Hindawi
Journal of Sensors
Volume 2019, Article ID 6295956, 17 pages
https://doi.org/10.1155/2019/6295956

Research Article

Omnidirectional 3D Point Clouds Using Dual Kinect Sensors

Seokmin Yun, Jaewon Choi, and Chee Sun Won

Department of Electronics and Electrical Engineering, Dongguk University, Dongguk University-Seoul, 30 Pildongro 1-gil, Junggu, Seoul 04620, Republic of Korea

Correspondence should be addressed to Chee Sun Won; cswon@dongguk.edu

Received 14 August 2018; Revised 16 October 2018; Accepted 29 October 2018; Published 17 January 2019

Academic Editor: Antonio Lazaro

This paper proposes a registration method for two sets of point clouds obtained from dual Kinect V2 sensors, which are facing each other to capture omnidirectional 3D data of the objects located in between the two sensors. Our approach aims at achieving a handy registration without the calibration-assisting devices such as the checker board. Therefore, it is suitable in portable camera setting environments with frequent relocations. The basic idea of the proposed registration method is to exploit the skeleton information of the human body provided by the two Kinect V2 sensors. That is, a set of correspondence pairs in skeleton joints of human body detected by Kinect V2 sensors is used to determine the calibration matrices, then Iterative Closest Point (ICP) algorithm is adopted for finely tuning the calibration parameters. The performance of the proposed method is evaluated by constructing 3D point clouds for human bodies and by making geometric measurements for cylindrical testing objects.

1. Introduction

LIDAR is a 360° omnidirectional scanning device that can estimate the distance by measuring the reflected laser pulses from the target object and is widely used in autonomous vehicles [1, 2]. The high price of the LIDAR equipment, however, is a major obstacle in adopting the laser sensor for small-scale 3D object scanning applications. Alternatively, 3D point cloud can be obtained by measuring the depth from the sensor to the target object by using a stereo camera or an IR sensor. Unlike the stereo camera, the IR sensor can be also used for the distance measurements even with no visible light. Although the depth measurements through the IR sensor do not have the same accuracy and distance coverage compared to the laser scanners, its low price and easy availability are definitely the merits.

Kinect sensor developed for the gesture recognition for indoor video games has the IR sensor, which can be used to measure the depth and to construct 3D point clouds. The first version of Kinect generates a color image of 640×480 resolution and a depth map of 320×240 with 30 frames per second [3]. The second version of it, Kinect V2, adopts the Time of Flights (ToF) method instead of the *Light Coding™* and produces a 512×424 depth map as well as a 1920×1080 full HD size color image [3]. The ToF scanning method used in Kinect V2 measures the depth by calculating the time that the infrared light is emitted and then reentered the camera sensor. Since the FOV (Field Of View) of Kinect V2 is horizontal 70 degrees, only the frontal sides of the objects can be scanned by a single Kinect sensor. Therefore, it is necessary to use multiple Kinect sensors to expand the FOV [4, 5]. On the other hand, to construct an omnidirectional 3D point clouds, multiple Kinect sensors should be installed around the target object. Then, the calibration among Kinect sensors to represent the scanned data onto a common coordinate system is necessary.

In this paper, we are interested in constructing 3D point clouds with Kinect V2 sensors. In particular, our goal is to propose a simple registration method for 3D point clouds in a dynamic relocation environment with only two Kinect sensors. Obviously, to use the smallest number of RGBD sensors to scan an object in 360°, it is necessary to locate them such that they are facing each other. In this case, however, the overlapped scanned data required for the correspondence-based registration methods may not be sufficient, which makes it difficult to employ the conventional calibration approaches. To solve this problem, a specially designed calibration device such as a spherical object with a rod [6] or a double-sided checker board [7] can be used. In this paper, a novel calibration method without using any calibration-assisting device is proposed. Specifically, the skeleton information of a human body provided by Kinect V2 sensors is exploited. Since Kinect V2 acquires 3D points at the joints of the human skeleton, we take advantage of their 3D joint coordinates as references for the registration. No user intervention during the registration process is required and all the processes are automatically performed.

This paper consists of five sections. In Section 2, we review the previous works about calibration methods related to Kinect sensors. Section 3 describes the proposed calibration method between two Kinect V2. Experimental results are presented in Section 3.4, followed with conclusions in Section 4.

2. Related Works

The calibration method for multiple cameras basically starts by finding pairs of corresponding points in the images taken by different cameras. The well-known Zhang method [8] for 2D color camera calculates the camera parameters from the corresponding key-points in a checker board. Here, key-point detection method such as SIFT [9–12] can be used for the matching of the corresponding key-point pairs. In the case of a 3D camera with both depth and color sensors such as Kinect, the checker board or the key-point matching algorithm can be also used [13]. However, to consider the 3D structure into the calibration, corresponding pairs must be selected from various locations and depths of the 3D object [14], where manually generated correspondence pairs using the checker board may be adopted [15, 16]. In the case of a laser scanner, however, since key-points cannot be readily detected as in RGB cameras, geometrically shaped devices [17, 18] may be required to identify correspondence points.

Soleimani et al. [7] used two identical checker patterns on a double-sided board for the calibration of two Kinect V2. Here, the same checker pattern should be placed at the same position on both sides of the flat board. The correspondence points of the same position can be identified by the Kinect IR camera on the opposite side of the board. Then, the rotation and translation transformation can be made for the two sensors by calculating the cross-covariance matrix obtained by 20 correspondence points on the checker patterns. Yang et al. [19] used Zhang's method [8] to calibrate four Kinect sensors. From each IR camera, a checker pattern from a board fixed to a tripod capable of rotating in three directions was taken and the relative pose was detected based on the checker board. For 3D data registration, there are also specially designed calibration-assisting devices other than the checker board. For example, in [20], a wand was used to register depth data from multiple RGBD cameras as a calibration-assisting device. Salau et al. [21] used the calibration device with three sticks perpendicular to each other, which were fixed at the center and six pieces of wood ball were placed at the end. Fornaser et al. [6] proposed a method of registering data of three Kinect V2 in one spot by putting a colored patch on a common soccer ball and attaching a stick. When a user selects an initial position of the ball, three or more cameras track the ball and generate a corresponding point. Su et al. [22] also used a spherical object (like the ball in [6]) with a rod as a calibration tool. In order to detect and track the spherical object, a specific color was painted on. Five cameras track the center of the ball and generate correspondence point pairs. There is another method that uses balls as calibration tools. In Kumara et al. [23], four balls with red, yellow, blue, and green colors were used to calibrate 3 Kinect sensors.

Note that all the abovementioned calibration methods for multiple Kinect sensors require specially made calibration-assisting devices. This can be not only cumbersome but also fragile for portable 3D scanning applications with frequent relocations. For the applications with such dynamic environments, in this paper, we propose to use the human skeleton data provided by the Kinect for the calibration. The skeleton data had been used for the calibration of multiple Kinect version 1 (V1) [24]. Note that, because of the sparse and noisy skeleton data from Kinect V1, a sequence of synchronized frames captured from the multiple Kinect V1 should be used. However, for Kinect V2, since a PC supports only one Kinect V2, the method based on a synchronized sequence of frames for the calibration is not appropriate. In fact, since Kinect V2 uses an updated sensing method of ToF depth, it provides improved depth quality with reduced interference artifacts. This enables us to use only one set of the captured RGBD data from each Kinect V2 for the registration of the 3D point clouds.

3. Calibration of Dual Kinect V2 Sensors by Using Skeleton Information

In this section, the calibration of two Kinect V2 sensors, facing each other, for the construction of the omnidirectional 3D point cloud is presented. The skeleton data provided by the Kinect are utilized to have the initial extrinsic camera parameters by matching the corresponding skeleton points between the two Kinect sensors.

3.1. Data Acquisition from Dual Kinect Sensors. According to the software development kit (SDK) provided by the Kinect V2 maker [25], since only one Kinect V2 can run on one PC, two PCs are required for our dual Kinect V2 system. Then, for the construction of 3D point cloud, the scanned data from the two Kinect V2 sensors should be combined and a synchronized data transfer from the client PC to the server is required. Note that this is not a problem for Kinect V1, where multiple Kinect sensors can be operated on one PC. Considering that Kinect V2 produces color and depth data at a rate of 30 frames per second, the amount of data that needs to be transmitted to the server PC is quite large. However, since the calibration is usually done as an off-line process, a real-time transmission of the scanned data from the client PC to the server is not necessary. So, the scanned data

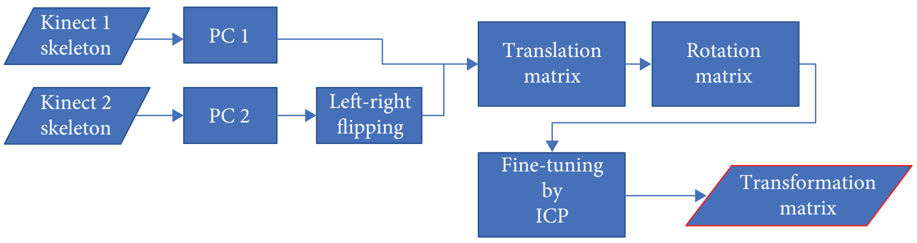


FIGURE 1: Flowchart of the proposed algorithm.

can be stored in each PC for a synchronized time frame with a trigger signal initiated by the detection of the skeleton data and the stored data of the client PC are transferred to the server PC for the calibration.

In this paper, the 3D skeleton joints in the human body provided by Kinect V2 for each depth frame are used as the correspondence point pairs. Therefore, our calibration needs only a standing person as a calibration reference. Since the skeleton data obtained from a human body are used as the calibration reference, discrepancies in body point locations can occur if there exists a quick body movement during the data acquisition. Therefore, the standing person needs to be at standstill for a while.

Note that the Kinect was originally developed as a device for detecting human gesture for more realistic video games. Since there is no need to distinguish the back of a person from the front in the game applications, Kinect assumes that the skeleton is obtained in the frontal view even if it is captured from the back side. Therefore, deploying two facing Kinects as in our method, it is required to flip the 3D data scanned by the back view horizontally (i.e., left-to-right flipping) before the registration starts (see Figure 1).

3.2. Registration by Geometric Transformations. We focus on the calibration of the extrinsic parameters of the dual sensors. But, for the intrinsic parameters of Kinect V2, the maker-provided parameters are used. So, the extrinsic calibration process regarding the relative positions of the two cameras is executed. The process of registering data obtained from two coordinate systems into a common one is performed by confirming information about the position and direction of the cameras and executing the transformation accordingly. Note that RGBD cameras, like the Kinect sensor, are equipped with IR depth as well as RGB color sensors and, to construct a colored 3D point cloud, both sensors are to be calibrated.

As shown in Figure 1, the calibration is executed with two major steps, namely, the initial calibration by estimating the geometric transformation matrices and the fine-tuning. In the initial calibration step, the point clouds in two coordinate systems are brought close to a common coordinate by translation and rotation transformations, where the skeleton joints are used as the reference points for determining the transformation matrices. Then, in the fine-tuning step, the final calibration is obtained by Iterative Closest Point (ICP) [26] algorithm.

As mentioned already, Kinect cannot distinguish the front from the back sides of a person and it assumes that the skeleton is obtained in the frontal view even if it is captured from the back side. Therefore, even after the left-right flipping in Figure 1, there may exist some discrepancies in the skeleton joints between the front and the flipped back 3D data for the head, foot, fingertip, and thumb. Figure 2 shows the skeleton data captured from the front-side and the back-side cameras. Since the head of the skeleton data creates skeleton points based on the face, they are slightly shifted forward from the actual positions in the back-side

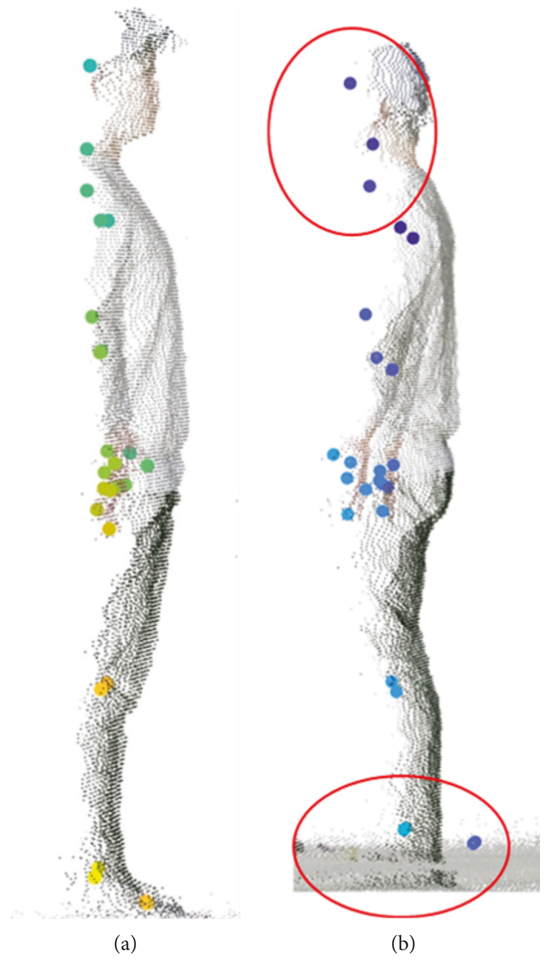


FIGURE 2: The side views of point clouds and skeleton joints captured by (a) the front-side camera and (b) the back-side camera. In the data from the back-side camera, skeleton joints of the head and the foot are incorrectly located.

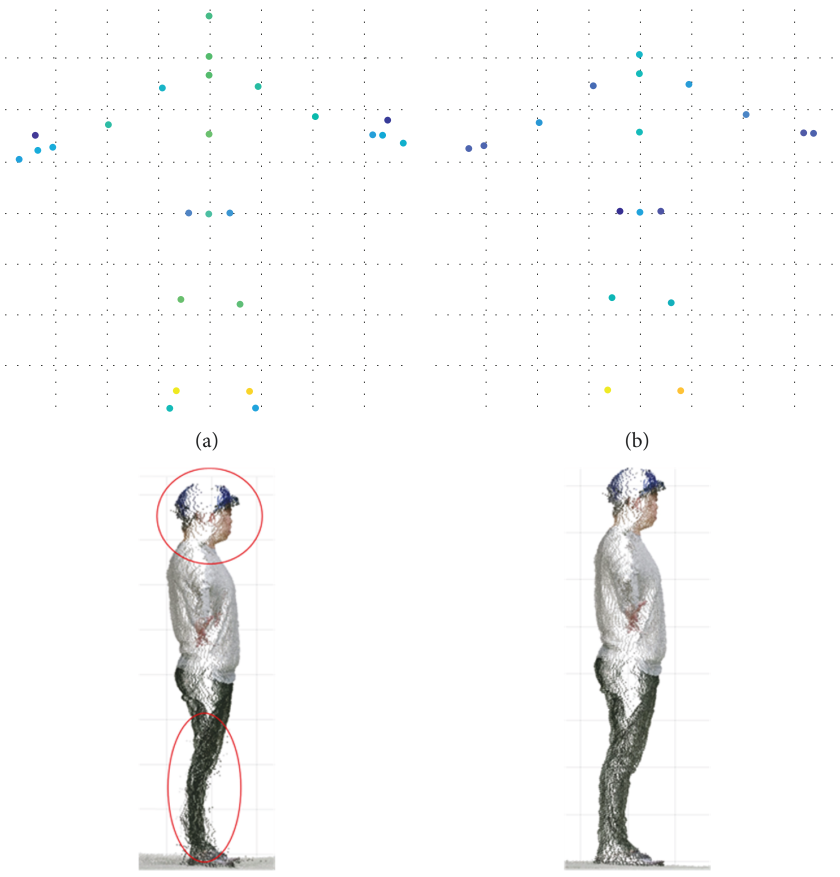


Figure 3: Locations of the skeleton joints for (a) all 25 positions and (b) selected 18 positions. Construction of 3D point cloud after the calibration by (c) all 25 skeleton joints and (d) selected 18 skeleton joints.

scan. Therefore, it is not recommended to use the joints in the head skeleton as correspondence pairs for the calibration. Also, although the skeleton points at the fingertip and thumb are more stable than the ones in the head, there are cases that make a consistent detection of the joints in the finger difficult. For example, if the finger is hidden, the estimated finger joints can be located arbitrarily. Finally, since the skeleton points at the foot are also designed to be positioned at the frontal view, there may exist a mismatch between the 3D data from the two cameras. Therefore, the skeleton data from the abovementioned skeleton joints are discarded and the remaining 18 points are selected for the calibration among a total of 25 skeleton joints generated by Kinect V2. Figure 3(c) shows the result of using all 25 skeleton joints of Figure 3(a); Figure 3(d) is the result of using only 18 joints of Figure 3(b). As you can see inside the circles in Figure 3(c), distortions occur at the foot and head sides when all 25 skeleton joints are used.

Scanned data can be translated and rotated by a rigid transformation matrix. Since the "SpineBase" joint is located at the center of the human skeleton (see Figure 4), firstly, the 3D points obtained by the two Kinect V2 are translated such that the two "SpineBase" points from the two Kinect sensors coincide at the same position, which is considered as the origin $(0,0,0)$ of the new coordinate system. That is, we select the "SpineBase" point located at the center of the skeleton as the origin of the new coordinate and apply the translation matrix such that the two "SpineBase" points match at the origin of the new coordinate. To this end, we apply the following translation matrices T_1 and T_2 for the two sets of 3D point clouds obtained by Kinect$_1$ and Kinect$_2$, respectively.

$$T_k = \begin{bmatrix} 1 & 0 & 0 & -t_{xk} \\ 0 & 1 & 0 & -t_{yk} \\ 0 & 0 & 1 & -t_{zk} \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad k = 1, 2,$$
$$O_1 = (t_{x1}, t_{y1}, t_{z1}),$$
$$O_2 = (t_{x2}, t_{y2}, t_{z2}), \tag{1}$$

where O_1 and O_2 represent the "SpineBase" points in Kinect$_1$ and Kinect$_2$ coordinates, respectively. By applying the translation matrices, the original 3D points $p_k = \{(x_k(i), y_k(i), z_k(i)), i = 1, \dots, N_k, k = 1, 2\}$, are translated and relocated to $p_k^T = \{(x_k^T(i), y_k^T(i), z_k^T(i)), i = 1, \dots, N_k, k = 1, 2\}$, where N_1 and N_2 are the total number of scanned 3D points

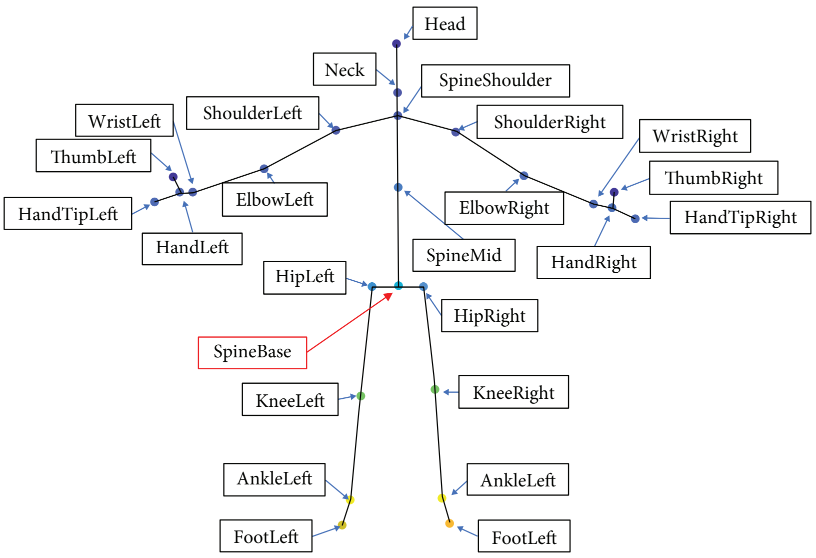


Figure 4: Positions of 25 skeleton joints provided by Kinect V2. In our method, "SpineBase" is set as the origin of the common coordinate.

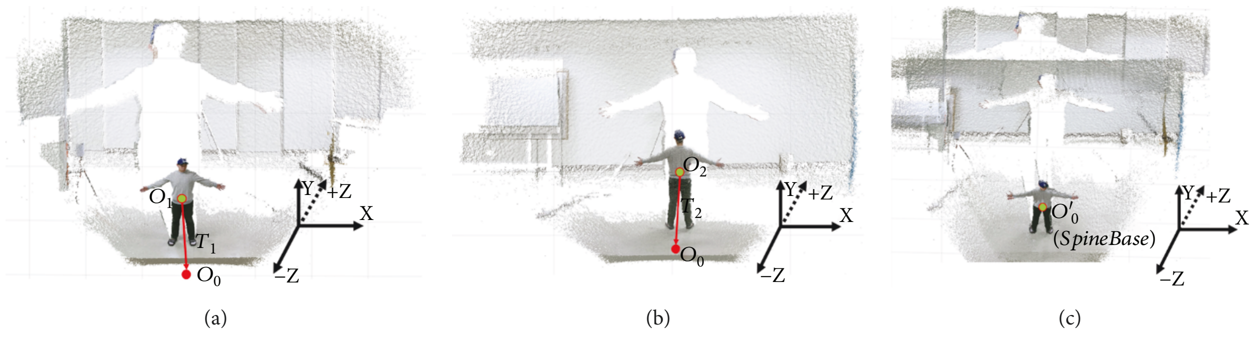


Figure 5: 3D point clouds (a) in the coordinate of the server Kinect V2, (b) in the coordinate of the client Kinect V2, and (c) in the common coordinate after the translation.

obtained by each Kinect V2, $k = 1, 2$, and p_k^T is obtained as follows

$$\begin{bmatrix} x_k^T(i) \\ y_k^T(i) \\ z_k^T(i) \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & -t_{xk} \\ 0 & 1 & 0 & -t_{yk} \\ 0 & 0 & 1 & -t_{zk} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_k(i) \\ y_k(i) \\ z_k(i) \\ 1 \end{bmatrix}, \quad (2)$$
$$i = 1, \ldots, N_k, k = 1, 2.$$

Figure 5 shows an example of the 3D point cloud before (Figures 5(a) and Figure 5(b)) and after (Figure 5(c)) the translation.

Next, the translated 3D points are aligned by rotations. Assuming that the floor is flat and the two Kinects are installed in almost parallel with the floor, no rotation alignment of the 3D data with respect to the z-axis will be executed (see Figure 6 for the orientation of z-axis rotation). Instead, small misalignments with respect to the z-axis rotation are expected to be corrected at the fine-tuning step. For the other two coordinates, the rotation alignments with respect to X and Y axes in the 3D space are executed, where the angles to be rotated can be obtained by using the straight line formed by the joints of the skeleton.

Note that the Kinect cameras are tilted down slightly (i.e., rotated with respect to the x-axis, see Figure 6) to cover the whole body and the floor. As a result, the 3D point clouds are inclined (see Figure 7) and should be aligned by rotating with respect to the x-axis. Here, the angle, α_k, to be rotated can be estimated by calculating the inner product between the vector formed by "SpineShoulder" and "SpineMid" joints in the skeleton and the y-axis as follows

$$v_{k1} \cdot v_{k2} = |v_{k1}||v_{k2}| \cos(\alpha_k), \quad k = 1, 2, \quad (3)$$

where v_{k1} is a straight-line vector formed by "SpineShoulder" and "SpineMid" joints, and v_{k2} corresponds to the vector of the y-axis. Then, the angle between the two vectors can be obtained as

$$\begin{aligned} \alpha_k &= \cos^{-1}\left(\frac{v_{k1} \cdot v_{k2}}{\|v_{k1}\|\|v_{k2}\|}\right) \\ &= \cos^{-1}\left(\frac{x_{k1}^T x_{k2}^T + y_{k1}^T y_{k2}^T + z_{k1}^T z_{k2}^T}{\sqrt{x_{k1}^{T\,2} + y_{k1}^{T\,2} + z_{k1}^{T\,2}}\sqrt{x_{k2}^{T\,2} + y_{k2}^{T\,2} + z_{k2}^{T\,2}}}\right). \end{aligned} \quad (4)$$

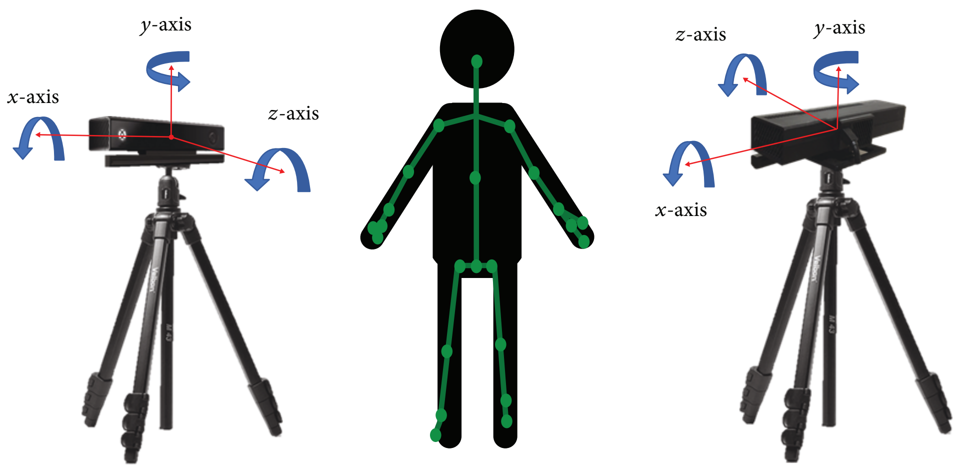


Figure 6: Layout of our dual Kinect V2 and the orientations of coordinate rotations.

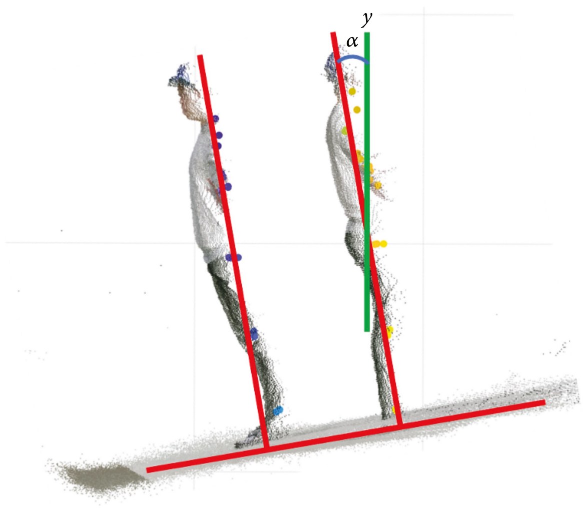


Figure 7: The inclined point clouds in Y-Z plane due to the tilt down installation of the Kinect. The angle, α, between the y-axis and the straight line connecting the "SpineShoulder" and "SpineMid" joints in the skeleton can be estimated.

Calculating the angle α_k for each Kinect, we have the rotation matrix R_{kx} as

$$R_{kx} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha_k & -\sin\alpha_k & 0 \\ 0 & \sin\alpha_k & \cos\alpha_k & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (5)$$

Finally, we need to take a rotation alignment with respect to the y-axis. Although the two Kinect V2 sensors are installed so that they are facing each other, the arc angle from the top view between them may not be exactly 180°, which needs to make a rotation with respect to the y-axis. As an example, the two Kinects in Figure 8(a) are not on the straight line and are separated by 135° arc angle. As in the case of the x-axis rotation, the angle, β_k, to be rotated with respect to the y-axis can be obtained by calculating the inner product between the vector formed by the straight line connecting "ShoulderRight" and "ShoulderLeft" and the x-axis (see Figures 8(b) and Figure 8(c)). Then, the 3D points are rotated with respect to the y-axis by the following matrix

$$R_{ky} = \begin{bmatrix} \cos\beta_k & 0 & \sin\beta_k & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\beta_k & 0 & \cos\beta_k & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (6)$$

Figure 8(c) shows the aligned result of the rotation by R_{ky}. The overall rotation matrix R_k with respect to X and Y axes can be obtained by multiplying the two rotation matrices R_{kx} and R_{ky} as follows

$$R_k = R_{kx} \times R_{ky} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha_k & -\sin\alpha_k & 0 \\ 0 & \sin\alpha_k & \cos\alpha_k & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} \cos\beta_k & 0 & \sin\beta_k & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\beta_k & 0 & \cos\beta_k & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos\beta_k & 0 & \sin\beta_k & 0 \\ \sin\alpha_k \sin\beta_k & \cos\alpha_k & -\sin\alpha_k \cos\beta_k & 0 \\ -\cos\alpha_k \sin\beta_k & \sin\alpha_k & \cos\alpha_k \cos\beta_k & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (7)$$

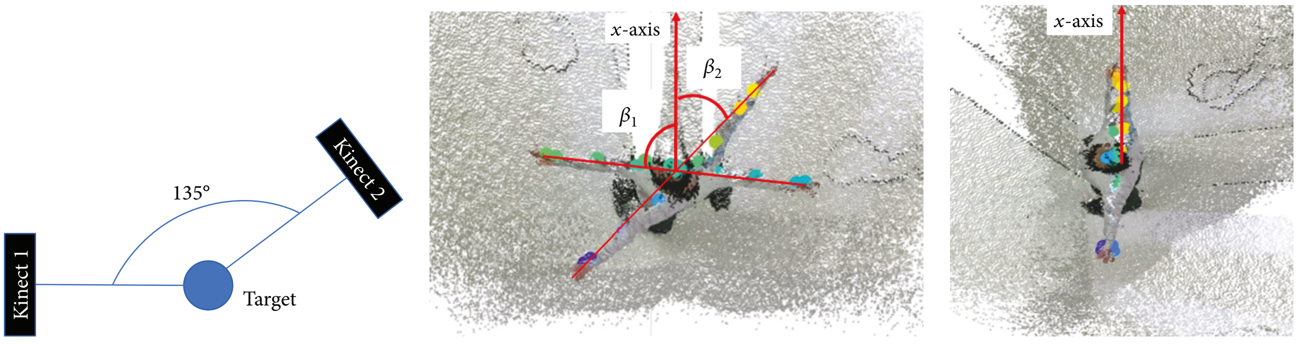


Figure 8: Rotation with respect to the y-axis. (a) Example of two Kinects with 135° arc angle. (b) 3D point cloud of P_k^T (i.e., before the rotation). (c) 3D point cloud of P_k^{TR} (i.e., after the rotation).

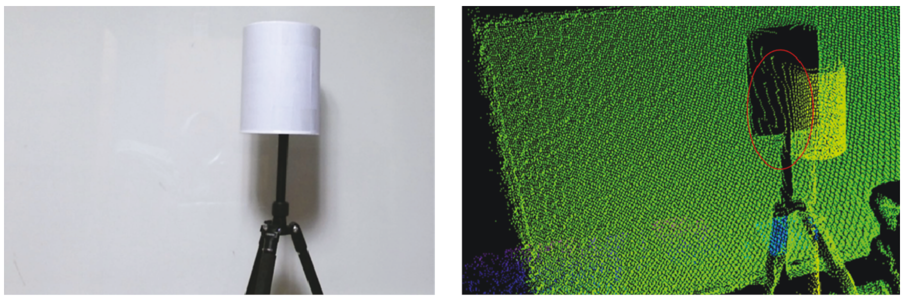


Figure 9: Flying errors. (a) Color image from the Kinect. (b) Point cloud from the Kinect with the flying errors (see inside the red circle).

Applying the rotation matrix in equation 7 to the translated point cloud P_k^T, we have the translated-rotated point cloud, $p_k^{TR} = \{(x_k^{TR}(i), y_k^{TR}(i), z_k^{TR}(i)), i = 1, \dots, N_k, k = 1, 2\}$, as follows

$$\begin{bmatrix} x_k^{TR}(i) \\ y_k^{TR}(i) \\ z_k^{TR}(i) \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\beta_k & 0 & \sin\beta_k & 0 \\ \sin\alpha_k \sin\beta_k & \cos\alpha_k & -\sin\alpha_k \cos\beta_k & 0 \\ -\cos\alpha_k \sin\beta_k & \sin\alpha_k & \cos\alpha_k \cos\beta_k & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_k^T(i) \\ y_k^T(i) \\ z_k^T(i) \\ 1 \end{bmatrix}, \quad i = 1, \dots, N_k, k = 1, 2. \tag{8}$$

3.3. Fine Tuning by Iterative Closest Point. The accuracy of the initial calibration obtained by the translation and the rotations in the previous subsection can be further improved by fine-tuning the calibration parameters. As a fine-tuning method, Iterative Closest Point (ICP) [26] algorithm is adopted. The ICP algorithm is iteratively performing a transformation that minimizes the distances between the initially registered 3D points of the two Kinect V2. If the ICP is attempted for the entire point clouds, then the computational complexity will be enormous. We also note that the ICP algorithm is designed for the matching with common areas or similar shapes. However, in our case, the two sets of the 3D points to register are obtained at the opposite directions, giving a limited common area. To solve these problems, we apply the ICP algorithm to the skeleton data rather than the entire point clouds. Since the distances between correspondence pairs of the skeleton by the initial calibration are sufficiently close and the number of skeleton pairs for the ICP is very small, the algorithm runs very fast. The ICP algorithm yields our final calibrated 3D point clouds.

3.4. Denoising. The scanning sensor of the point cloud suffers from some noises. In particular, flying errors occur frequently for ToF sensor due to wrong distance measurements at the edge of the object [27]. Figure 9 shows an example of the flying errors. These errors are characterized by the fact that the distance between points is longer than the data generated by the actual object. Therefore, a simple denoising method can be applied by calculating the distances between the points. Specifically, computing the average μ and standard deviation σ of the nearest neighbors of each point, we use only the points with $d < \mu \pm k\sigma$ for distance d [28]. Figure 10(a) and Figure 10(b) show the results of the denoising.

3.5. Registration by Multiple Skeletons. The results of the calibration should be consistent regardless of time, place, and

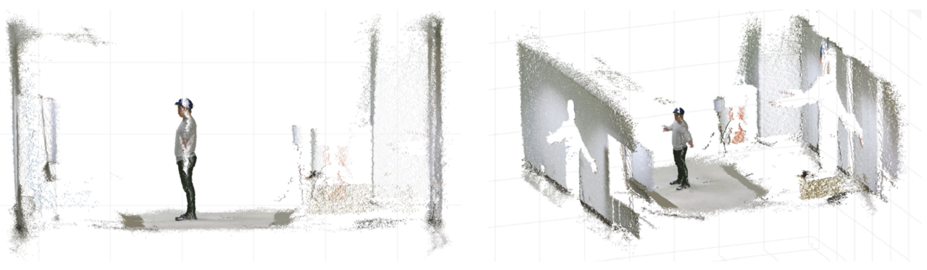


Figure 10: Final point clouds after the denoising. (a) Side view. (b) Bird's-eye view.

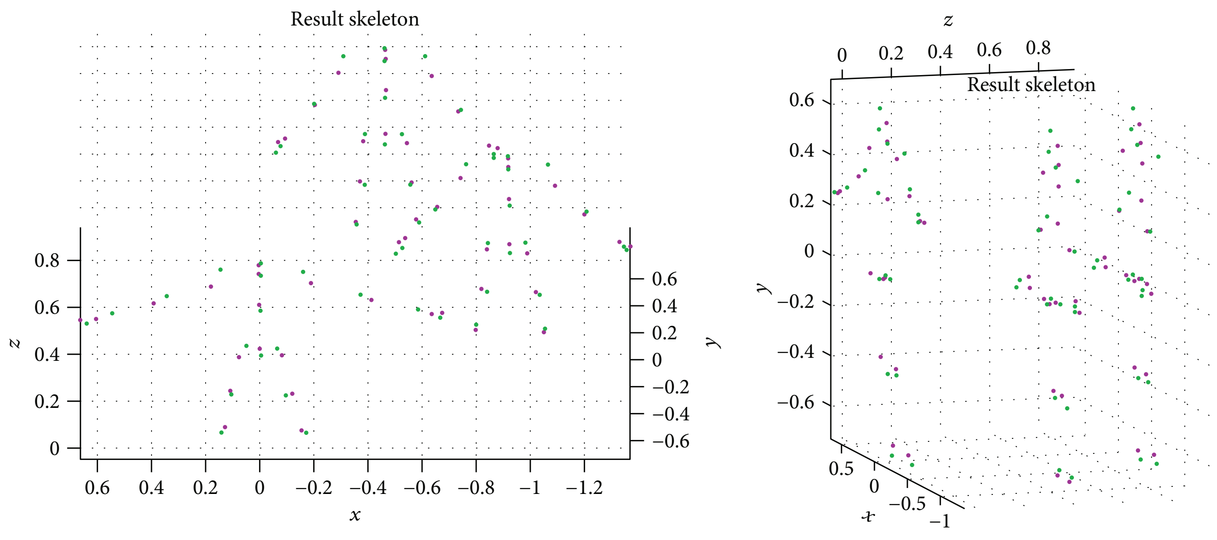


Figure 11: Multiple skeleton data. (a) Front view. (b) Side view.

sensor. However, the Kinect V2 sensor is not a perfect machine and it sometimes produces inaccurate measurements. To alleviate this problem, we can scan the human body multiple times, acquiring multiple skeleton data from various locations between the two fixed Kinect V2 sensors. Having multiple 3D point clouds for the calibration, in the initial calibration step, the origin is set to the "SpineBase" point of the first skeleton data. Then, all subsequently obtained skeletons are registered to the same coordinate system to create a new point cloud. Figure 11(a) and Figure 11(b) show multiple skeleton data from the front and side views, respectively. The rotation angles α' and β' of the entire point clouds are calculated by averaging the vectors of M multiple skeletons as follows

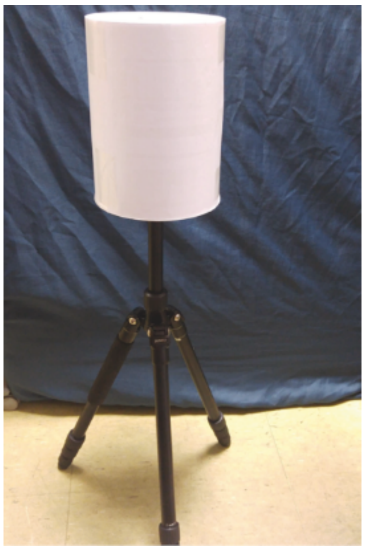
$$\alpha'_k = \cos^{-1}\left(\frac{v_{k1}^{\text{ave}} \cdot v_{k2}^{\text{ave}}}{\|v_{k1}^{\text{ave}}\|\|v_{k2}^{\text{ave}}\|}\right),$$
$$v_{k1}^{\text{ave}} = \frac{\sum_{m=1}^{M} v_{k1}^{m}}{M}, v_{k2}^{ave} = \frac{\sum_{m=1}^{M} v_{k2}^{m}}{M}, k = 1, 2. \quad (9)$$

In the fine-tuning step, since the points applied to the ICP algorithm are increased, a more stable result can be obtained.

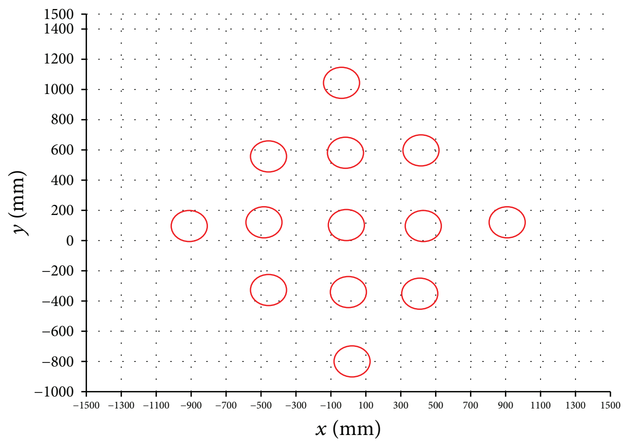
4. Experimental Results

The SDK for Kinect uses MATLAB tool box [29]. However, due to the nature of MATLAB, multithreaded programming to continuously transfer large point cloud data over the network is difficult. So, the 3D scanned data captured from one PC (client) are transferred to another PC (server) via a SMB (Server Message Block) Protocol. The flipped and transferred 3D data from the client PC are processed in the server PC for the registration. In the following subsections, the accuracy of the proposed registration method is evaluated in various aspects.

4.1. Registration Accuracy for a Cylindrical Object. After the calibration by a human body, a cylindrical test object is scanned with the two Kinects in various locations and their radii are measured from the registered 3D point clouds and compared with the actual radius. The accuracy of the proposed calibration method also is evaluated by observing registered 3D point clouds in various situations by changing the distances between the two Kinect V2 sensors, the angles between the two Kinect V2, and the number of skeletons used. Specifically, 3D measurements of 13 cylindrical testing objects as shown in Figure 12 are evaluated. The cylindrical testing object is detected and its radius is measured



(a)



(b)

Figure 12: Experimental setup. (a) Cylindrical testing object (actual dimension: 0.1 m radius and 0.25 m height). (b) Placement of 13 cylindrical objects for the evaluation of the calibration accuracy.

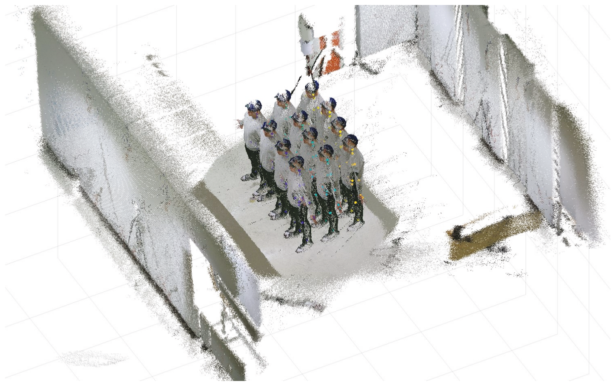


Figure 13: 12 positions of the standing person as a calibration target for acquiring multiple skeleton data.

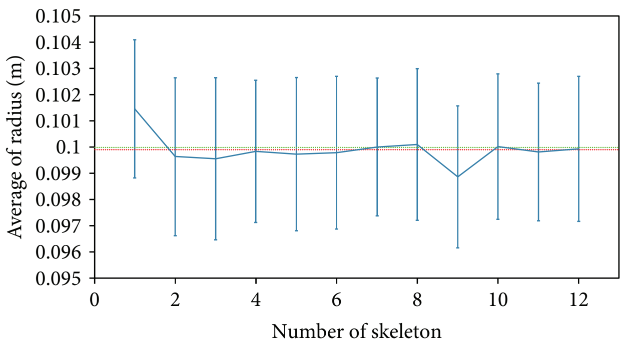


Figure 14: Comparison of the measured averages for 5 measurements (blue lines) and the actual radius of the cylinder (red dotted line). The x-axis represents the number of skeletons used for the calibration.

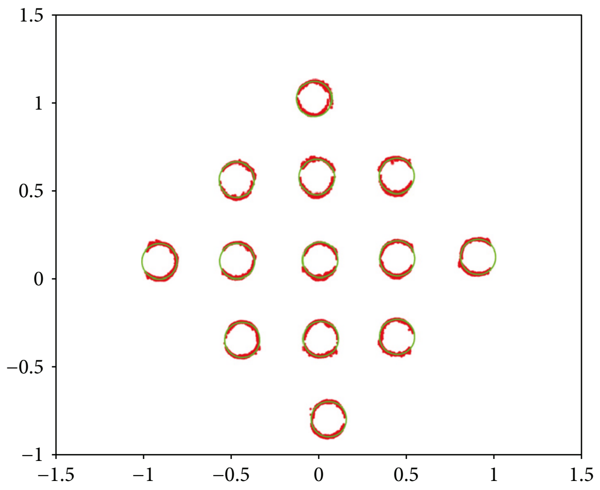


Figure 15: The aligned 3D point cloud with red dots and the actual cylinder radii with green circles of 13 cylinders positioned as in Figure 12(b) from the top view.

by the MATLAB function of "pcfitcylinder," which is based on MSAC (M-estimator SAmple Consensus) [30], a variation of RANSAC (RANdom SAmple Consensus) [31] algorithm. From the calibrated 3D point cloud, the measured radius of the cylinder is compared with its actual size for a quantitative evaluation, where the actual radius and the height of the cylinder used in the experiment are 0.1 m and 0.25 m, respectively.

To test the stability and consistency of the calibration and the ICP algorithm, we used multiple skeleton data. A total of 12 skeleton data, which were acquired at 12 different positions within the common FOV coverage of the two Kinect V2 sensors (see Figure 13), were used in the experiments. Transformation matrices were obtained by adding the skeleton data from 1 to 12. With the 12 transformation matrices (i.e., from 1 to 12 skeletons), we constructed the 3D point clouds of the cylindrical objects in 13 different positions shown in Figure 12(b) and measured the cylinders' radii. For each position of the cylinder, its radius is measured 5 times and the average values for each transformation matrix obtained from 1 to 12 skeletons are plotted in Figure 14. As shown in the figure, just a couple of human skeleton dataset may be sufficient for the calibration and its calibration matrix is useful for all different positions within the common FOV

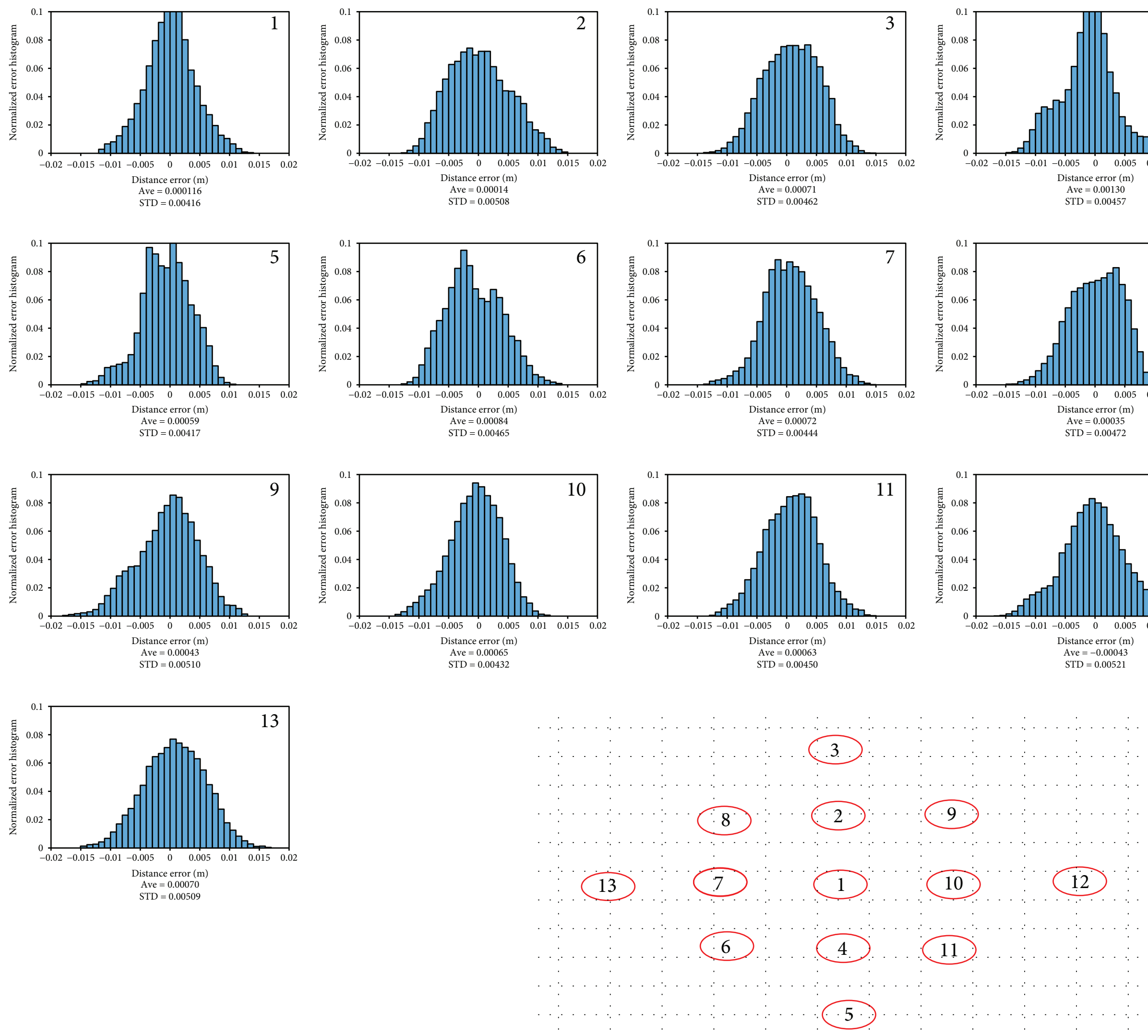


Figure 16: Histograms of the radius errors at each cylinder position.

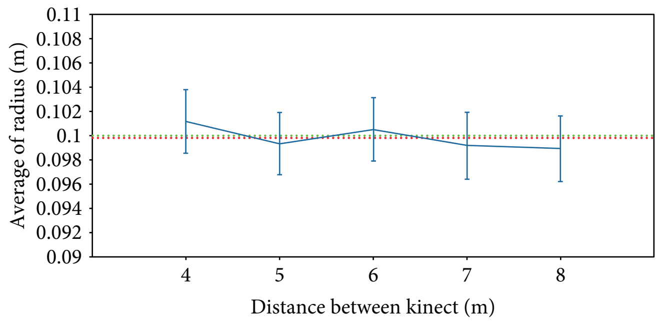
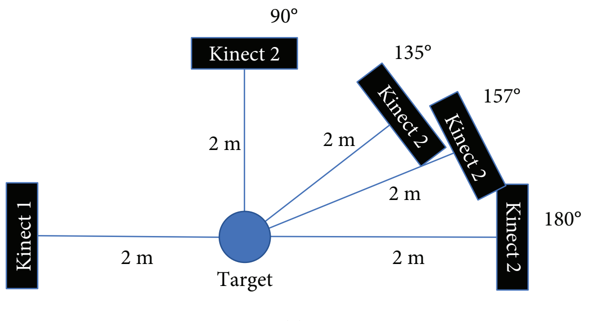


Figure 17: Evaluation of our registration method with various distances between the two Kinects from 4 m to 8 m. The red and green dotted lines represent the average values of the measured and the actual radii for the 13 cylinders positioned as in Figure 12(b), respectively.

of the two Kinect V2 sensors. The actual radius of the model (green dotted line in Figure 14) is 0.1 m, and the average value of the measured radius (red dotted line) is 0.0999 m. Thus, the average error is 0.0001 m (0.1 mm).

Figure 15 shows the point clouds of 13 cylinders from the top view. The red points represent the scanned 3D point clouds of the cylinders, and the green ones are the traces of the actual cylinders' radii, which are the ground truth for the measurement evaluation. As you can see, the red 3D points almost coincide with the green circles, showing the accuracy of the calibration. Because of the front-back deployment of the two facing Kinect V2, there are some blind spots



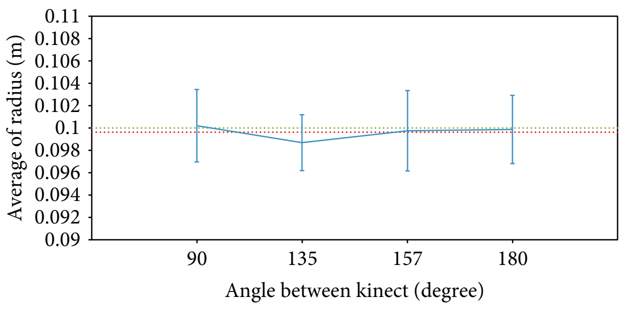


Figure 18: Evaluation of our registration method with various arc angles between the two Kinects from 90 to 180 degrees. The red and green dotted lines represent the average values of the measured and the actual radii for the 13 cylinders positioned as in Figure 12(b), respectively.

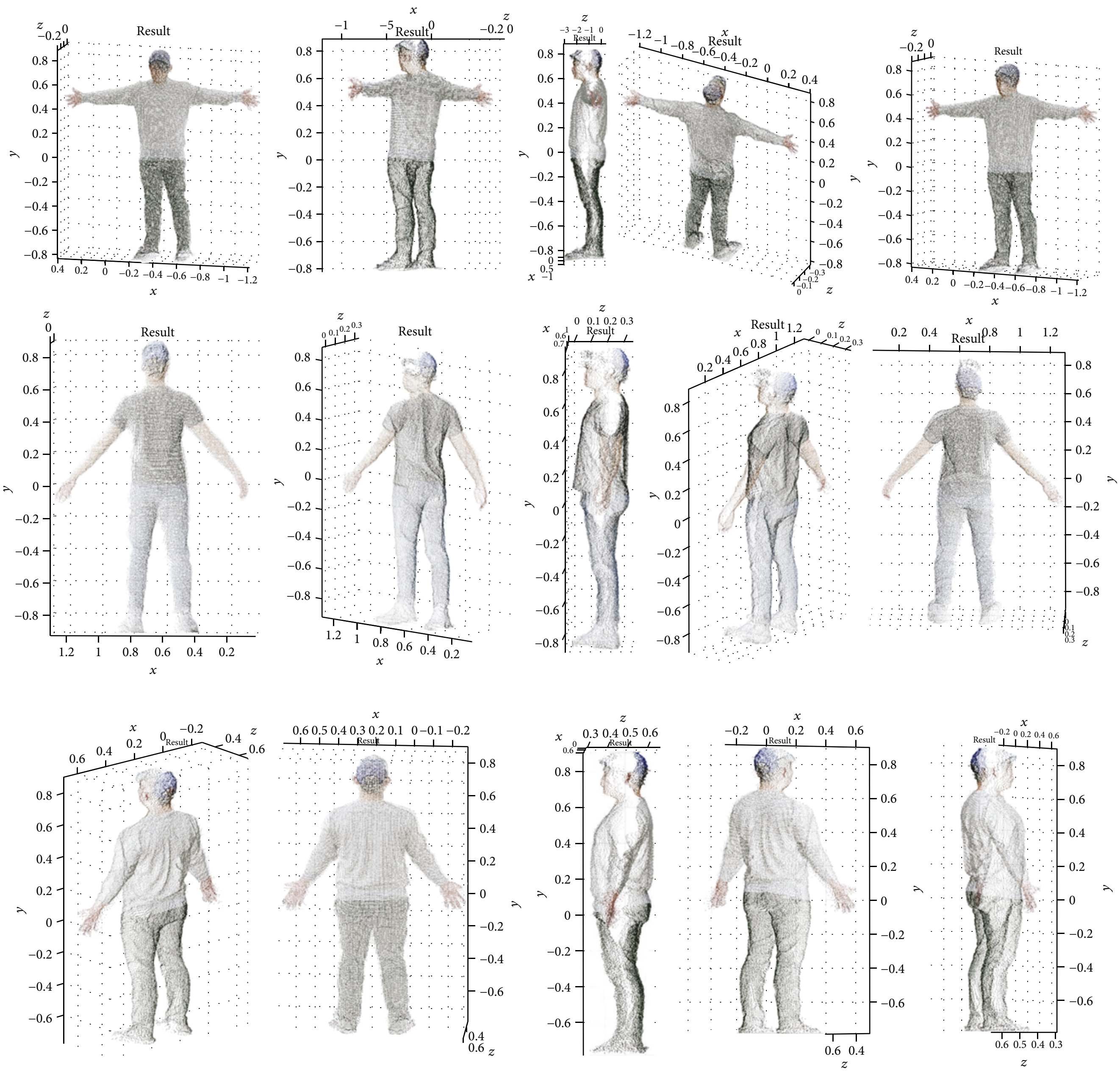


Figure 19: Registered 3D point clouds for three persons (one person per each row) in various views.

Table 1: Results for 16 participants as references for the proposed calibration. The average error and the standard deviation (STD) are calculated from the 5 cylinders (i.e., cylinders numbered 1, 2, 4, 7, and 10 in Figure 16), where the calibrated parameters are obtained for each participant.

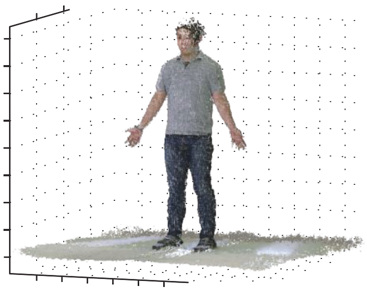
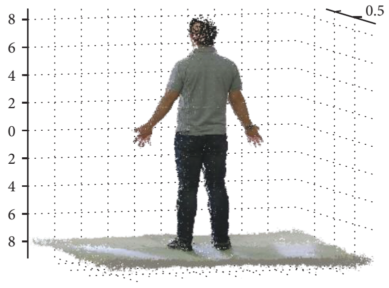
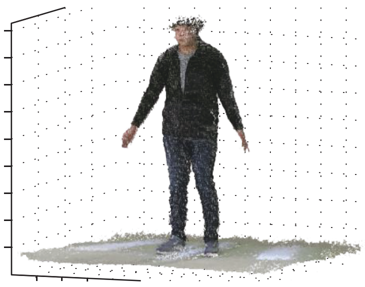
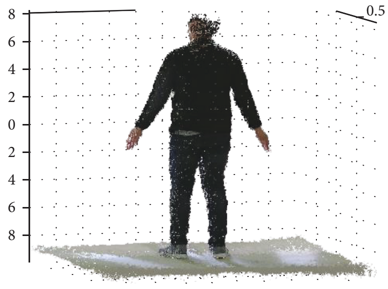
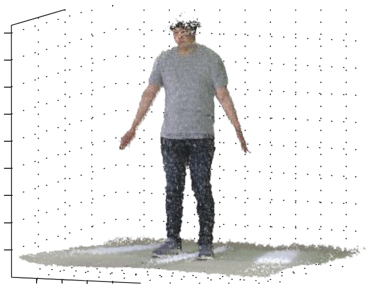
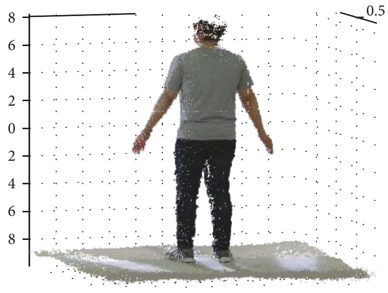
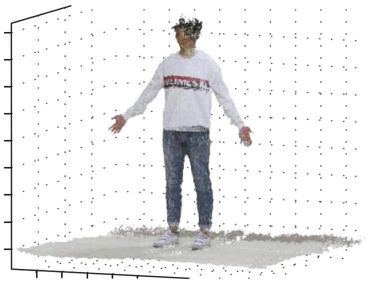
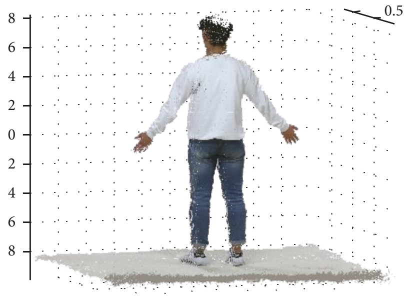
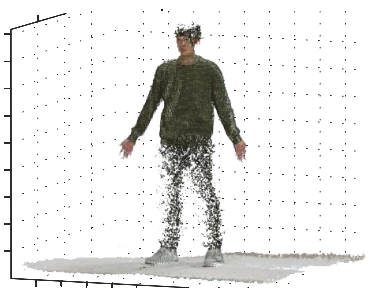
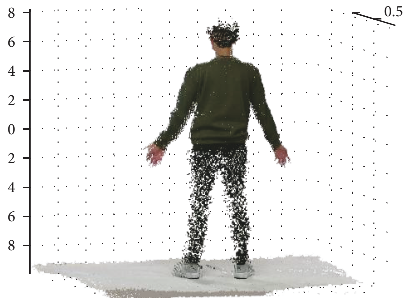
Reference number	Gender/body shape	Average error/STD (unit: m)	Point cloud in front view	Point cloud in back view
Participant 1	Male/medium	0.00960/0.0042		
Participant 2	Male/big	0.09740/0.0048		
Participant 3	Male/big	0.09615/0.0051		
Participant 4	Male/medium	0.09790/0.0049		
Participant 5	Male/medium	0.09898/0.0049		

Table 1: Continued.

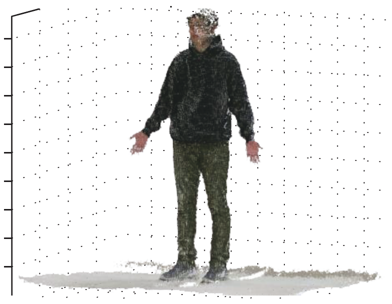
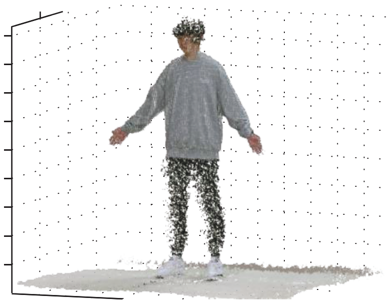
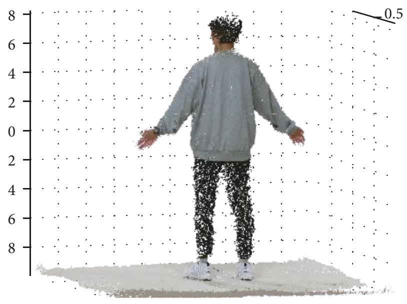
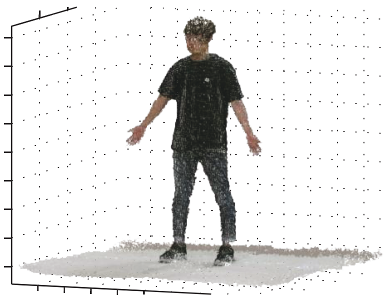
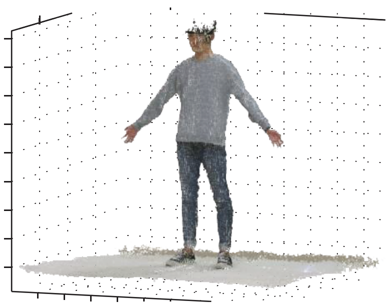
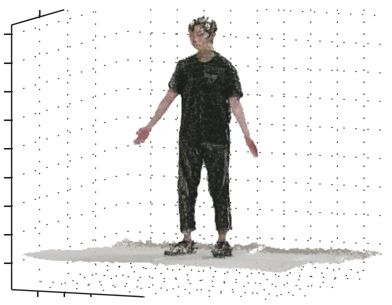
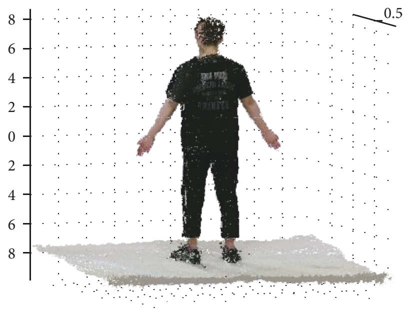
Reference number	Gender/body shape	Average error/STD (unit: m)	Point cloud in front view	Point cloud in back view
Participant 6	Male/big	0.09611/0.0050		
Participant 7	Male/medium	0.09522/0.0050		
Participant 8	Male/medium	0.09724/0.0051		
Participant 9	Male/big	0.09875/0.0051		
Participant 10	Male/medium	0.09777/0.0050		

Table 1: Continued.

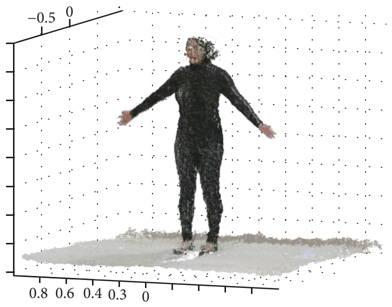
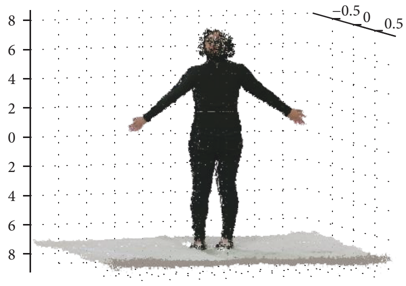
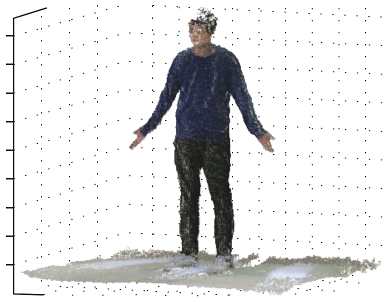
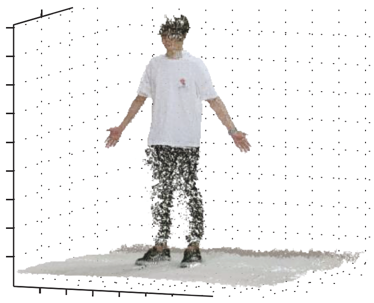
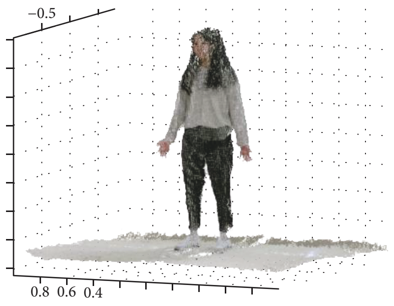
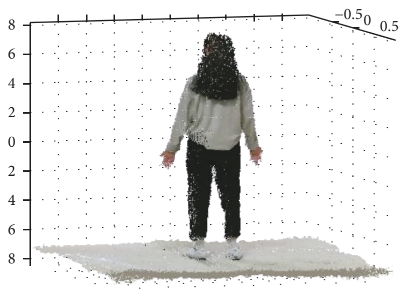
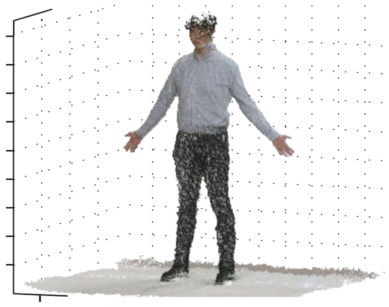
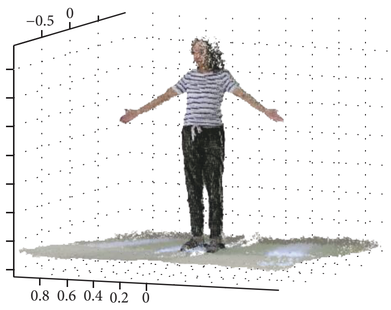
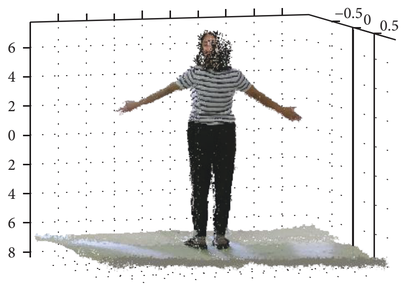
Reference number	Gender/body shape	Average error/STD (unit: m)	Point cloud in front view	Point cloud in back view
Participant 11	Female/medium	0.09720/0.0051		
Participant 12	Male/big	0.09916/0.0051		
Participant 13	Male/medium	0.09687/0.0051		8 6 4 2 0 2 4 6 8 0.5
Participant 14	Female/small	0.09866/0.0051		
Participant 15	Male/big	0.09612/0.0051		

Table 1: Continued.

Reference number	Gender/body shape	Average error/STD (unit: m)	Point cloud in front view	Point cloud in back view
Participant 16	Female/small	0.10096/0.0052		

of the scanning and some of 3D point clouds are missing in Figure 15.

Figure 16 shows the histograms of the distance errors for the measured radii in 13 different positions used in the experiments. As can be seen from the histograms, for all cases, the average errors are less than 0.0013 m (1.3 mm), which is the result for the cylinder 4. Also, the standard deviations (STD) are less than 0.00521 (5.21 mm), which is the result of the cylinder 12.

4.2. Registrations with Various Distances and Angles between the Two Kinects. In this section, we compare the registration results by varying the distance between the two Kinect sensors from 4 m to 8 m with 1 m increment. Three skeleton data and three cylinder data were used. As shown in Figure 17, within the distance range from 4 m to 8 m, the registration accuracy of 3D point clouds is consistently maintained. At the distances less than 4 m or larger than 8 m, however, the skeleton of the whole human body was not completely captured by the Kinect sensor. The average radius of 10 measurements is 0.0998 m and the error is less than 0.0002 m (0.2 mm).

Although our experiments are based on the deployment of the two facing Kinect V2 sensors with about 180° arc angle, we also evaluate the registration accuracy of the 3D point clouds by varying the arc angles between them. As shown in Figure 18(a), our experiments were carried out at a distance of 2 m from the central position of the skeleton. We tested 90°, 135°, 157°, and 180° arc angles. The results of calibration and cylinder fitting are shown graphically in Figure 18(b). The average measured radius of the four angles is 0.0996 m and the error is about 0.0004 m (0.4 mm). This implies that our calibration method is robust against the arc angle between the two Kinects, which makes the deployment of the two Kinects easy.

4.3. Registration Accuracy for Human Bodies. In Figure 19, the registered 3D point clouds for human bodies are demonstrated with various views for the subjective evaluation. Figure 19 shows the 3D point clouds for the human body for three different persons. Each row of the figure shows a different person and each column represents the different view. We also conducted experiments with 16 additional human bodies with different genders and body shapes. The registered 3D point clouds are shown in Table 1 for the subjective evaluation. As shown in the table, no noticeable bias is observed in the calibration accuracy in terms of the gender and the body shape. However, there was a difficulty in capturing 3D points for those participants who wear dark pants with specific fabric materials (see Participants # 5, 7, and 13 in Table 1). This is due to the limitation of the infrared sensor in Kinect V2 rather than the calibration method.

5. Conclusions

In this paper, we have proposed a registration method for 3D point clouds scanned by two Kinect V2 sensors, which are facing each other. Our method is especially suitable for applications with frequent reinstallation and relocation of the Kinect sensors. To this end, no calibration-assisting devices but a standing person in between the two Kinect V2 sensors are needed. Scanning a person as a calibration reference, the skeleton joints provided by the two Kinect sensors are used for the correspondences required for the calibration. Since we used skeleton joints without any specific tools for the correspondence, the detection process of the corresponding keypoints between the two 3D point clouds is not necessary, enabling a fast calibration. Once the calibration is done by human body as a calibration reference, its accuracy has been demonstrated by measuring 13 cylindrical testing objects in various possible locations. The average error of the measured radii to the actual ones for the cylinder experiments is negligible, which are less than 1.3 mm and the standard deviations (STD) are no more than 5.21 mm. It is also shown that average errors of the cylinder radii for varying the distances from 4 to 8 meters and the arc angles from 90 to 180 degrees between the two Kinects are quite small. The experimental results demonstrate that our calibration method is robust within certain ranges of distance and arc angle.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

This research was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2018R1D1A1B07043542).

References

[1] X. Du, M. H. Ang, and D. Rus, "Car detection for autonomous vehicle: LIDAR and vision fusion approach through deep learning framework," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 749–754, Vancouver, Canada, 2017.

[2] D. Feng, L. Rosenbaum, and K. Dietmayer, "Towards safe autonomous driving: capture uncertainty in the deep neural network for lidar 3D vehicle detection," 2018, http://arxiv.org/abs/1804.05132.

[3] C. Dal Mutto, P. Zanuttigh, and G. M. Cortelazzo, "Microsoft kinect™ range camera," in *Time-of-Flight Cameras and Microsoft Kinect™*, pp. 33–47, Springer, Boston, MA, USA, 2012.

[4] W. Song, S. Yun, S.-W. Jung, and C. S. Won, "Rotated top-bottom dual-kinect for improved field of view," *Multimedia Tools and Applications*, vol. 75, no. 14, pp. 8569–8593, 2016.

[5] A. M. Pinto, P. Costa, A. P. Moreira, L. F. Rocha, G. Veiga, and E. Moreira, "Evaluation of depth sensors for robotic applications," in *2015 IEEE International Conference on Autonomous Robot Systems and Competitions*, pp. 139–143, Vila Real, Portugal, 2015.

[6] A. Fornaser, P. Tomasin, M. De Cecco, M. Tavernini, and M. Zanetti, "Automatic graph based spatiotemporal extrinsic calibration of multiple Kinect V2 ToF cameras," *Robotics and Autonomous Systems*, vol. 98, pp. 105–125, 2017.

[7] V. Soleimani, M. Mirmehdi, D. Damen, S. Hannuna, and M. Camplani, "3D data acquisition and registration using two opposing kinects," in *2016 Fourth International Conference on 3D Vision (3DV)*, pp. 128–137, Stanford, CA, USA, 2016.

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

[9] R. Liu, H. Zhang, M. Liu, X. Xia, and T. Hu, "Stereo cameras self-calibration based on sift," in *2009 International Conference on Measuring Technology and Mechatronics Automation*, pp. 352–355, Zhangjiajie, China, 2009.

[10] D. Dwarakanath, A. Eichhorn, C. Griwodz, and P. Halvorsen, "Faster and more accurate feature-based calibration for widely spaced camera pairs," in *2012 Second International Conference on Digital Information and Communication Technology and it's Applications (DICTAP)*, pp. 87–92, Bangkok, Thailand, 2012.

[11] M. Subramanyam, "Automatic feature based image registration using SIFT algorithm," in *2012 Third International Conference on Computing, Communication and Networking Technologies (ICCCNT'12)*, pp. 1–5, Coimbatore, India, 2012.

[12] P. Rathnayaka, S.-H. Baek, and S.-Y. Park, "An efficient calibration method for a stereo camera system with heterogeneous lenses using an embedded checkerboard pattern," *Journal of Sensors*, vol. 2017, Article ID 6742615, 12 pages, 2017.

[13] Y. Schröder, A. Scholz, K. Berger, K. Ruhl, S. Guthe, and M. Magnor, "Multiple kinect studies," *Computer Graphics*, vol. 2, no. 4, p. 6, 2011.

[14] Y. Park, S. Yun, C. Won, K. Cho, K. Um, and S. Sim, "Calibration between color camera and 3D LIDAR instruments with a polygonal planar board," *Sensors*, vol. 14, no. 3, pp. 5333–5353, 2014.

[15] H. Yang, X. Liu, and I. Patras, "A simple and effective extrinsic calibration method of a camera and a single line scanning lidar," in *Pattern Recognition (ICPR), 2012 21st International Conference*, pp. 1439–1442, Tsukuba, Japan, 2012.

[16] L. Zhou and Z. Deng, "A new algorithm for computing the projection matrix between a LIDAR and a camera based on line correspondences," in *2012 IV International Congress on Ultra Modern Telecommunications and Control Systems*, pp. 436–441, St. Petersburg, Russia, 2012.

[17] G. Li, Y. Liu, L. Dong, X. Cai, and D. Zhou, "An algorithm for extrinsic parameters calibration of a camera and a laser range finder using line features," in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3854–3859, San Diego, CA, USA, 2007.

[18] K. Kwak, D. F. Huber, H. Badino, and T. Kanade, "Extrinsic calibration of a single line scanning lidar and a camera," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3283–3289, San Francisco, CA, USA, 2011.

[19] R. S. Yang, Y. H. Chan, R. Gong et al., "Multi-Kinect scene reconstruction: calibration and depth inconsistencies," in *2013 28th International Conference on Image and Vision Computing New Zealand (IVCNZ 2013)*, pp. 47–52, Wellington, New Zealand, 2013.

[20] B. Yang, H. Dong, and A. El Saddik, "Development of a self-calibrated motion capture system by nonlinear trilateration of multiple Kinects v2," *IEEE Sensors Journal*, vol. 17, no. 8, pp. 2481–2491, 2017.

[21] J. Salau, J. H. Haas, W. Junge, and G. Thaller, "Extrinsic calibration of a multi-Kinect camera scanning passage for measuring functional traits in dairy cows," *Biosystems Engineering*, vol. 151, pp. 409–424, 2016.

[22] P.-C. Su, J. Shen, W. Xu, S.-C. Cheung, and Y. Luo, "A fast and robust extrinsic calibration for RGB-D camera networks," *Sensors*, vol. 18, no. 1, p. 235, 2018.

[23] W. G. C. W. Kumara, S. H. Yen, H. H. Hsu, T. K. Shih, W. C. Chang, and E. Togootogtokh, "Real-time 3D human objects rendering based on multiple camera details," *Multimedia Tools and Applications*, vol. 76, no. 9, pp. 11687–11713, 2017.

[24] S. Li, P. N. Pathirana, and J. Bonacci, "A general pose estimation algorithm in a multi-Kinect system," in *7th International Conference on Information and Automation for Sustainability*, pp. 1–5, Colombo, Sri Lanka, 2014.

[25] Microsoft, *Kinect for Windows SDK 2.0*2018, https://www.microsoft.com/en-us/download/details.aspx?id=44561.

[26] P. J. Besl and N. D. McKay, "Method for registration of 3-D shapes," in *Proceedings of SPIE 1611, Sensor Fusion IV: Control Paradigms and Data Structures*, pp. 586–607, Boston, MA, USA, April 1992.

[27] M. Reynolds, J. Doboš, L. Peel, T. Weyrich, and G. J. Brostow, "Capturing time-of-flight data with confidence," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference*, pp. 945–952, Colorado Springs, CO, USA, 2011.

[28] R. B. Rusu, Z. C. Marton, N. Blodow, M. Dolha, and M. Beetz, “Towards 3D point cloud based object maps for household environments,” *Robotics and Autonomous Systems*, vol. 56, no. 11, pp. 927–941, 2008.

[29] J. R. Terven and D. M. Córdova-Esparza, “Kin2. A Kinect 2 toolbox for MATLAB,” *Science of Computer Programming*, vol. 130, pp. 97–106, 2016.

[30] P. H. S. Torr and A. Zisserman, “MLESAC: a new robust estimator with application to estimating image geometry,” *Computer Vision and Image Understanding*, vol. 78, no. 1, pp. 138–156, 2000.

[31] M. A. Fischler and R. C. Bolles, “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography,” *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.