. Non-evenly distributed structure of a camera network

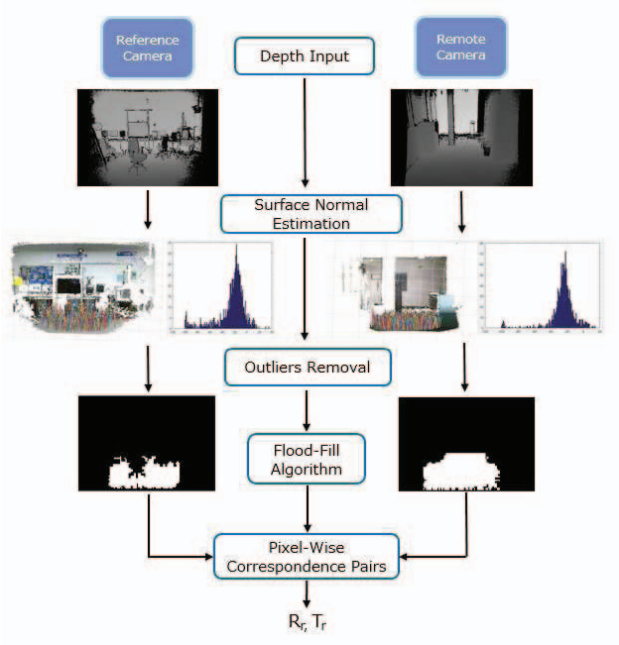


Fig. 3. Workflow of planar floor alignment

center trajectories from each local camera coordinate to the world coordinate and take the average of them. The averaged 3D points are then transformed from the global coordinate back to the local camera coordinates based on $\boldsymbol{R}_j$ and $\boldsymbol{T}_j$ extracted from extrinsic matrix $\boldsymbol{P}_j$:

$$f(\boldsymbol{P}_j^{-1}, \widetilde{\boldsymbol{C}}) = \begin{bmatrix} \boldsymbol{R}_j & \boldsymbol{T}_j \\ 0 & 1 \end{bmatrix}^{-1} \widetilde{\boldsymbol{C}} \tag{3}$$

The optimized extrinsics $\{\boldsymbol{P}_1^*, \boldsymbol{P}_2^*, ..., \boldsymbol{P}_m^*\}$ are then applied to the online 3D renderings.

B. Remote Camera Fusion

After completing the local camera network calibration, the remote camera view is then merged into the reference camera's field of view to virtually teleport users to the local environment. Our remote camera fusion relies on 3D information of planar floors. To extract the floor from a background scene, first, depth images are smoothed by applying bilateral filter [42], which can effectively interpolate missing values while preserving depth discontinuities:

$$D(p) = \frac{1}{C_w} \sum_{q \in L} e^{\left(\frac{-\|p-q\|^2}{2\sigma_s^2} + \frac{-|R(p)-R(q)|^2}{2\sigma_r^2}\right)} R(q) \tag{4}$$

where q is the neighboring pixel in the filter window L based on the central pixel p, σ_s is the Gaussian spatial weight, σ_r is the Gaussian depth weight, and C_w is the normalization of the exponential term. The radius of the filter window is denoted as l, which means the size of window L can be expressed as $(2l+1) \times (2l+1)$. σ_s is determined by the Euclidean distance between p and q on the 2D image plane. σ_r is determined by the similarity of the depth values between $R(p)$ and $R(q)$.

Next, to obtain the relative transformation between the reference camera and the remote camera, we detect the co-existing pixel locations of the floor in the depth images and then calculate their corresponding 3D points for point clouds alignment. We assume the floor occupies most of the lower portion in an image. A statistical classification on 3D points' surface normal can extract the floor from the static background. The surface normal of each 3D point is calculated by depth value $D(p)$. More specifically, for each pixel and its neighboring pixels $p_1 = (u, v)$, $p_2 = (u+1, v)$ and $p_3 = (u, v+1)$ in the depth image $\{p_1, p_2, p_3\} \in \boldsymbol{I}_{d_i}$, we can obtain the normal vector N_s by taking the cross product of the vector $L_1 = V(p_2) - V(p_1)$ and $L_2 = V(p_3) - V(p_1)$, where u and v are the column and row indices in the $\boldsymbol{I}_{d_i}$, $V(\cdot)$ is the 3D point estimated by back-projecting 2D point to 3D space given the 3×3 camera intrinsic K_c:

$$V(p) = D(p) \left(K_c^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \right) \tag{5}$$

To initially segment out the floor, we take the median of all the estimated surface normals N_a and empirically define a threshold ϵ to filter out the outliers. The segmentation result is interpreted into the binary image. The flood-fill algorithm can search the largest connected component Q in the binary image [43]. Based on the assumption that the floor dominates at the lower portion of the image, we can build the correspondence pairs between the reference camera Q_l

Fig. 4. Local camera network calibration using a spherical object.

and the remote camera Q_r pixel-wise. Same as we estimate extrinsics for the camera network, to align the floors at both places, the rotation R_r and the translation T_r can be obtained by minimizing the distance difference of the point-pairs [40]. The workflow of our floor detection algorithm is shown in Figure 3.

To fuse a remote user into the local scene, a background mask is trained to extract the user's body for online 3D rendering. A set of depth images are first captured, and pixels that don't have valid depth measurement are eliminated. We then take the average of each qualified pixel's depth value among the depth images. As such, for each pixel the gaussian distribution of measured depth value with standard deviation σ_d is built. If input of the pixel's depth is out of the range $2\sigma_d$ based on the averaged depth value μ_d, the pixel is considered as foreground.

IV. Experiments

We use Microsoft Kinect V2 cameras to capture color and depth images in our experiments. The local camera network consists of 5 cameras non-evenly distributed in a $10.2\text{m} \times 6.4\text{m}$ space where background and dynamic foreground objects within the coverage region can be reconstructed. Another camera for telepresence is placed at different places. Figure 4 is our camera network configuration calibrated by a spherical object. A client-server architecture is built for data collection from the cameras. Each camera is connected to a separate computer and send aligned color and depth image data with 960×540 resolution to a server during the calibration stage and online 3D environment rendering. Local and remote users can view the telepresence result with an arbitrarily virtual viewpoint through share screen software. To ensure the server receives the accurate corresponding frames sent by each camera, we set up a local NTP time server to synchronize all computers [37]. The result showed that system time for capturing each frame among all computers are within 4 ms after synchronizing with the local time server.

Our system is implemented by C/C++. We use OpenCV library to perform image processing and mathematical calculation for 3D point clouds. For visualization, we use OpenGL library to render 3D scenes. To achieve real-time performance, the hardware setting for the server is Intel Core (TM) i7-5820k CPU at 3.30 GHz and 32GB of RAM with the powerful GPU GTX-1080 to accelerate rendering speed. For each client, the hardware setting is Intel Core (TM) i7-4770s CPU at 3.1 GHz and 16GB of RAM.

A. Client-Server Data Flow Evaluation

To ensure data sending from client to server is real-time for 3D rendering, we evaluate data transmission rate on the client-server architecture. Each RGB-D camera is connected to a individual computer which compresses the color and geometry data and sends them to the server. The total bandwidth for each frame is about 1.02MB, with 0.96MB for geometry data and 0.06MB for color data. Therefore, the total data bandwidth requirement is about 6.1MB per frame or 1,464 Mbps at 30fps under 6 RGB-D camera. With the prevalence of 40/100 Gigabit Ethernet or even with the latest 802.11ad wireless standard, the server can receive and render 3D points in real-time without having any bottleneck issue.

B. Error Analysis on Camera Network Calibration

To measure alignment error in physical distance unit, the results of our camera network calibration are quantitatively evaluated in this section. After the calibration procedure is done, as described in Section III, we wave the sphere ball to capture another 5 RGB-D image sequences from each camera for alignment error analysis. The sphere centers are detected and transformed to the same coordinate system by applying previously estimated extrinsics. We use root mean square error (RMSE) to measure distance difference of the detected sphere centers between the cameras in the reference coordinate. The results in Table I showed that each camera pair only had about averaged 3.32cm RMSE. The relative error was 0.58% in the reconstructed environment. For remote camera fusion, we randomly chosen three different camera poses at the different places. The results showed that RMSE of the corresponding pairs built by the detected floors between the reference camera and the remote camera had average of 2.721cm. We also inspect whether the surface normal of two floors pointing to the same direction in the reference coordinate. The 3D points from the corresponding pairs are fitted into a plane for both cameras. The angle difference between the surface normals was within 0.023 degrees.

C. Reconstructed Scenes for 3D Telepresence

In this section, we evaluate quality of the rendering results on our 3D telepresence system. We again carry the sphere ball in the captured area and examine alignment accuracy of the reconstructed sphere and environment between cameras. Figure 5 is the top-down view of our 3D rendering result with each individual camera field of view encoded by different color at the bottom. We can see each camera view is accurately merged into a large reconstructed scene from visual standpoint. The sphere is well-aligned without any distortion. Next, we

TABLE I
ALIGNMENT ERROR ANALYSIS BETWEEN CAMERAS BY RMSE

Sequence	RMSE (cm)	# of frames
#1	3.21	260
#2	3.19	277
#3	3.35	334
#4	3.79	329
#5	3.06	318

examine our 3D telepresence results of fusing a user into the local place. We randomly select three different places for telepresence experiments. In Figure 6, the remote user is naturally fused into the reconstructed environment and can interact with the local users smoothly. For each case in Figure 6, the remote camera view and each local camera views are attached at upper right and the bottom, respectively. The backside of the remote user in the reconstructed scene is missing due to only using one single camera at the remote places in our experiments. This issue can be addressed by setting up two or multiple cameras at the remote place.

V. CONCLUSION

In this paper, we have presented a framework for 3D telepresence by using a calibrated RGB-D camera network that is scalable to capture and reconstruct an arbitrary large environment. The experimental results demonstrate our calibration methods can accurately and efficiently combine multi-views into a unified coordinate system. Furthermore, our proposed client-server architecture can transmit and process large volumes of RGB-D data in real-time. This would benefit many telepresence systems towards the goal to create a perception of physically present in a shared non-physical environment. Despite its potential impacts, admittedly, our proposed system is still in the early stage that still needs further improvement in several aspects. For example, the quality of the rendered results has not reached the satisfactory level for real applications yet. The virtual views are rendered based on combined 3D point cloud rather than mesh surface. Also, more investigation will be conducted on how the scene acquisition and reconstruction framework can be effectively integrated with remote user interaction. As our future work, we plan to investigate optimal strategies to improve the rendering quality, especially on those objects that require higher fidelity and details, such as human face, or thin objects. Also, more specific attentions should be paid to reflective surfaces or transparent materials that often cause depth sensors fail to measure the correct depth distances.

REFERENCES

[1] X. Lu, J. Shen, S. Perugini, and J. Yang, "An immersive telepresence system using rgb-d sensors and head mounted display," in *2015 IEEE International Symposium on Multimedia (ISM)*. IEEE, 2015, pp. 453–458.

[2] 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*, Dec 2010, pp. 220–227.

[3] O.-H. Kwon, S.-Y. Koo, Y.-G. Kim, and D.-S. Kwon, "Telepresence robot system for english tutoring," in *Advanced Robotics and its Social Impacts (ARSO), 2010 IEEE Workshop on*. IEEE, 2010, pp. 152–155.

[4] A. Maimone and H. Fuchs, "Encumbrance-free telepresence system with real-time 3d capture and display using commodity depth cameras," in *Proceedings of the 2011 10th IEEE International Symposium on Mixed and Augmented Reality*. 2011, pp. 137–146, IEEE.

[5] T. Blum, V. Kleeberger, C. Bichlmeier, and N. Navab, "mirracle: An augmented reality magic mirror system for anatomy education," in *Virtual Reality Short Papers and Posters (VRW), 2012 IEEE*. IEEE, 2012, pp. 115–116.

[6] C. Kamphuis, E. Barsom, M. Schijven, and N. Christoph, "Augmented reality in medical education?," *Perspectives on medical education*, pp. 300–311, 2014.

[7] P. Eisert, P. Fechteler, and J. Rurainsky, "3-D Tracking of Shoes for Virtual Mirror Applications," in *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, 2008, CVPR 2008.

[8] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2007.

[9] 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 *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*. ACM, 2011, pp. 559–568.

[10] C. Kerl, J. Sturm, and D. Cremers, "Dense visual slam for rgb-d cameras," in *Proc. of the Int. Conf. on Intelligent Robot Systems (IROS)*, 2013.

[11] P. C. Su, J. Shen, and S. c. S. Cheung, "A robust rgb-d slam system for 3d environment with planar surfaces," in *2013 IEEE International Conference on Image Processing*, 2013, pp. 275–279.

[12] F. Kahlesz, C. Lilge, and R. Klein, "Easy-to-use calibration of multiple-camera setups," in *Workshop on Camera Calibration Methods for Computer Vision Systems (CCMVS2007)*, 2007.

[13] J. Puwein, R. Ziegler, J. Vogel, and M. Pollefeys, "Robust multi-view camera calibration for wide-baseline camera networks," in *Applications of Computer Vision (WACV), 2011 IEEE Workshop on*. IEEE, 2011, pp. 321–328.

[14] Kuo T., Ni Z., Sunderrajan S., and Manjunath B.S., "Calibrating a wide-area camera network with non-overlapping views using mobile devices," in *ACM Transactions on Sensor Networks (TOSN)*, 2014, vol. 10.

[15] J. Shen, P. C. Su, S. c. S. Cheung, and J. Zhao, "Virtual mirror rendering with stationary rgb-d cameras and stored 3-d background," *IEEE Transactions on Image Processing*, vol. 22, no. 9, pp. 3433–3448, 2013.

[16] A. T. Tran, K. Harada, et al., "Depth-aided tracking multiple objects under occlusion," *Journal of Signal and Information Processing*, p. 299, 2013.

[17] Z. Ma and E. Wu, "Real-time and robust hand tracking with a single depth camera," *The Visual Computer*, pp. 1133–1144, 2014.

[18] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[19] L. V. Gool, T. Tuytelaars, and C. Strecha, "Dense matching of multiple wide-baseline views," in *Proceedings Ninth IEEE International Conference on Computer Vision*. IEEE, 2003, pp. 1194–1201.

[20] J. Barreto and K. Daniilidis, "Wide area multiple camera calibration and estimation of radial distortion," in *Proceedings of the 5th Workshop on Omnidirectional Vision, Camera Networks and Non-Classical Cameras, Prague, Czech Republic*, 2004, p. 64.

[21] R. K. Kumar, A. Ilie, J.-M. Frahm, and M. Pollefeys, "Simple calibration of non-overlapping cameras with a mirror," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, 2008, pp. 1–7.

[22] C. Kuster, T. Popa, C. Zach, C. Gotsman, M. H. Gross, P. Eisert, J. Hornegger, and K. Polthier, "Freecam: A hybrid camera system for interactive free-viewpoint video.," in *VMV*, 2011, pp. 17–24.

[23] T. Svoboda, D. Martinec, and T. Pajdla, "A convenient multicamera self-calibration for virtual environments," *Presence: Teleoper. Virtual Environ.*, pp. 407–422, 2005.

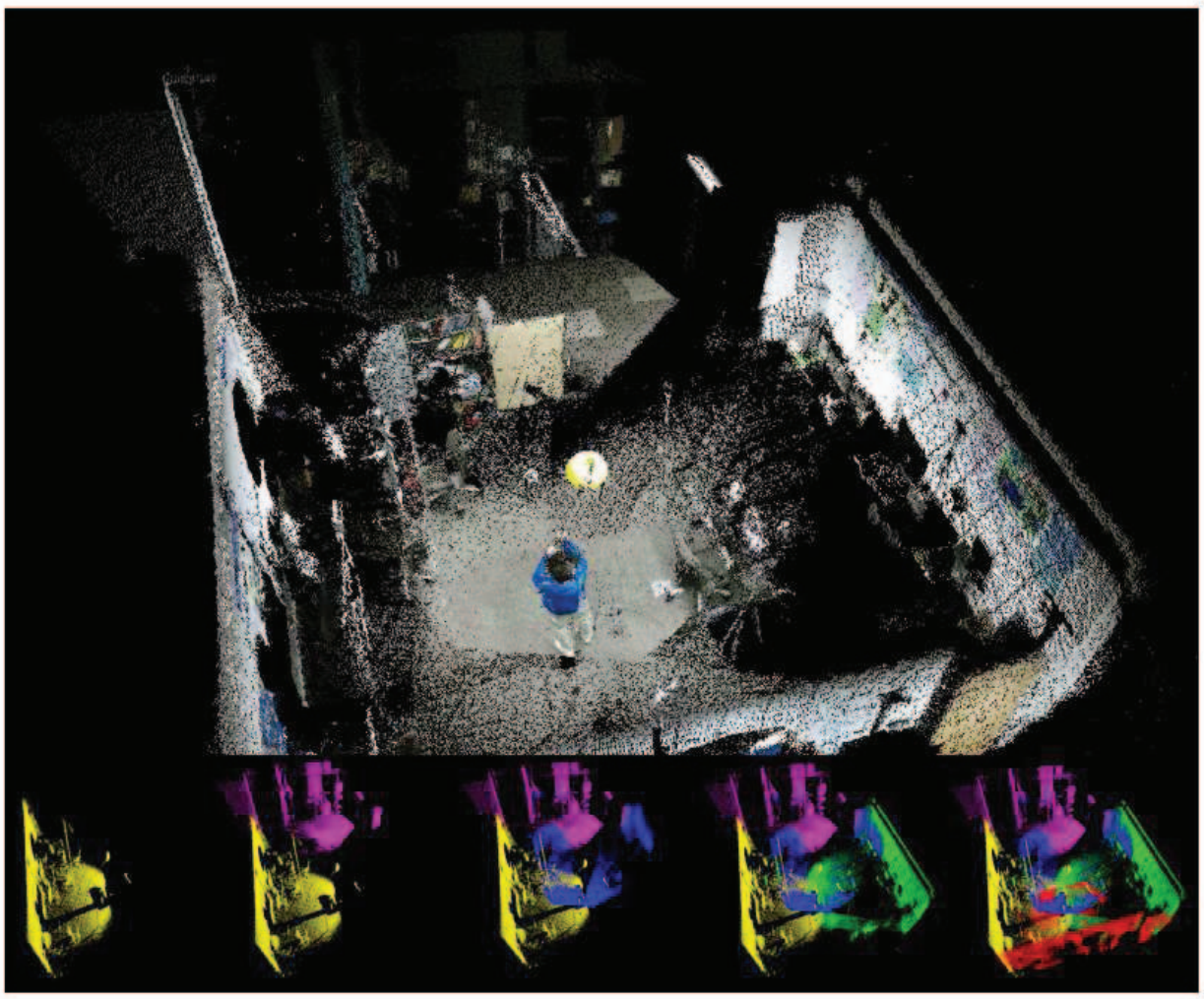


Fig. 5. The rendering result of the reconstructed environment from a virtual viewpoint. Each individual camera view is merged into the reference coordinate.

[24] J.-H. Kim and B.-K. Koo, "Convenient calibration method for unsynchronized camera networks using an inaccurate small reference object," *Optics express*, pp. 25292–25310, 2012.

[25] D. S. Ly, C. Demonceaux, P. Vasseur, and C. Pegard, "Extrinsic calibration of heterogeneous cameras by line images," *Machine Vision and Applications*, pp. 1601–1614, 2014.

[26] M. Bruckner, F. Bajramovic, and J. Denzler, "Intrinsic and extrinsic active self-calibration of multi-camera systems," *Mach. Vis. Appl.*, pp. 389–403, 2014.

[27] E. Fernandez-Moral, J. Gonzalez-Jimenez, P. Rives, and V. Arevalo, "Extrinsic calibration of a set of range cameras in 5 seconds without pattern," in *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, 2014, pp. 429–435.

[28] M. Ruan and D. Huber, "Calibration of 3d sensors using a spherical target," in *3D Vision (3DV), 2014 2nd International Conference on*, 2014, pp. 187–193.

[29] W. Matusik and H. Pfister, "3d tv: a scalable system for real-time acquisition, transmission, and autostereoscopic display of dynamic scenes," *ACM Transactions on Graphics (TOG)*, pp. 814–824, 2004.

[30] J. Edelmann, P. Gerjets, P. Mock, A. Schilling, and W. Strasser, "Face2facea system for multi-touch collaboration with telepresence," in *Emerging Signal Processing Applications (ESPA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 159–162.

[31] M. Gross, S. Würmlin, M. Naef, E. Lamboray, C. Spagno, A. Kunz, E. Koller-Meier, T. Svoboda, L. Van Gool, S. Lang, et al., "blue-c: a spatially immersive display and 3d video portal for telepresence," in *ACM Transactions on Graphics (TOG)*. ACM, 2003, pp. 819–827.

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

[33] 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 Transactions on Multimedia*, pp. 573–584, 2011.

[34] D. S. Alexiadis, D. Zarpalas, and P. Daras, "Real-time, realistic full-body 3d reconstruction and texture mapping from multiple kinects," in *IVMSP Workshop, 2013 IEEE 11th*. IEEE, 2013, pp. 1–4.

[35] P.-A. Blanche, A. Bablumian, R. Voorakaranam, C. Christenson, W. Lin, T. Gu, D. Flores, P. Wang, W.-Y. Hsieh, M. Kathaperumal, et al., "Holographic three-dimensional telepresence using large-area photorefractive polymer," *Nature*, pp. 80–83, 2010.

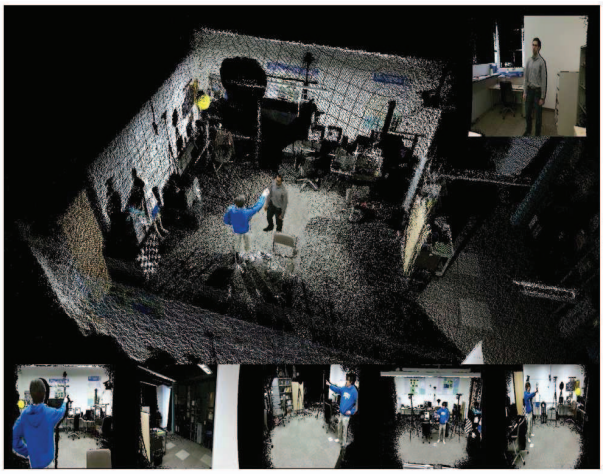
[36] T. Balogh and P. T. Kovács, "Real-time 3d light field transmission," in *SPIE Photonics Europe*. International Society for Optics and Photonics, 2010, pp. 772406–772406.

[37] NTP, "The network time protocol," http://www.ntp.org, 2014.

[38] J. Shen, W. Xu, Y. Luo, P. C. Su, and S. c. S. Cheung, "Extrinsic calibration for wide-baseline rgb-d camera network," in *2014 IEEE*

(a)

(b)



(c)

Fig. 6. The rendering results of 3D telepresence (a)(b)(c): The remote camera view is at upper right and each local camera view is at the bottom.

16th International Workshop on Multimedia Signal Processing (MMSP), 2014, pp. 1–6.

[39] V. Pratt, "Direct least-squares fitting of algebraic surfaces," in *ACM SIGGRAPH computer graphics*. ACM, 1987, pp. 145–152.

[40] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-squares fitting of two 3-d point sets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 698–700, 1987.

[41] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon, "Bundle adjustment - a modern synthesis," in *Vision Algorithms: Theory and Practice*, 1999.

[42] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Proceedings of the Sixth International Conference on Computer Vision*, Washington, DC, USA, 1998, pp. 839–, IEEE Computer Society.

[43] J. Dunlap, "Queue-linear flood fill: A fast flood fill algorithm," http://www.codeproject.com/KB/GDI-plus/queuelinearfloodfill.aspx, 2006.