Multiview 3D Sensing and Analysis for High Quality Point Cloud Reconstruction

[1]Andrej Satnik, [1]Ebroul Izquierdo, [2]Richard Orjesek
[1]Multimedia Vision Group, School of Electrical Engineering, Queen Mary University of London, United Kingdom
[2]Department of multimedia and information-communication technologies, Faculty of Electrical Engineering, University of Zilina, Slovakia

ABSTRACT

Multiview 3D reconstruction techniques enable digital reconstruction of 3D objects from the real world by fusing different viewpoints of the same object into a single 3D representation. This process is by no means trivial and the acquisition of high quality point cloud representations of dynamic 3D objects is still an open problem. In this paper, an approach for high fidelity 3D point cloud generation using low cost 3D sensing hardware is presented. The proposed approach runs in an efficient low-cost hardware setting based on several Kinect v2 scanners connected to a single PC. It performs autocalibration and runs in real-time exploiting an efficient composition of several filtering methods including Radius Outlier Removal (ROR), Weighted Median filter (WM) and Weighted Inter-Frame Average filtering (WIFA). The performance of the proposed method has been demonstrated through efficient acquisition of dense 3D point clouds of moving objects.

Keywords: Multiview sensing, Kinect v2, calibration, reconstruction, point cloud.

1. INTRODUCTION

Dynamic reconstruction for digital representation and description of 3D objects has been at the centre of research for many years. Nowadays, 3D sensorial devices are becoming pervasive and several approaches have emerged for high quality 3D point cloud acquisition. Among low-cost commercially available hardware and software for 3D cloud point acquisition, analysis and skeleton tracking, the Microsoft Kinect v2 and corresponding SKD, is a popular choice. However, it is not designed for multi-view multi-Kinect point cloud capturing using a single conventional PC. This in turn limits the applicability of such sensors and its software. To overcome this limitation a suitable hardware setting is presented in this paper and further elaborated in section 3.

In systems using multiple Time-of-Flight (ToF) sensors, as the Kinect v2, calibration is critical for accurate 3D point cloud capturing. Most conventional calibration techniques are based on pattern detection from suitable calibration chessboards or scene blobs. They assume that the real size of the calibration pattern is known [1]. However, this assumption can be eliminated for more effective autocalibration, since the transformation between 3D points can be computed by exploiting combinations of several rotation matrices generated by the underlying planar pattern. In this paper, we use such a technique for efficient calibration of several Kinects v2. The corresponding approach is further described in section 4. Once a multi-Kinect v2 platform and its corresponding calibration software is available, 3D point clouds of full real word objects can be captured. Such point clouds are the basis for 3D object surface reconstruction and sevral related techniques can be found in the literature [2], [3]. Unfortunately, such “raw point clouds” are far from reflecting an optimal 3D representation of the dynamic 3D objects. Thus, complex post-processing and analysis is required to achieve high quality 3D representations.

There is a key difference between how the data is acquired and the method used to reconstruct it. By registering a dynamic set of depth images from a moving 3D object, one can obtain an increased point density, and further create a complete point cloud of an indoor environment possibly in real time. Related approaches have been extensively studied and reported in the literature. For instance, in [4] and [5] a multi-view sensing solution with Kinect v2 devices using separate computers connected to a manging server is presented.

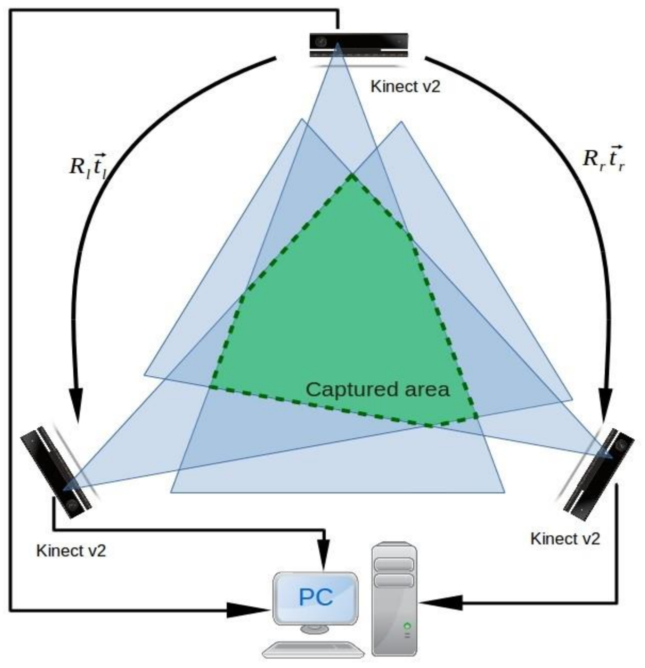


Figure 1. Reconstruction system composed of three Kinect v2 sensors.

However, after the initial point cloud capturing complex post-processing stages are required to achieve high quality representations. In this paper, several state-of-the-art filtering techniques are combined in an efficient manner to achieve this goal. This aspect is further elaborated in section 5. The main novelty of the holistic approach presented in this paper does not reside in any of the individual components but rather on an integrated low-cost hardware and software suite able to deliver high-quality 3D point clouds form complete moving objects. The performance of the introduced system has been assessed in a real-time application and it is presented in section 6.

2. RELATED WORK

Early solutions addressing multi-view reconstruction by Kinect sensors suffered from severe drawbacks derived from the limitations of the available sensing hardware. A multi-Kinect system to seamlessly render a scene from a wide range of angles was described in [6]. The main drawback of related solutions relates to depth inconsistencies between the measurements acquired with different Kinect devices.

The calibration problem has been also studied thoroughly and a large number of related publications are available. The most widespread methods for calibration of extrinsic parameters are based on planar pattern detection [1] and offline Iterative Closest Point (ICP) [7]. The ICP calibration applied in [2] and [5] shows good result with overlapping areas of point clouds. The method presented in this paper is based in the classic planar pattern detection in 3D space described in [4].

3D reconstruction also features prominently in previous works [3], [8]. In most cases fusion of depth data captured by multiple Kinect I sensors is presented. However, such systems rely on Kinect I sensors with lower transfer bandwidth and lower accuracy. Similar systems with multiple Time-of-Flight (ToF) cameras have been previously proposed. Some of them use custom devices [9], that are not commercially available. Furthermore, software originally designed for Kinect I cannot be easily modified to work with Kinect v2. Another challenge with multi-view sensing lays in filtration, correction, and reconstruction of sensed data. In [5], the authors stress the advantage of multi-view sensing using separate PCs since the processing time is reduced by distributing computational load among multiple devices. For computationally demanding algorithms [10], the frame rate becomes too low to achieve real-time processing as described in [2].

3. HARDARE SETTINGS

Multi-sensor system can provide depth image, colour image and audio signal at the same time. As shown in Figure 2, the Prime Sensor encode the IR light and project it to the scene while IR camera capture the IR light and send the signal back to sensor. The camera system estimates distances based on the known speed of light, producing a depth image, where each pixel is encoded with the distance to the corresponding point in the scene. In a Kinect v2 device, the RGB camera captures colour information with a resolution of 1920x1080 pixels. The IR camera is used for the real-time acquisition of depth maps with a 512x424 pixels resolution [11]. With this size of images and the acquisition of framerate up to 30 Hz, the required transfer bandwidth is substantial, closer to 3Gbps. According to a software bandwidth requirement, Kinect's isochronous endpoint will reserve 2.47Gbps bandwidth with RGB consumption 0.25Gbps.

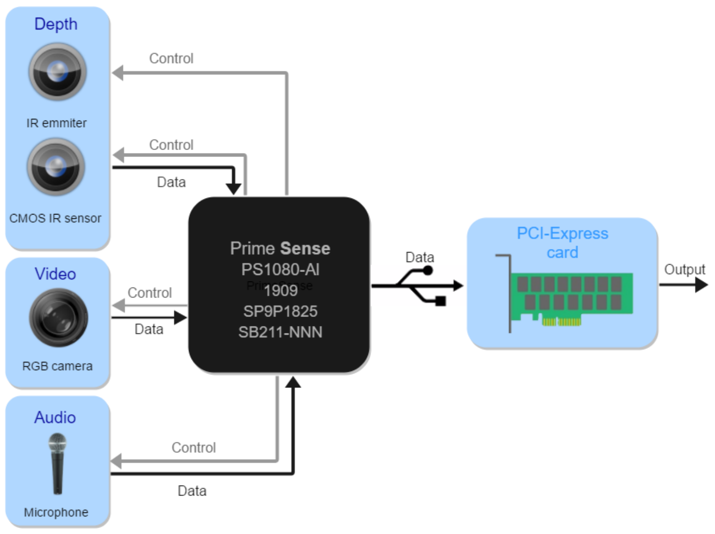


Figure 2. Prime sense diagram of Kinect v2.

The projector of a Kinect v2 emits infrared coded light with modulated waves and reflecting off surfaces in front of the sensor array, where the frequency is modulated randomly. Infrared light is detected by the infrared camera. The pre-processor in the Kinect v2 calculates the distance from the array to the reflecting surfaces. In a multi-Kinect v2 setting, each device can interfere with the neighbouring devices. To avoid such interference, Kinects v2 sensing must be triggered one by one. This process takes about 1.5 seconds in each case.

As stated before, the number of Kinects v2 devices that can be connected to a single PC is limited by the number of internal USB buses in the PC system. A single bus can handle multiple Kinect streams, as long as the high-speed serial computer expansion bus card (PCI-Express) is able to cope with the correspondingly data transfer stream. An important aspect of the work presented in this paper is the proposed hardware and software configuration to enable the connection of multiple Kinect v2 devices to a single adequately adapted PC. These problems are solved by using libfreenect2 driver [12] to unwrap multi-frequency phase estimates for time-of-flight ranging by using kernel density estimation (KDE) in a spatial neighbourhood [13]. Additionally, data transfer with predefined bandwidth reservation requires extension of the USB Filesystem (USBFS) memory buffers determined entirely by the controller's firmware.

4. CALIBRATION

The camera's extrinsic matrix describes the camera's location in the world and what direction it is pointing. It has two components, a rotation matrix R_{3x3} and a translation vector $\vec{t} = (x, y, z)$. For each device, an extrinsic matrix $R\vec{t}$ has to be calculated. Following the approach presented in [4], we calibrated the relative pose between Kinect v2 sensors using the IR camera of each Kinect. In Figure 3, the captured IR images from Kinect v2 sensors with a standard chessboard planar object is depicted. The underlying process is outlined next.

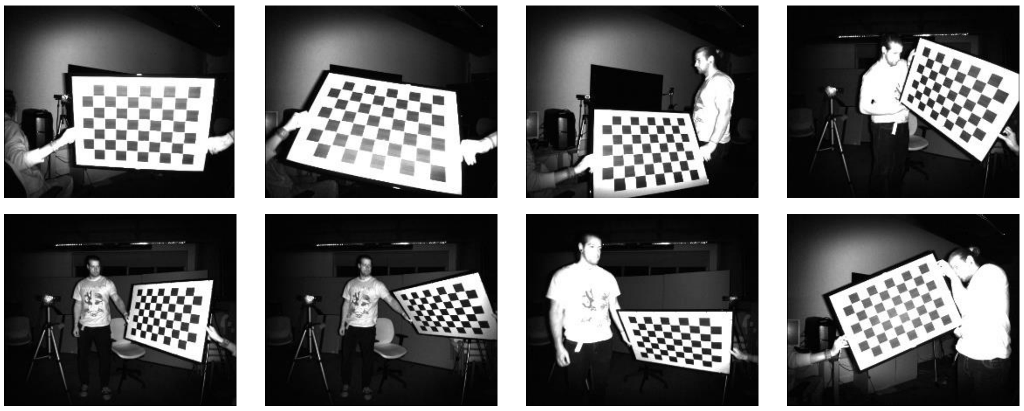


Figure 3. Calibration patterns captured with IR camera for left and right Kinect v2 sensor.

First, we find a 3D point representation for each Kinect v2 to calculate the rotation matrix *R*. We mainly detect representative corner points P_{Mx2} on image captured from the IR cameras using a chessboard planar pattern similar as the one described in [1]. This approach provides features that can be computed as intersection of lines with sub-pixel accuracy. For a given undistorted depth map and IR camera with focal length f_x,f_y and coordinates of the principle point c_x,c_y we the estimate the 3D coordinates of points $Q_{ij} = (x_{ij}, y_{ij}, z_{ij})$ for each Kinect:

$$x_{ij} = z_{ij} \frac{j + 0.5 - c_x}{f_x} \tag{1}$$

$$y_{ij} = z_{ij} \frac{i + 0.5 - c_y}{f_y} \tag{2}$$

Where *i*, *j* are pixel positions of corner points. In the next step, we find a covariance matrix *H* between two sets of 3D points $Q_A = (x, y, z)$ and $Q_B = (x, y, z)$ by expressing:

$$H = \sum_{i=1}^{N} (Q_A^i - C_A)(Q_B^i - C_B), \;\; Q_A \neq Q_B \tag{3}$$

Where *N* is number of corner points and C_A, C_B are centroids of Q_A and Q_B. Using Singular Value Decomposition (SVD) [14] rotation matrix *R* can be expressed:

$$H = U\Sigma V^T \tag{4}$$

$$R = V \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & |VU^T| \end{bmatrix} U^T \tag{5}$$

Where *U* is an orthogonal matrix whose columns are the eigenvectors of HH^T, *V* is an orthogonal matrix whose columns are the eigenvectors of H^TH, and Σ is a diagonal matrix represented by singular values $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0$. The determinant $|VU^T|$ corrects rotation matrix *R* to ensure a right-handed coordinate system. The translation vector $\vec{t}$ is calculated as the difference of centroids C_A and C_B.

$$\vec{t} = -RC_A^T + C_B^T \tag{6}$$

5. POST-PROCESSING FOR THE GENERATION OF HIGH QUALITY 3D POINT CLOUDS

The quality of the depth image is a key factor for model generation and reconstruction. The infrared projector of time-of-flight cameras (ToF) emits the pseudo random pattern light by a diffractive mask [15], which cause a distortion in the depth channel generating distorted depth data. This in turn leads to inaccurate point clouds and subsequently to artefacts in the reconstruction. Our goal is to improve the captured raw points and smooth out noisy data. In this section, we considered to use a several existing filtering approaches.

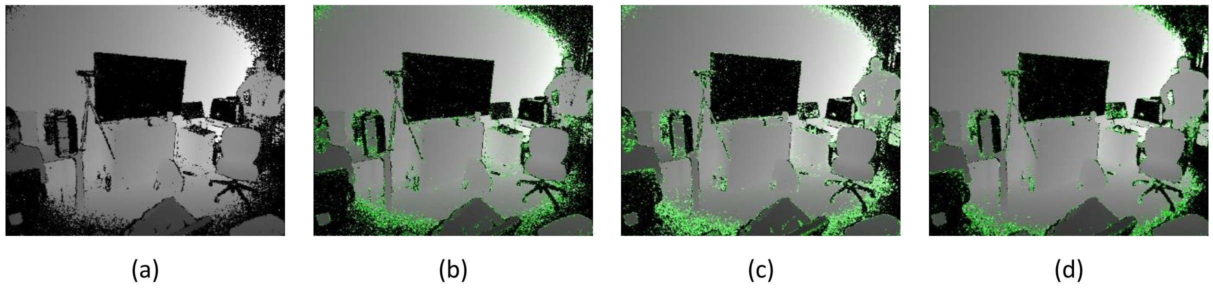


Figure 4. Filtration results of depth map, where green colour represents differences between filtration approaches: (a) Raw depth. (b) Depth filtered by BF (c) Depth filtered by BF +WM (w=5x5) (d) Depth filtered by BF + WM(w=5x5) + ROR(w=5x5, n=10).

Our work builds on selected approaches from the literature. Starting with bilateral filtering (BF) [15], [16], points at specific positions are processed using this edge-preserving smoothing filter. The BF filter is applied to points with invalid depth values before "flying points" are removed as shown in Figure 4b. In the next step, outlier removal is performed exploiting a combination of Weighted Median filter (WM) and Radius Outlier Removal filter (ROR). Depth pixel witch zero value are processed by the WM, where weights are set to *0* for each invalid pixel in region of interest. As outlined in Algorithm 1, the ROR is defined according to a radius given by

$$r_d = z_d \frac{0.5 - c_x}{f_x} - z_d \frac{2r_n + 0.5 - c_x}{fx} \tag{7}$$

Where r_n is size of window in pixels, c_x and f_x are intrinsic parameters of the Kinect v2 (see Figure 4d). Additionally, we also applied a Weighted Inter-Frame Average filter (WIFA) that buffers multiple samples of previous frames $t = \{1,2,...N\}$, where N is number of frames. This approach removes spatial noise in the depth channel of the actual frame $D_{t=0}$, when the accuracy decreases quadratically with increasing depth [17].

Algorithm 1: Fusion of several filtering methods (WM+ROR+WIFA) of depth map.

```
1:  Function DMF($D,r,nthresh$) ▷ where $D_t$ - input depth map, nthresh - number of neighbours
2:  Let $x$ and $y$ be current index of pixel
3:  $n = 0$
4:  if $D_{t=0}[x][y] == 0$ then
5:      $MED(D_{t=0}[x][y])$
6:  for $i = x-rn$ to $x+rn$ do
7:      for $j = y-rn$ to $y+rn$ do
```

```
8:          if $D_{t=0}[x][y]$ 6= 0 then
9:              $M = abs(D_{t=0}[x][y]-D_t[i][j])$
10:             if $M < r_d$ then
11:                 $n = n+1$
12:     if $n < nthresh$ then
13:         $D_{t=0}[x][y] = 0$
14:     $WIFA(D_t[x][y], D_t=\{1,2,..._N\}[x][y])$
```

The WIFA is based on weighting of value by calculation of differences between depth, where high differences between depth values in previous frames are weighted to *0*. Motion blurs makes the application of WIFA unsuitable for highly dynamic scenes. Hence, to prevent blurring, the condition

$$\left|D_{t=0} - D_{t=\{1,2,...N\}}\right| < kD_{t=0} \tag{8}$$

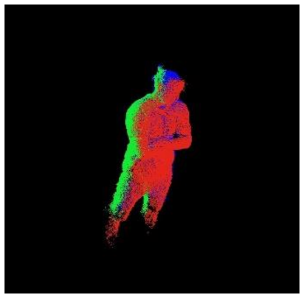
is enforced. Here, where k represents a threshold derived from the depth accuracy of the Kinect v2 [18]. Point clouds from all Kinect v2 in the system are subsequently fused to generate a single uniform point cloud representation of the scene (see Figure 5a). The reconstruction algorithm is used to produce a faithful and detailed representation of the scanned shapes surface. Our approach is divided on two phases. First, generation of point cloud and the second is indexation of points which lead to description of polygons in the target mesh. In a second phase, a triangular mesh of the captured 3D object is generated as follows:

Considering depth image represents organized 3D point cloud where adjacent points are connected, pixels at the right, bottom and right-bottom generate triangles by connecting the corresponding 3D points unless 3D Euclidean distance between them is higher than threshold *t* [2].

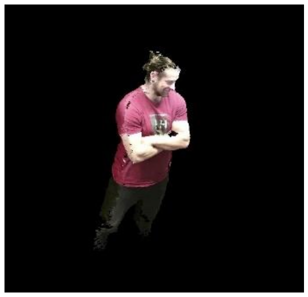
The main idea of threshold *t* is to detect discontinuities between adjacent points. Two factors affect threshold. The depth resolution of sensor [2] and spatial noise increasing with the depth. For presented results, the value of the threshold was fixed and set equal to *t = 10 cm.*

For each pixel $p = I(x, y)$ and adjacent pixels $p_R = I(x+1, y)$, $p_B = I(x, y+1)$, $p_B = I(x+1, y+1)$ triangles can be denoted as $T_1 = \{p, p_R, p_B\}$ if $|p - p_R| < t \wedge |p - p_B| < t \wedge |p_R - p_B| < t$. Similarly, second triangle can be denoted as $T_2 = \{p_R, p_B, p_{RB}\}$. Example of reconstructed point cloud is depicted in Figure 5c. For reasons of wrong sensing caused Kinect sensor, some pixels in depth image contain defective points represented as black pixels $I(x, y) = 0$. For this reason, triangle is not retrieved.

(a)



(b)


(c)

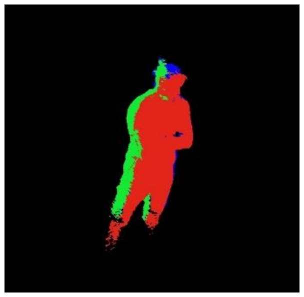

(d)

Figure 5. Reconstruction of scene from three Kinect v2 sensors: (a) Fused point cloud generated by each Kinect. (b) Fused point cloud, where Red, Green and Blue points represent each Kinect. (c) Fused mesh calculated from fused point cloud. (d) Fused mesh, where Red, Green and Blue points represent each Kinect.

6. EXPERIMENTAL EVALUATION

In this section, we present experimental results of the proposed calibration method and time load of different post-processing methods leading to mesh reconstruction in real-time. For our experiments, we used CUDA parallel computing platform with GeForce 780GTX graphic card using 1580 cores and connection of 3 Kinect v2 RGB-Depth sensors. The project was written in C++ with operating system Linux Ubuntu 16.04LTS. Each Kinect v2 sensor is connected to single machine by using USB 3.0 PCI-Express enhancement card with maximum transfer rate 5Gbps and libFreenect2 driver [12]. Kinects were distributed around the specific area as is depicted in Figure 1.

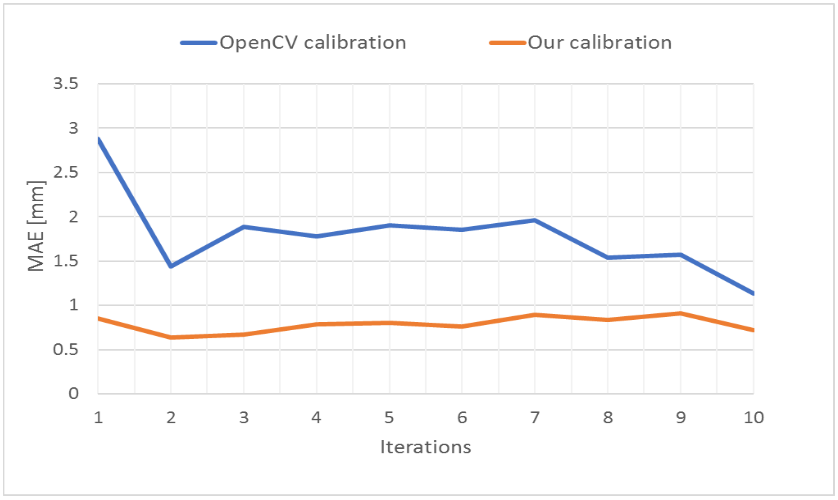


Figure 6. Mean absolute error of two calibration methods.

We calibrated our system with double sided planar pattern with size 6x9 corners and 5cm corner distance to find extrinsic parameters $R\vec{t}$ for each Kinect v2 IR camera. We compared the performance of our calibration approach with Zhang's method by calculating Mean Absolute Error (MAE) between detected 3D corner points Q_A and Q_B for every iteration used to estimate R and $\vec{t}$ (see Figure 6).

$$MAE = \frac{1}{P}\sum_{i=0}^{P}\left|Q_A^i - Q_B^i\right| \tag{9}$$

We decided to evaluate calibration in distance of 4.5 meters from depth sensor. Depth data beyond this distance yields inconsistent and lead to false calculations of extrinsic parameters.

Table 1. Performance Test for Each Module.

Process	Time[ms]	Load [%]
Weighted Median filter	1.69	5.66
Radius Outlier Removal	1.55	5.18
Interframe Average filter	1.88	6.28
Acquisition of point cloud	3.18	10.66
Mesh reconstruction	5.88	19.71
Overall	14.18	47.49

To test the performance of each post-processing approach, we captured 400 depth frames from Kinect v2 sensor. As was presented in [2], the execution time were calculated by averaging time of 400 frames for each depth sensor as is shown in Table 1. We tested the load for three Kinects v2 sensors by averaging computation time with result 43.71ms.

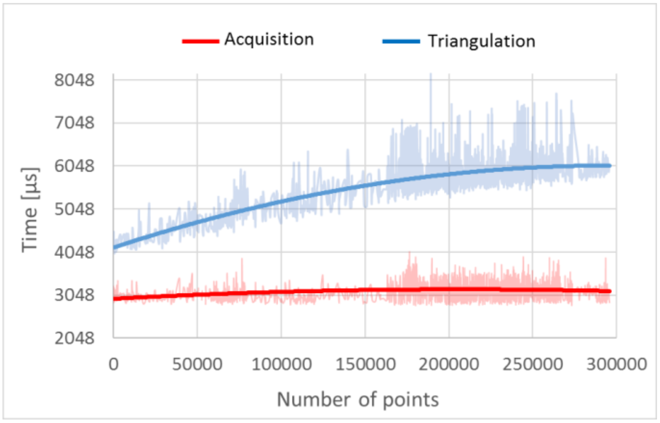


Figure 7. Performance test of acquisition and triangulation of point cloud.

In the next part of experiment, we measured a power load of modules with increasing number of reconstructed data (see Figure 7 and Figure 8) by capturing more than 2 000 depth frames. For our experiments, we set window size of Weighted Median filter (WM) to $w_{WM} = 3x3$, window of Radius Outlier Removal (ROR) to $w_{ROR} = 5x5$ and for Weighted Inter-Frame Average filter (WIFA) we set buffering of frames to $t = 2$.

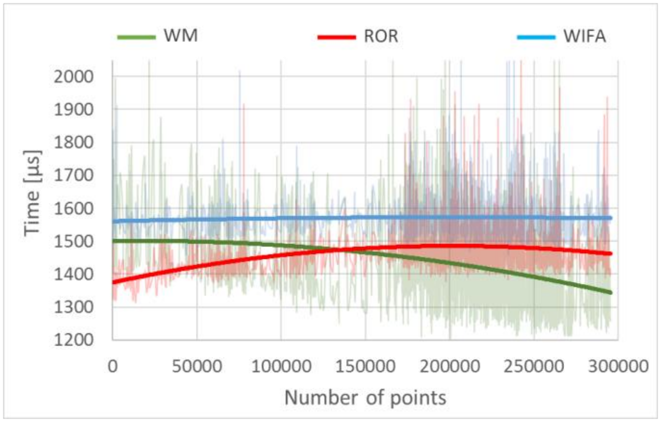


Figure 8. Load of filtration methods for depth map processing.

7. CONCLUSION

In this paper, we have proven a possibility of multiple connection of Kinect v2 depth sensor with low cost hardware on single machine. All the details of the connection of multiple Kinect capturing system with the applied methods for accurate external calibration, were explained. The approach is based on the generation of separate point clouds from multiple depth streams, calibration based on planar pattern detection and post-processing algorithms for the creation of a single full 3-D mesh.

As we have demonstrated, our calibration method was getting better precision in first iterations as long as planar pattern remains in distance of 4.5 meters from depth sensor due to spatial noise. Additionally, we measured a performance of real-time capturing, important post-processing modules and triangulation of point cloud with three Kinect v2 sensors.

In future, we also plan to perform more complex algorithms with aim to improve and extend the proposed framework.

REFERENCES

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

[2] D. S. Alexiadis, D. Zarpalas and P. Daras, "Real-Time, Full 3-D Reconstruction of Moving Foreground Objects From Multiple Consumer Depth Cameras," *IEEE Transactions on Multimedia,* vol. XV, no. 2, pp. 339 - 358, 21 November 2012.

[3] D. S. Alexiadis, D. Zarpalas and P. Daras, "Real-time, realistic full-body 3D reconstruction and texture mapping from multiple Kinects," in IVMSP Workshop, Seoul, South Korea, 10-12 June 2013.

[4] P. Palasek, H. Yang, Z. Xu, N. Hajimirza, E. Izquierdo and I. Patras, "A flexible calibration method of multiple. Kinects for 3D human reconstruction," in *IEEE International Conference on Multimedia & Expo Workshops (ICMEW),* Turin, Italy, 29 June-3 July 2015.

[5] M. Kowalski, J. Naruniec and M. Daniluk, "Livescan3D: A Fast and Inexpensive 3D Data Acquisition System for Multiple Kinect v2 Sensors," in International *Conference on 3D Vision (3DV)*, France, October 2015.

[6] R. S. Yang, Y. H. Chan, R. Gong, M. Nguyen, A. G. Strozzi, P. Delmas, G. Gimel'farb and R. Ababou, "Multi-Kinect Scene Reconstruction: Calibration and Depth Inconsistencies," in *28th International Conference of Image and Vision Computing New Zealand (IVCNZ),* Wellington, New Zealand, 27-29 Nov. 2013.

[7] P. Besl and N. D. McKay, "A method for registration of 3-D shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. XIV, no. 2, pp. 239 - 256, February 1992.

[8] D.-M. Córdova-Esparza, H. Jiménez-Hernández and J. R. Terven, "A multiple camera calibration and point cloud fusion tool for Kinect V2, Ana Marcela Herrera-Navarro," *Science of Computer Programming,* vol. CXLIII, p. 1–8, December 2016.

[9] D. Yong, C. Lei, W. Yucheng, Y. Min, Q. Xiameng, H. Shaoyang and J. Yunde, "A real-time system for 3D recovery of dynamic scene with multiple RGBD imagers," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, Colorado Springs, CO, USA, 20-25 June 2011.

[10] M. Berger, A. Tagliasacchi, L. M. Seversky, P. Alliez, J. A. Levine, A. Sharf and C. T. Silva, "State of the Art in Surface Reconstruction from Point Clouds," in *The Eurographics Association*, France, April 2014.

[11] E. Lachat, H. Macher, M.-A. Mittet and P. Grussenmeyer, "First experiences with kinect V2 sensor for close range 3D modelling," *The International* Archives *of the Photogrammetry, Remote Sensing and Spatial Information Sciences,* Vols. XL-5/W4, 25-27 February 2015.

[12] X. Lingzhu, E. Florian, K. Christian, W. Thiemo, H. Lars and Alistair, "libfreenect2: Release 0.2.," in *doi:10.5281/zenodo.50641*, (2016, April 28).

[13] F. J. Lawin, P.-E. Forssén and H. Ovrén, "Efficient Multi-frequency Phase Unwrapping Using Kernel Density Estimation," in *ECCV 2016,* Amsterdam, The Netherlands, 2016.

[14] Y. B. Jia, "Singular Value Decomposition," Department of Computer Science, Iowa State University, Ames, Iowa, USA, 2013.

[15] J. Fu, S. Wang, Y. Lu, S. Li and W. Zeng, "Kinect-like depth denoising," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, Seoul, South Korea, 20-23 May 2012.

[16] L. Chen, H. Lin and S. Li, "Depth image enhancement for Kinect using region growing and bilateral filter," in *21st International Conference on* Pattern *Recognition (ICPR 2012)*, Tsukuba, Japan, November 11-15, 2012.

[17] T. Mallick, P. P. Das and A. K. Majumdar, "Characterizations of Noise in Kinect Depth Images: A Review," *IEEE Sensors Journal*, vol. XIV, no. 6, pp. 1731 - 1740, March 2014.

[18] P. Fürsattel, S. Placht, M. Balda, C. Schaller, H. Hannes, M. Andreas and R. Christian, "A Comparative Error Analysis of Current Time-of-Flight Sensors," *IEEE Transactions on Computational Imaging,* vol. II, no. 1, pp. 27-41, 18 December 2015.

[19] G. Han and W. Song, "Motion capture of maintenance personnel based on multi-Kinect," in *International Conference on Quality, Reliability, Risk,* Maintenance*, and Safety Engineering (QR2MSE)*, Chengdu, China, 15-18 July 2013.