International Journal of Computer Applications (0975 – 8887)
Volume 167 – No.13, June 2017

FRANSAC: Fast RANdom Sample Consensus for 3D Plane Segmentation

Ramy Ashraf Zeineldin
Computer Science and Engineering Dept.,
Faculty of Electronic Engineering,
Menofia University
Menof, Egypt

Nawal Ahmed El-Fishawy
Computer Science and Engineering Dept.,
Faculty of Electronic Engineering,
Menofia University
Menof, Egypt

ABSTRACT

Scene analysis is a prior stage in many computer vision and robotics applications. Thanks to recent depth camera, we propose a fast plane segmentation approach for obstacle detection in indoor environments. The proposed method Fast RANdom Sample Consensus (FRANSAC) involves three steps: data input, data preprocessing and 3D RANSAC. Firstly, range data, obtained from 3D camera, is converted into 3D point clouds. Next, a preprocessing stage is introduced where a pass through and voxel grid filters are applied. Finally, planes are estimated using a modified 3D RANSAC. The experimental results demonstrate that our approach can segment planes and detect obstacles about 7 times faster than the standard RANSAC without losing the discriminative power.

Keywords

RANSAC, point cloud, plane segmentation, Kinect, RGB-D, Voxel

1. INTRODUCTION

Understanding the structural information of the surrounding environment is a principal issue for most computer vision applications, robots, and wearable obstacle avoidance devices. Man-made environments consist of many planes which are the basic units of objects. Plane detection and segmentation is the fundamental technique for understanding such scenes and can be used in many important applications, such as 3D reconstruction [1]–[6], object recognition [7]–[9], virtual reality [10], [11] and 3D mapping [12].

Undoubtedly, the accuracy of plane segmentation is highly related to the performance of the whole scene understanding system. In addition, plane segmentation should be processed in real-time as the basic step of scene understanding systems. Recently, three-dimensional (3D) sensors have emerged which help acquiring data about the surrounding environment. A comparison of the most common sensors is listed in Table 1. One example of 3D sensors is the LASER scanners [1] which can cover large areas accurately. However, LASER scanners can only process few frames per second (FPS). Furthermore, LASER scanners are also very expensive and require high power.

Moreover, another example of 3D sensors is structured light scanners [2] where a light pattern is projected on the object before the distortion between the received and the projected light is calculated. Based on triangulation structured light scanners can give a depth map with high resolution (640 x 480 pixels). Structured light scanners are recently popular in computer vision applications because they are relatively low cost, low power and give higher frame rate 30 FPS and more. Unfortunately, structured light scanners are limited to few meters unlike LASER scanners.

Table 1. Comparison of various 3D sensors

Graphics	Range	Speed	Cost
Laser	High	Slow	High
Structured light	Medium	Fast	Low
ToF	Low	Very Fast	Medium

Time of Flight (ToF) scanners is another choice for acquiring 3D point clouds. Photonic Mixer Devices [3] (PMD), for example, uses pulsed LEDs for illumination. The principle of PMD is modulating the outgoing beam with an RF carrier and detects the phase shifted beams by the receiver. Although large number of frames can be processed (120 FPS), its range is limited to 1 meter with small resolution (160 x 120 pixels).

This paper presents a fast RANSAC enhancement for 3D plane segmentation using structured light sensors. Firstly, the depth image is obtained from the sensor. Then, it is converted to a 3D point cloud. After that a preprocessing stage is applied to the point cloud where a pass through and voxelization filters are used. The segmentation is then performed using the 3D enhanced RANSAC.

The paper is organized as follows: The following section includes a brief overview of the RANSAC. Section 3 discusses previous work related to our research. Then our proposed approach Fast FRANSAC is shown in Section 4. Section 5 includes methodology and experimental analysis of our proposed approach. Finally, the paper is then concluded in Section 6.

2. THE RANSAC

Many plane segmentation algorithms have been proposed in recent years. Each segmentation method has its own pros and cons. Most significantly, the RANSAC is one of the most commonly used algorithms for plane detection which is firstly introduced by Fischler and Bolles [13] for 2D estimations. The RANSAC is a global iterative method that robustly finds model parameters from a set of data points. In addition, the RANSAC treats depth data as 2d images where each pixel is in the range of 0 to 255.

Figure 1 shows an example of applying RANSAC for 2D line fitting problem. By assuming data as a collection of inliers, points belong to the line, and outliers, points outside the line, the RANSAC can robustly estimate the parameters of planes with high degree of accuracy even number of outliers exceed 50% of the sample points. Unlike other statistical sampling techniques such as M-estimators and least-median squares [14], [15] that use as much as possible of the data, the RANSAC uses the smallest data set (starting from three points for a plane model) and proceeds to enlarge this set with consistent inliers.

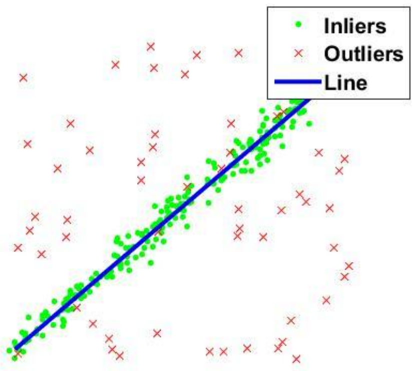

Fig 1: 2D line fitting using RANSAC.

The RANSAC algorithm simply consists of two steps: a hypothesis stage, where a random selection of inliers and computing the model parameters, and an evaluation stage, where a verification of the hypothesized model to the entire dataset is done.

In the hypothesis step, the RANSAC randomly selects a subset of data points, then the parameters of the mod-el are estimated from the sample points. After the model parameters have been estimated, an evaluation is essential to ensure that the candidate model is the best one exists, which is supported by the largest number of inlier candidates. Likewise, a model is considered as an inlier model when the distance error is within a predefined threshold (d) that can be calculated as the distance from point P = (x_1, y_1, z_1) to the plane $Ax + By + Cz + D = 0$ from the following equation:

$$d \leq \frac{|Ax_1 + By_1 + Cz_1 + D|}{\sqrt{A^2 + B^2 + C^2}} \quad (1)$$

3. RELATED WORK

Although RANSAC is one of the most distinct plane detection algorithms, RANSAC suffers from one major drawback. RANSAC is a heavy computation approach that requires many processing time cycles. Consequently, several enhancements have been proposed to eliminate this problem that are figured out in the following subsections.

3.1 Randomized RANSAC (RRANSAC)

A new randomized version of the RANSAC is introduced by Matas and Chum [16]. RRANSAC enhances the time for evaluating the hypothesis (TE) by adding a preliminary test ($T_{d,d}$ Test) before the hypothesis evaluation stage.

Hypothesis evaluation is first performed using a small number of data points d from the total of N points (where d << N). The $T_{d,d}$ test is passed if all d data points out of d randomly selected are consistent with the hypothesized model. Setting of the length d of the $T_{d,d}$ test to 0, means standard RANSAC, where setting of d = 1 is recommended as the optimal value.

3.2 The Voxel RANSAC

Researchers in [17], [18] proposed a preprocessing stage. By applying this filtering process, the amount of data points is reduced without losing the main features of the point cloud. As a result, the computational speed is significantly increased. The used down sampling filter is a voxel grid filter.

A voxel grid filter is a set of small 3D boxes that have identical size. Firstly, this filter is applied to all data points of the point cloud. Then, in each 3D box, all data points in each 3D box are approximated with their equivalent center. Using the centroid could be costlier than using the center of the cube, but this maintains the geometric features of the whole point cloud.

3.3 Hardware accelerated RANSAC approaches

In contrast to the mentioned earlier approaches that enhance the speed of RANSAC using some modifications in the software, some methodologies make use of the advances in the hardware as well. In [19], different parallel implementations based on OpenMP, POSIX Threads, and CUDA. The results show that CUDA is the best choice. However, in such situations where no GPU exists, using POSIX threads is a better choice compared with OpenMP because POSIX allows programmers to manage and control thread more directly, and hence boosting their performance. Unfortunately, this approach is not suitable for robotics and embedded devices thanks to its high power consumption.

Other approaches in [20]–[22] implements a Field Programmable Gate Array (FPGA). A hardware architecture and organization of the RANSAC for feature-based image registration are proposed in [20]. Additionally, a hardware/software co-design platform of RANSAC algorithm for real-time affine geometry estimation is presented in [21]. The most intensive computation, i.e. fitness scoring task, is performed on the hardware by using double buffering technique to enable process pipelining. In [22], the proposed system is capable to perform tasks extremely fast due to its high level of parallelism.

4. FAST RANSAC (FRANSAC)

Fast RANdom Sample Consensus (FRANSAC), as demonstrated in Fig. 2, consists of three main stages: Firstly, the range data is obtained from the sensor then it is converted to 3D point cloud. After that, a preprocessing stage is applied to the point cloud where pass-through filter and voxelization are used. Thirdly, the ground plane is then segmented using the 3D RANSAC.

4.1 Data Input Stage

The input of our proposed approach comes from a 3D structured light sensor, for instance, the Xbox Kinect which consists of two cameras and IR projector. One of the cameras is a standard RGB camera, while the other camera is an IR camera which, along with the IR projector, forms the 3D image. Thus, the depth is calculated using the disparity of each pixel using the projected pattern and the IR camera.

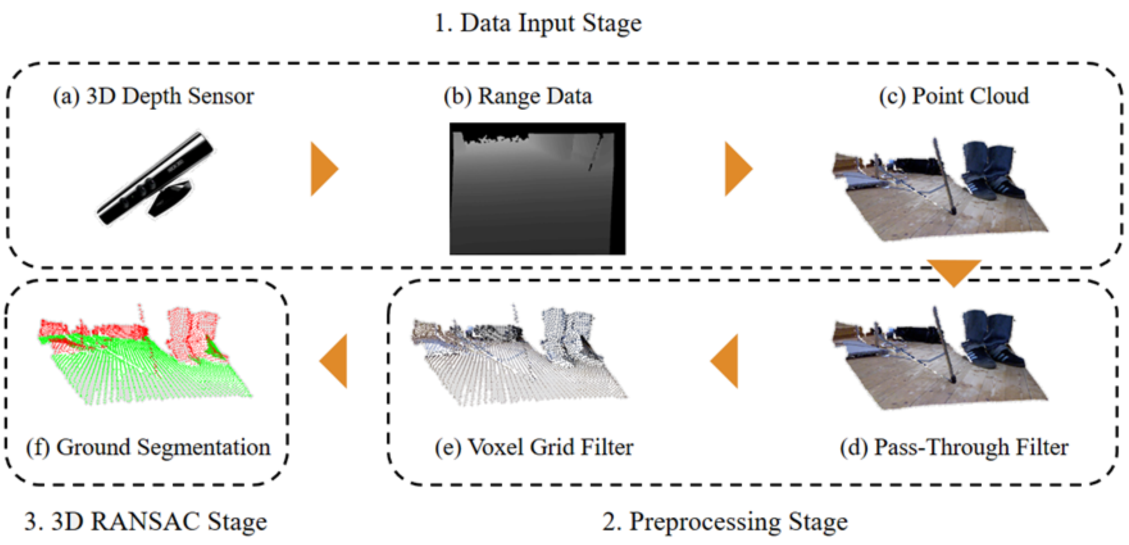

Fig 2: Our proposed approach FRANSAC

The Kinect gives a 2D depth map in which each pixel holds the distance information. After that, this map is converted to a point cloud representation which is a set of data points in some coordinate system. In a 3D coordinate system, X, Y, and Z coordinates usually define these points, and often are intended to represent the external surface of an object.

4.2 Preprocessing Stage

The Kinect sensor can provide a depth data with 307200 (640x480) points. However, this is a large number of in-formation to process even with simple operations. The complexity will be O(n), n is the number of points, if we perform a simple operation on every point of the cloud. Therefore, a reduction of the number of points is essential to eliminate complexity, processing time and efficiency problems. As a consequence, two filtering operations are introduced which are Pass Through and Voxel Grid filters.

4.2.1 Pass Through Filter

The accuracy of the sensor depends on the distance, for example, far objects suffers from lower accuracy than near ones. In order to solve that limitation, a pass through filter is introduced to remove inaccurate far data points. It passes points in a point cloud based on constraints for one particular field, for such as Z-axis, by iterating through the entire input cloud, and automatically filters non-relevant points. In fact, this stage is very vital to remove unnecessary objects far away from the Kinect.

4.2.2 Voxel Grid Filter

After using a pass through filter, the whole point cloud is then down sampled giving back an equivalent point cloud with fewer points. A 3D voxel grid filter is used for this purpose which divides the input point cloud into small 3D boxes with a line length of 3 cm. Then the points in each voxel (3D box) are down sampled to a single point. As a result, there are two options as to how to represent the distribution of points in a voxel by a single point: first option is to take the centroid or spatial average of the point distribution. Second option is to take the geometrical center of the voxel. The first option is more accurate since it takes into account the point distribution inside the voxels. However, it is more computationally intensive since the centroid must be computed for each voxel. The computational cost increases linearly with the number of points in the cloud and the number of voxels.

4.3 Plane Segmentation using 3D RANSAC

The next step is to identify the most representative planes of the scene from the point cloud. The algorithm used for plane detection is RANSAC (RANdom SAmple Consensus) which simply consists of two steps: a hypothesis stage, where a random selection of inliers and computing the model parameters, and an evaluation stage, where a verification of the hypothesized model to the entire dataset is done.

In the hypothesis stage, The RANSAC algorithm provides a robust estimation of the dominant plane parameters, performing a random search in the space of solutions. After that, the RANSAC randomly selects a subset of data points, before the parameters of the model are estimated from the input points. If the given model is plane, $Ax + By + Cz + D = 0$, and $M = [A, B, C, D]^T$ are the parameters to be estimated. In contrast to common regression techniques such as least square method, the RANSAC is a resampling technique that generate candidate solutions using the smallest number of points. In other words, the RANSAC converts the estimation problem from the continuous domain to the discrete domain.

In order to obtain a good plane, the RANSAC loops for number of iterations Nit, which can be obtained from the following equation:

$$N_{it} = \frac{\log(1-p)}{\log(1-q^S)} \quad (2)$$

Where p is the probability of finding a good plane from the input points, q is the probability that a point is an inlier, and S is the number of points in the sample.

After the hypothesis stage, the RANSAC evaluates the candidate hypotheses to find the most suitable one, which is supported by the largest number of inlier candidates. Input data is considered inliers if only they fall below a predefined distance threshold (d), as given in Eq. (1).

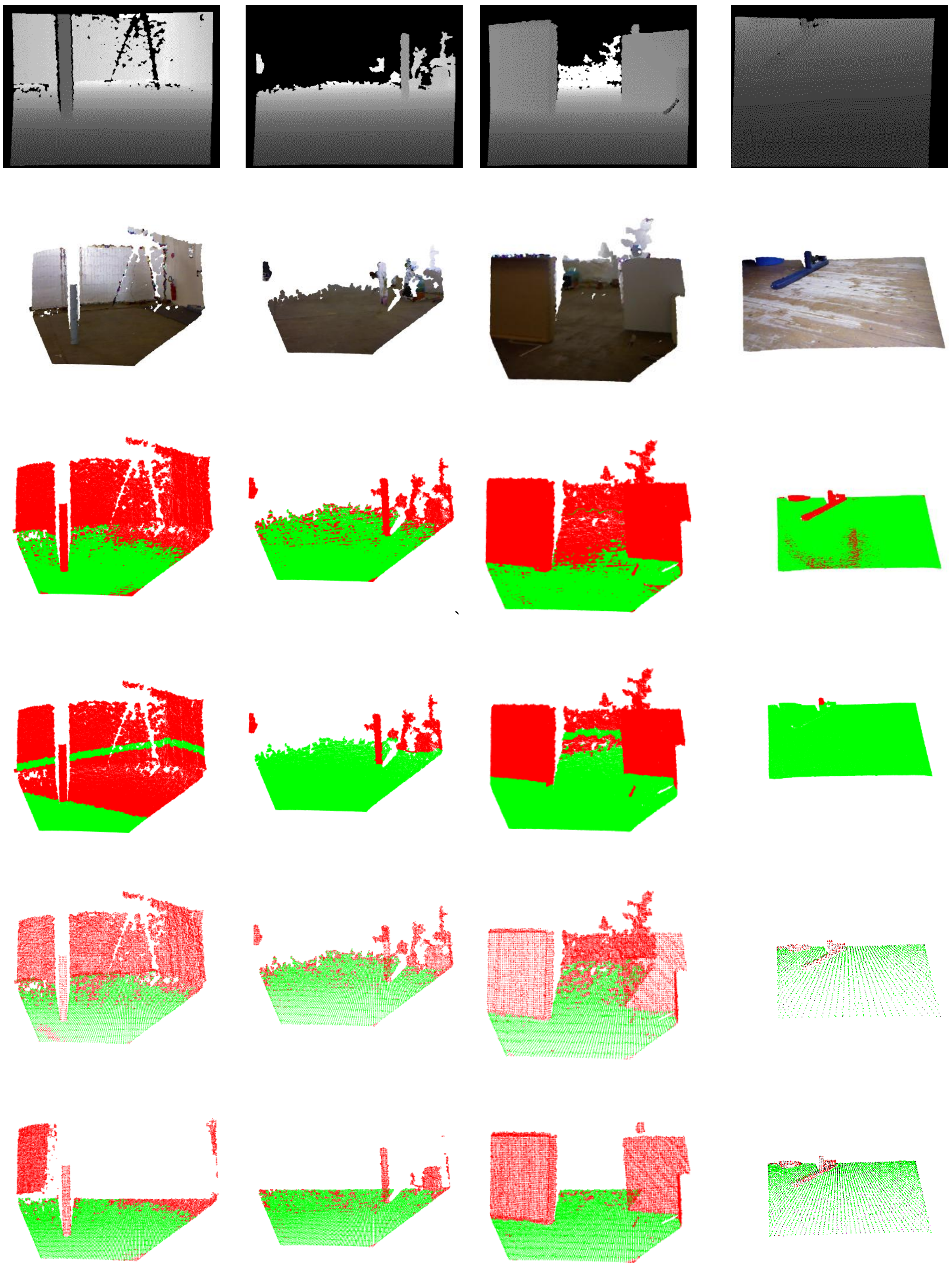
Fig 3: Example of real data testing. From the top: Original depth frame; Input point cloud; Detected planes by RANSAC; Detected planes by RRANSAC; Detected planes by Voxel RANSAC; Detected planes by FRANSAC

Fortunately, the RANSAC does not have to extensively evaluate all the input data points, since two termination criteria can be used before that. Firstly, the evaluation process may finish if the probability of finding a better model than the current best candidate falls below a predefined threshold. Secondly, the termination could be achieved if the number of evaluated samples exceeds the number expected to select an uncontaminated sample

5. EXPERIMENTAL ANALYSIS

5.1 Test Data

The TUM dataset [23] is used which contains a large number of color and depth images of a Microsoft Kinect sensor recorded at full frame rate (30 fps) with 640x480 resolutions. The selected point clouds have a gradual complexity in the sense of the number of objects.

5.2 Initialization Parameters

Some parameters are common between all mentioned algorithms such as number of selected inliers (n), maxi-mum number of iterations (I) and distance threshold (d), as given in Table 2.

Table 2. RANSAC parameters

Parameter	Value
Number of random inliers (n)	3
Maximum number of iterations (I)	1000
Distance threshold (d)	1 cm

Moreover, our algorithm FRANSAC filters out all points with Z values not in the [0-4.5] meters range. The size of every voxel is 2x2x2 cm3, which means only one point per every 8 cm3 will survive. Table 3 shows additional parameters for FRANSAC.

Table 3. Supplementary FRANSAC parameters

Parameter	Value
Pass through limits	0-4.5 m
Size of voxels	2x2x2 cm^3

5.3 Experimental environment

The algorithms are all implemented with C++ under the Linux Ubuntu 14.04 LTS operating system. A personal computer with Intel Core i7, 2.4 GHz CPU, 8GB memory is used for the testing. The ground truth of ground segmentation for quality evaluation is obtained through manual editing.

5.4 Evaluation metrics

The metrics utilized to evaluate FRANSAC and the compared algorithms are computation time, precision, recall, and f1-score.

Precision, also called Confidence, is the ratio of Predicted Positive points to the total number of retrieved points. Precision can be calculated as follows:

$$\text{Precision} = \frac{\text{TP}}{\text{TP}+\text{FP}} \tag{3}$$

Recall, also called Sensitivity, is the proportion of Real Positive points that are correctly Predicted Positive. Re-call can be calculated as follows:

$$\text{Recall} = \frac{\text{TP}}{\text{TP}+\text{FN}} \tag{4}$$

F-Score, also called F-measure or F1, is a measure of the test's accuracy. It considers both the precision (P) and recall (R) as follows:

$$\text{F}-\text{Score} = 2\cdot\frac{\text{P}\cdot\text{R}}{\text{P}+\text{R}} \tag{5}$$

6. RESULTS AND DISCUSSION

In this section results of applying metrics in Sect. 5.4 to different RANSAC enhancements are presented and analyzed as given in Fig. 3. Algorithms are applied to 7 scenes from datasets mentioned in Sect. 5.1. The following subsections present the results obtained:

6.1 Runtime Speed

Figure 4 shows the results obtained by applying the test for each point cloud. Note that the Runtime Speed results varies among the chosen point clouds because of the different semantic features of each scene. Overall, FRANSAC has the highest average of processing frame per seconds with 39.5 fps, which is almost 7 times faster than standard RANSAC. In contrast, the lowest performance of only 4 fps are occupied by RRANSAC. RRANSAC's behavior is random and cannot be expected and good hypotheses may be neglected. Voxel RANSAC comes secondly behind FAN-RANSAC with average of 9.35 fps.

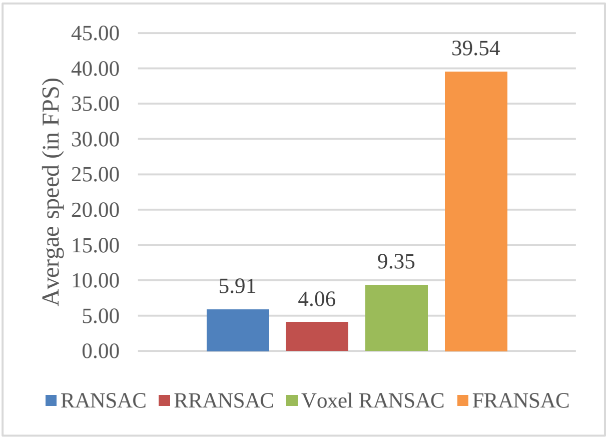

Fig 4: Runtime speed results.

6.2 Precision

Table 4 illustrates the results for the Precision as described in Sect. 5.4. Precision takes all retrieved points into account, which can be used for random errors description or a measure of statistical dispersion.

By far the most common trend is that FRANSAC, standard RANSAC and Voxel RANSAC obtains the best proportions of about 100%. On the other hand, RRANSAC has the lowest percentage of 90% because of its unexpected random behavior. This approaches succeeded in most scenes such as 3 and 4, but it returns unreliable results on scene 7.

Table 4. Precision results

Scene	RANSAC	RRANSAC	Voxel RANSAC	FRANSAC
1	99.91	96.24	99.58	99.70
2	100.00	87.21	100.00	100.00
3	100.00	99.16	100.00	99.98
4	100.00	98.62	100.00	99.99
5	100.00	97.09	99.98	99.99
6	99.81	93.32	99.99	99.99
7	100.00	58.00	99.98	99.90
Average	99.96	89.95	99.93	99.94

6.3 Recall

Results obtained for the Recall by applying Eq. 4 are given in Table 5. Recall is called sensitivity, which can be explained as the ability to detect relevant points from the retrieved input data. The most striking feature is that FRANSAC returns the lowest Recall percentages with average of 77.5%. In contrast, RANSAC ranked first giving about 90% on average. It is trivial to achieve a Recall of 100% by returning all input data points as shown in scenes 1, 3 and 4 by RRANSAC.

Table 5. Recall results

Scene	RANSAC	RRANSAC	Voxel RANSAC	FRANSAC
1	97.48	100.00	97.66	99.07
2	86.20	60.14	74.65	70.20
3	92.55	100.00	82.38	73.89
4	94.16	100.00	86.37	83.97
5	90.10	88.37	73.49	66.63
6	84.89	98.30	75.39	72.44
7	83.99	56.08	76.09	76.40
Average	89.91	86.13	80.86	77.51

6.4 F1-score

Table 6 gives an overview of the results obtained for the F-Score measurements. F-Score considers both Precision (P) and Recall (R) of the test to compute the score as illustrated in Eq. 5 and it can be used to indicate the test's accuracy. It can be seen that RANSAC ranks first giving back an average measurement of nearly 94.5%. Moreover, FRANSAC, RRANSAC and Voxel RNASAC are very similar with averages of 87%, 87.5% and 89% respectively.

7. CONCLUSION

In this work a simple, yet very fast approach to segment range images and 3D point clouds. Firstly, the 3D point cloud is obtained from the sensor then a pre-processing stage is applied to the point cloud where pass-through filter and voxelization are used before plane segmentation is performed using enhanced 3D RANSAC.

The performance and quality are assessed using the public Technische Universität München dataset which contains a large number of scenes with gradual increase in complexity. Experimental evaluation has shown that our approach excels in runtime speed result with about 40 fps in average. In addition, our approach does not considerably rank behind state-of-the-art depth image segmentation techniques in case of quality metrics.

Table 6. F-score results

Scene	RANSAC	RRANSAC	Voxel RANSAC	FRANSAC
1	98.68	98.08	98.61	99.39
2	92.59	71.19	85.49	82.49
3	96.13	99.58	90.34	84.98
4	96.99	99.30	92.69	91.28
5	94.79	92.53	84.72	79.97
6	91.75	95.74	85.97	84.02
7	91.30	57.02	86.41	86.58
Average	94.60	87.64	89.17	86.96

8. REFERENCES

[1] Y. M. Kim, N. J. Mitra, D.-M. Yan, and L. Guibas, "Acquiring 3D indoor environments with variability and repetition," ACM Trans. Graph., vol. 31, no. 6, p. 1, 2012.

[2] C. V. Nguyen, S. Izadi, and D. Lovell, "Modeling kinect sensor noise for improved 3D reconstruction and tracking," Proc. - 2nd Jt. 3DIM/3DPVT Conf. 3D Imaging, Model. Process. Vis. Transm. 3DIMPVT 2012, pp. 524–530, 2012.

[3] M. Niessner, M. Zollhöfer, S. Izadi, and M. Stamminger, "Real-time 3D Reconstruction at Scale Using Voxel Hashing," ACM Trans. Graph., vol. 32, no. 6, p. 169:1--169:11, 2013.

[4] M. Zollh et al., "Real-time Non-rigid Reconstruction using an RGB-D Camera," 2013.

[5] A. Dai, M. Nießner, M. Zollhöfer, S. Izadi, and C. THEOBALT, "BundleFusion: Real-time Globally Consistent 3D Reconstruction using On-the-fly Surface Re-integration," arXiv.org, 2016.

[6] M. Innmann, M. Zollhöfer, M. Nießner, C. Theobalt, and M. Stamminger, "VolumeDeform: Real-time Volumetric Non-rigid Reconstruction," pp. 1–17, 2016.

[7] L. Alexandre, "3D Object Recognition using Convolutional Neural Networks with Transfer Learning between Input Channels," 13th Int. Conf. Intell. Auton. Syst., 2014.

[8] S. Aigerim, A. Askhat, and A. Yedilkhan, "Recognition of 3D object using Kinect," Appl. Inf. Commun. Technol. (AICT), 9th Int. Conf., pp. 341–346, 2015.

[9] G. Pang and U. Neumann, "Fast and Robust Multi-view 3D Object Recognition in Point Clouds," 3D Vis. (3DV), Int. Conf., pp. 171–179, 2015.

[10] O. Hilliges, Kim, S. Izadi, M. Weiss, and A. Wilson, "Holodesk: Direct 3D interactions with a situated see-through display," Proc. CHI 2012, pp. 2421–2430, 2012.

[11] A. Wilson, H. Benko, S. Izadi, and O. Hilliges, “Steerable augmented reality with the beamatron,” UIST ’12 Proc. 25th Annu. ACM Symp. User interface Softw. Technol., pp. 413–422, 2012.

[12] R. Du, “Video Fields : Fusing Multiple Surveillance Videos into a Dynamic Virtual Environment,” pp. 1–8, 2016.

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

[14] P. J. Huber, Robust Statistics, vol. 82, no. 3. 1981.

[15] C. V. Stewart, “Robust Parameter Estimation in Computer Vision,” SIAM Rev., vol. 41, no. 3, pp. 513–537, 1999.

[16] J. Matas and O. Chum, “Randomized RANSAC with Td,d test,” Image Vis. Comput., vol. 22, no. 10 SPEC. ISS., pp. 837–842, 2004.

[17] P. Gyawali and J. McGough, “Simulation of detecting and climbing a ladder for a humanoid robot,” IEEE Int. Conf. Electro Inf. Technol., 2013.

[18] J. Pardeiro, J. V. Gómez, D. Álvarez, and L. Moreno, “Learning-based floor segmentation and reconstruction,” Adv. Intell. Syst. Comput., vol. 253, pp. 307–320, 2014.

[19] A. Hidalgo-Paniagua, M. A. Vega-Rodríguez, N. Pavón, and J. Ferruz, “A Comparative Study of Parallel RANSAC Implementations in 3D Space,” Int. J. Parallel Program., vol. 43, no. 5, pp. 703–720, 2014.

[20] L. Dung, C. Huang, and Y. Wu, “Implementation of RANSAC Algorithm for Feature-Based Image Registration,” J. Comput. Commun., vol. 2013, no. November, pp. 46–50, 2013.

[21] J. W. Tang, N. Shaikh-Husin, and U. U. Sheikh, “FPGA implementation of RANSAC algorithm for real-time image geometry estimation,” Proceeding - 2013 IEEE Student Conf. Res. Dev. SCOReD 2013, no. December, pp. 290–294, 2015.

[22] J. Vourvoulakis, J. Lygouras, and J. Kalomiros, “Acceleration of RANSAC algorithm for images with affine transformation,” 2016 IEEE Int. Conf. Imaging Syst. Tech., pp. 60–65, 2016.

[23] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, “A benchmark for the evaluation of RGB-D SLAM systems,” in IEEE International Conference on Intelligent Robots and Systems, 2012, pp. 573–580.