

EXPRESS PAPER

Open Access



Incremental structural modeling on sparse visual SLAM

Rafael A. Roberto^{1*}, Hideaki Uchiyama², João Paulo S. M. Lima^{1,3}, Hajime Nagahara², Rin-ichiro Taniguchi² and Veronica Teichrieb¹

Abstract

This paper presents an incremental structural modeling approach that improves the precision and the stability of existing batch-based ones for sparse and noisy point clouds from visual simultaneous localization and mapping (SLAM). The main idea is to use the generating process of point clouds on SLAM effectively. First, a batch-based method is applied to point clouds that are incrementally generated from SLAM. Then, the temporal history of reconstructed geometric primitives is statistically analyzed to suppress incorrect reconstruction. The evaluation shows that both precision and stability are improved compared to an existing batch-based method, and the proposed method is suitable for real-time structural modeling.

Keywords: Incremental structural modeling, Geometric modeling, Sparse visual SLAM

1 Introduction

The generation of 3D shape has become common thanks to the popularization of low-cost depth sensors. Such sensors acquire the shape described by a dense point cloud that is converted into a continuous surface defined by meshes. Then, the surface is converted into geometric primitives by estimating the types of the shapes and their parameters. This structural modeling is useful in various applications such as 3D scene understanding and reverse engineering. Also, it saves plenty of memory because a dense point cloud is basically redundant for scene description, and geometric primitives can be described by fewer parameters.

Most of the existing methods on structural modeling estimate geometric primitives from a dense point cloud acquired with LiDAR scanners at large outdoor scales [1] and with Kinect-like sensors at indoor environments [2]. Their research issue is how to estimate model parameters of particular primitives [3]. For example, there exist methods to determinate the parameters of 3D planes [4] and cylinders [5, 6]. Different types of primitives can be simultaneously detected using iterative model fitting [7, 8]. Hough transform-based approaches are also proposed to be robust to a noisy point cloud [9]. However,

they are highly time-consuming due to the high density of the cloud. Also, the input data is usually a dense point cloud, and the performance with a sparse point cloud is not investigated.

In the existing methods, random sample consensus (RANSAC)-based methods can apply to sparse point clouds because they estimate primitives by initially picking a minimal group of points for each shape and detecting the one that approximates the maximum number of points [8]. Besides, they can also work with data containing a large number of outliers [3]. Therefore, the performance of a RANSAC-based method [8] for sparse point clouds is first investigated, and its drawbacks are analyzed in Section 2. Note that there exist methods that deal with sparse point clouds, but they can detect planes only such that textured planes are reconstructed [10].

Another aspect of existing methods is that they basically work in a batch and an off-line manner. It means that an input data is analyzed all together only once. In recent years, the performance of on-line and incremental 3D reconstruction such as visual simultaneous localization and mapping (SLAM) has drastically been improved regarding both the accuracy of the reconstruction and the computational cost for real-time applications [11, 12].

*Correspondence: rar3@cin.ufpe.br

¹Voxar Labs, Informatics Center, Federal University of Pernambuco, Recife, Brazil
Full list of author information is available at the end of the article

Compared with batch acquisition of a point cloud, an incremental approach can provide not only a point cloud but also the generating process of the cloud, which, as far as the authors know, has never been incorporated into structural modeling in the literature.

This paper presents an incremental structural modeling that estimates geometric primitives on sparse keypoint-based visual SLAM. The key idea is to use not only a sparse point cloud but also the generating process of the cloud on SLAM. First, an existing batch-based structural modeling is applied to a sparse point cloud that are incrementally updated on SLAM. Then, the types of shapes and their parameters in the cloud are statistically determined from the temporal history of estimated geometric primitives. Since the estimation of geometric primitives on sparse point clouds is sensitive to noisy reconstruction, both precision and stability can be improved by using the generating process. Also, the proposed method is suitable for real-time structural modeling, as discussed in the evaluation.

2 Evaluation of efficient RANSAC

The performance of RANSAC-based structural modeling with sparse point clouds is first evaluated. Efficient RANSAC [8]¹ was selected as a benchmark method because it was able to detect different types of geometric primitives under noisy data including outliers. For the evaluation, a test scene was created as illustrated in Fig. 1a. Sparse point clouds were generated using a monocular SLAM system [12], as shown in Fig. 1b. The reconstruction of the clouds was calibrated using a chessboard pattern so that the clouds were represented in metric scale.

Since Efficient RANSAC has five heuristic parameters, different combinations of these parameters were tested to find the best ones for sparse point clouds. It was searched for the parameters that maximized the number of assigned points, which were the ones from the input point cloud that were modified to be fitted to a shape. These parameters also minimized the distance between these points and the ones on the input point cloud. Figure 1c is the result of the best set of parameters, which assigned 90.38% of the points with an average distance of 3.77 centimeters for each assigned point. From this result, it was found that geometric primitives were able to be detected even from sparse point clouds if best parameters were selected. However, the instability of Efficient RANSAC was also unveiled.

Figure 1d shows the shapes detected using the same data as the previous test with different initial values for RANSAC. In the result, 89.75% of the points were assigned with an average distance of 2.01 cm. Compared to the previous test, six of them were differently detected, and five of them were wrong. This unstable result often

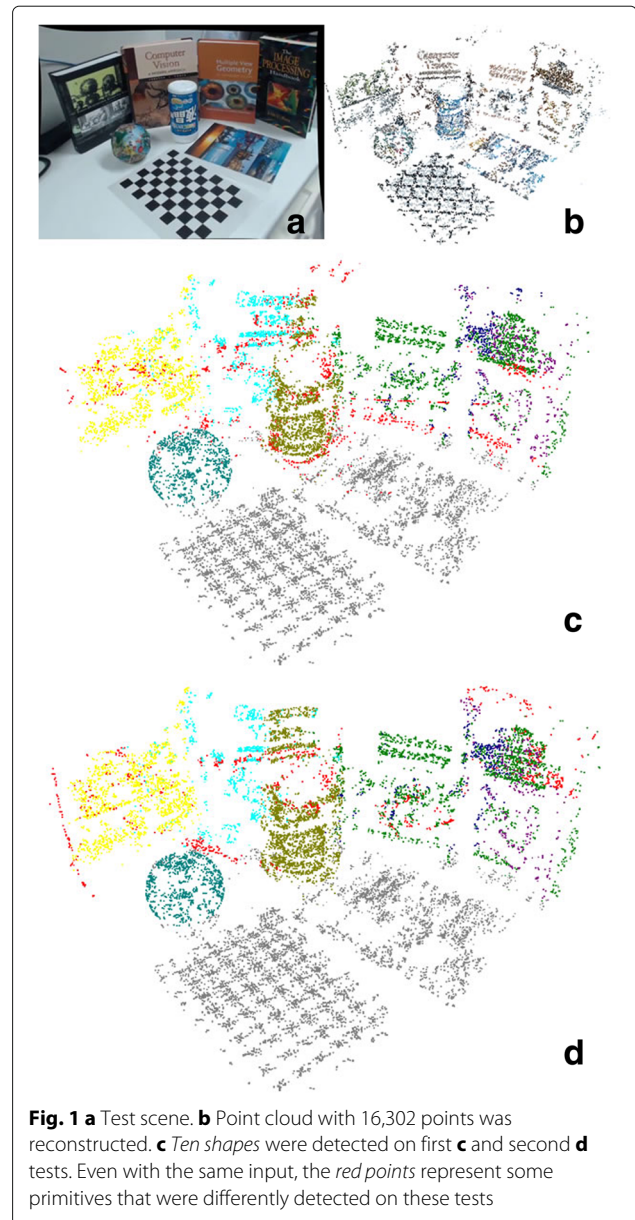


Fig. 1 a Test scene. b Point cloud with 16,302 points was reconstructed. c Ten shapes were detected on first c and second d tests. Even with the same input, the red points represent some primitives that were differently detected on these tests

occurred with noisy and sparse point clouds from visual SLAM. To stably estimate geometric primitives from such clouds, a method to incorporate the generating process of the clouds into structural modeling is proposed in the next section.

3 Incremental structural modeling

Existing structural modeling worked with all the input point cloud available before primitive estimation. In visual SLAM, point clouds are incrementally built, and their generating process is also available. From this process, the temporal history of the primitives detected at each update of the point clouds can be computed. In order to improve both reliability and stability of structural modeling, the

history of estimated primitives is statistically analyzed. In other words, the proposed method evaluates the history of estimated primitives to eliminate unreliable results, replace incorrect primitives with correct ones and restore shapes that were not detected at a particular moment.

3.1 Fusion of shapes

One characteristic of sparse point clouds from keypoint-based visual SLAM is that keypoints are clustered at highly textured areas. For example, in Fig. 1a, two different patterns over the table were observed and formed two distinct clusters of points as illustrated in Fig. 1b. In this case, efficient RANSAC detected them as two separated planes even though they belong to the same plane on the table and should be detected as one. In other cases, one object can be detected as several different objects due to non-uniform distribution of points in the cloud.

In order to improve the results from efficient RANSAC, different shapes that belong to the same primitive are first fused. It not only provides a more representative primitive but also increases the overall information regarding the shape. Since a primitive with a small number of points can be unreliable, it is better to discard it. However, if more than one shape with similar parameters are detected, even if they are small, it is better to assume that they correspond to the same element in the scene. The reason is that it is unlikely that two primitives are wrongly detected with the same parameters. Therefore, the fusion of primitives based on parameters helps in the history evaluation. The fusion process is simply based on the geometric information of the primitives as follows:

- Plane: the planes are parallel given an angle threshold α_t , and the distance between them is smaller than the distance threshold d_t ;
- Sphere: the distance between their centers is lower than d_t , and the difference between their radii is less than the radius threshold r_t ;
- Cylinder: the angle between the axis direction of both cylinder is smaller than α_t , the distance between them is less than d_t , and the difference between their radii is smaller than r_t .

While d_t and r_t are controlled for each case such that they are 2% of the largest size of the point cloud bounding box, α_t is always set to 5°.

3.2 Temporal update of shapes

In keyframe-based visual SLAM, the map is updated at each keyframe, which adds new shapes in the scene in an incremental manner. Figure 2 is the flow of the proposed method. As explained in Section 3.1, shape fusion fuses the different shapes that belong to the same primitive to subsequently evaluate the history of shapes. Then, both

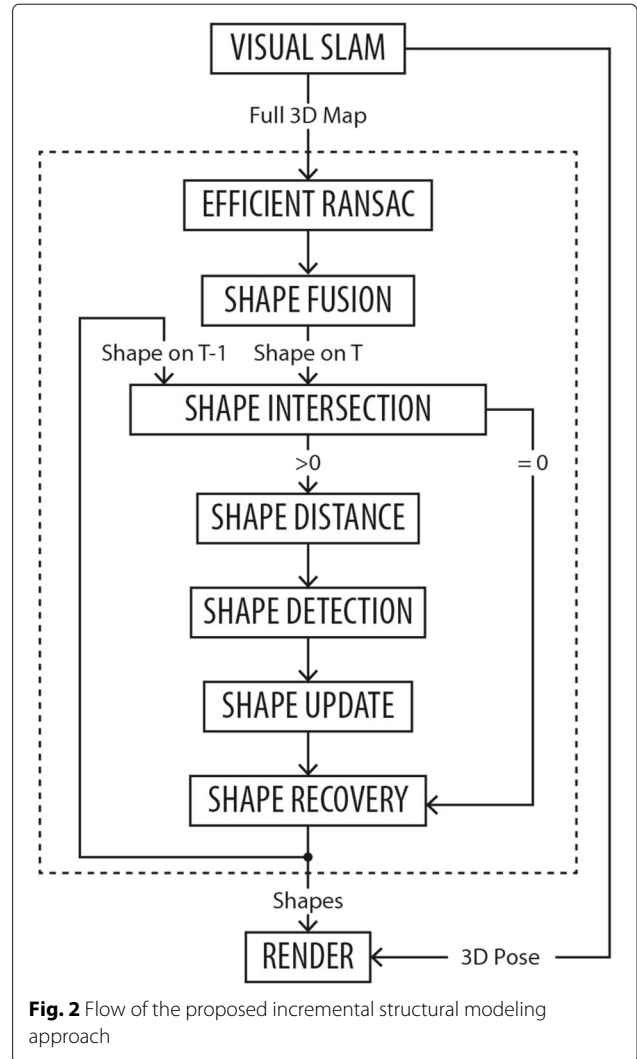


Fig. 2 Flow of the proposed incremental structural modeling approach

the shapes detected on the current keyframe using efficient RANSAC [8] and the ones found previously are used to stably estimate geometric primitives. The following steps are as follows:

Shape intersection: the intersection ratio between the bounding boxes of every shape on the current keyframe and those on the previous one is computed. Shapes that appear for the first time have no intersection and they are registered for future evaluation.

Shape distance: for primitives on the current keyframe that intersect with other ones on the previous keyframe, the distance between their centers of mass is computed. Shapes with high intersection ratio and small distance in two consecutive moments have more chance to be correspondents, assuming that the scene is stationary.

Shape detection: due to noise and inconsistency on the shape detection of such point cloud, it occurs that a primitive on the current keyframe matches with several others on the previous one, including different shape types.

Therefore, it is necessary to detect the class of the shape on previous keyframe that is the most likely to be the correspondent of the current one. It is computed as s_c value for each class of primitive, and the one with maximum value is selected as correspondence:

$$s_c = \frac{\sum_i^{ns_c} |\psi_i| p_i}{\sum_i^{ns_c} |\psi_i| d_i} \frac{|\psi_s|}{\sum_i^{ns_c} |\psi_i|}, \quad (1)$$

where ns_c are the indices of the shapes of the given class with intersection on previous keyframes, $|\psi_i|$ is the number of points of that shape, p_i and d_i are the intersection ratio and distance between centers of mass, respectively, and $|\psi_s|$ is the number of points in the primitive on the current keyframe.

Shape update: each shape keeps the history of primitive classes since its first appearance. The current shape inherits this record from the one detected as its correspondent, and the shape type is updated to determine the class of primitive detected on the current keyframe. Then, this historical data is evaluated to verify if the current type is compatible with the whole history. The shape type can change if it is different from the dominant class, which is the one that appeared on more than 50% of the time.

Shape recovery: it is verified if there are shapes that were not detected on current keyframe but were on the previous one. These shapes are brought back to the current set of primitives with the same parameters from last appearance. However, the historic of the shape will register the number of keyframes in which it was recovered instead of detected. If this shape is not detected anymore, it will disappear when it is recovered more than detected. Finally,

only the stable estimated shapes will be displayed by the Render.

4 Evaluation

The proposed approach was implemented in C++ and integrated with a SLAM system [12]. In the evaluation, the proposed method named incremental structural modeling (ISM) was compared with regular structural modeling (RSM), which is efficient RANSAC [8]. As illustrated in Fig. 3, datasets with five different scenarios were created since there is no public dataset. The number of keyframes computed from [12] for each image sequence and target objects are as follows: *Case1* has 31 keyframes and different objects for each class of shape, being the most complex case. *Case2* and *Case3* have 24 and 20 keyframes, respectively, and they focus more on cylinders. *Case4* and *Case5* have 7 and 17 keyframes, respectively, and include planes only. Regarding the quantitative evaluation, it was executed twice for each case, and the average result of each keyframe was used to compute precision, recall, and F_1 -score.

As described in Table 1, the F_1 -score was improved in all the cases. The result was the same for precision. In fact, no shape was wrongly detected in *Case2* and *Case3* by ISM. One primitive was incorrectly estimated by ISM for only two keyframes in one of the tests for both *Case1* and *Case5*. As compared, considering the 92 keyframes on *Case1*, *Case2*, *Case3*, and *Case5*, for the two round of tests, both approaches detected shapes 184 times. In 153 of them, the efficient RANSAC approach detected at least one shape wrong. Note that results for *Case4* were much

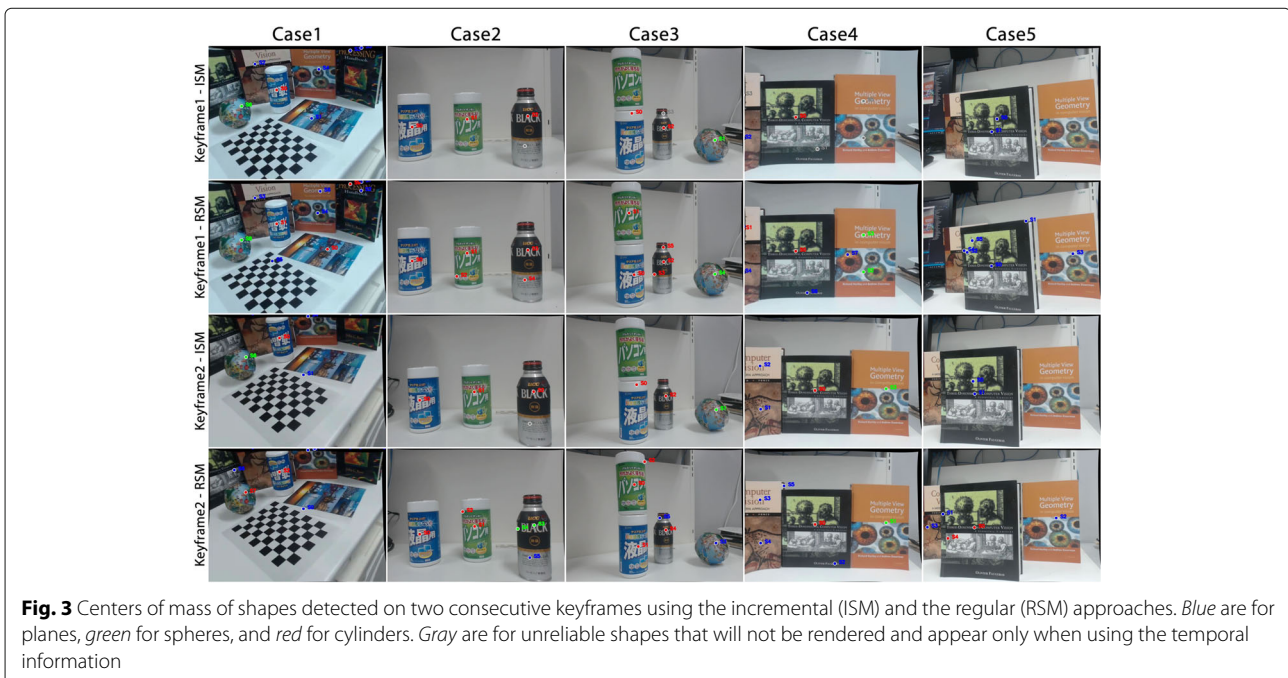


Fig. 3 Centers of mass of shapes detected on two consecutive keyframes using the incremental (ISM) and the regular (RSM) approaches. *Blue* are for planes, *green* for spheres, and *red* for cylinders. *Gray* are for unreliable shapes that will not be rendered and appear only when using the temporal information

Table 1 The comparison of precision, recall, and F₁-score between the proposed method (ISM) and efficient RANSAC (RSM)

		Precision	Recall	F ₁ -Score
Case1	RSM	0.822	0.819	0.820
	ISM	0.993	0.832	0.906
Case2	RSM	0.756	0.756	0.756
	ISM	1.000	0.963	0.981
Case3	RSM	0.795	0.795	0.795
	ISM	1.000	0.900	0.947
Case4	RSM	0.500	0.489	0.494
	ISM	0.647	0.440	0.524
Case5	RSM	0.761	0.750	0.755
	ISM	0.942	0.645	0.766

lower than the others mainly due to the quality of the tracking and the mapping of visual SLAM.

In two cases, RSM achieved a better recall because it outputs twice more shapes on average. However, some of them were incorrect and the precision became lower. In ISM, such shapes were filtered out because they were not reliable and the precision became higher.

Due to these improvements, ISM achieved more precise and more stable detection. Figure 3 (rows 2 and 4) shows some cases in which the same object was detected as distinct shapes at different moments by RSM. This fact is rarely noticed when using ISM, which usually shows the correct primitive regardless if the one detected does not display it if the shape is not reliable. Moreover, the results become more stable as the time passes because there is more information to decide the correct primitive in ISM.

As for the computational cost, the process to estimate the shape from a point cloud depends on its size. The time was collected on every test and on average takes 29.04 ± 16.21 ms per group of thousand points to retrieve the primitives on a computer with an Intel Core i5-6300U (2.40 GHz) and 8 GB of RAM. The largest and the smallest point clouds are from the last keyframe of Case1 (16,698 points) and first keyframe of Case2 (940 points). The entire process of combining different shapes and computing the temporal information takes an additional 5.77 ± 1.00 ms per group of thousand points.

5 Conclusions

It was proposed in this work a method that incrementally estimated geometric primitives from the sparse maps generated by visual SLAM. This approach used the history information from every reconstruction to select the correct type of shape and their parameters so that the result was stable even when dealing with noisy point clouds. The

evaluation showed that the use of temporal information contributed to more precise and more stable modeling and presents better results regarding precision F₁-score and, in most cases, recall.

One improvement for future versions of this method is to also use the color information available on the images on both the fusion of shapes as well as on the incremental modeling. Another future work is to develop an application that uses the incremental approach in combination with a real-time visual SLAM system.

Endnote

¹ Source code available at <https://goo.gl/XINs6N>.

Acknowledgements

The authors would like to thank Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) (processes 140898/2014-0 and 456800/2014-0) for partially funding this research.

Authors' contributions

RAR contributed by conceiving, designing, and programming the technique, acquiring, analyzing, and interpreting the data, and drafting the manuscript. HU contributed by conceiving and designing the technique, analyzing and interpreting the data, and drafting and revising the manuscript. JPSML and HN contributed by conceiving and designing the technique, analyzing and interpreting data, and revising the manuscript. RT and VT contributed by conceiving and designing the technique, revising the manuscript and coordinating the research. All authors read and approved the final manuscript.

Authors' information

Rafael A. Roberto is a Ph.D. candidate in Computer Science at the Federal University of Pernambuco and a researcher at Voxar Labs. His research interests include 3D tracking, computer vision, and augmented reality for mobile devices. Hideaki Uchiyama is an assistant professor at Kyushu University. His research interests include computer vision and augmented reality. João Paulo S. M. Lima is an assistant professor at Statistics and Informatics Department of at the Federal Rural University of Pernambuco and senior scientist at the Voxar Labs. His research interests include 3D tracking, augmented reality, computer vision, and computer graphics. Hajime Nagahara is an associate professor at Kyushu University. His research interests include computer vision and computational photography. Rin-ichiro Taniguchi is a full time professor at Kyushu University. His research interests include image processing, pattern recognition, and computer vision. Veronica Teichrieb is an associate professor at the Federal University of Pernambuco and head of the Voxar Labs research group. Her research interests include augmented reality, visualization, tracking and interaction.

Competing interests

The authors declare that they have no competing interests.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Author details

¹Voxar Labs, Informatics Center, Federal University of Pernambuco, Recife, Brazil. ²Kyushu University, Fukuoka, Japan. ³Statistics and Informatics Department, Federal Rural University of Pernambuco Recife, Brazil.

Received: 21 February 2017 Accepted: 7 March 2017

Published online: 15 March 2017

References

- Musialski P, Wonka P, Aliaga DG, Wimmer M, Gool L, Purgathofer W (2013) A survey of urban reconstruction. *Comput Graph Forum* 32(6):146–177. doi:10.1111/cgf.12077
- Kim YM, Dolson J, Sokolsky M, Koltun V, Thrun S (2012) Interactive acquisition of residential floor plans. In: 2012 IEEE International

- Conference on Robotics and Automation. IEEE Publications, St. Paul, pp 3055–3062. doi:10.1109/ICRA.2012.6224595
3. Roth G, Levine MD (1993) Extracting geometric primitives. *CVGIP: Image Underst* 58(1):1–22. doi:10.1006/ciun.1993.1028
 4. Nguyen T, Reitmayr G, Schmalstieg D (2015) Structural modeling from depth images. *IEEE Trans Vis Comput Graph* 21(11):1230–1240. doi:10.1109/TVCG.2015.2459831
 5. Liu YJ, Zhang JB, Hou JC, Ren JC, Tang WQ (2013) Cylinder detection in large-scale point cloud of pipeline plant. *IEEE Trans Vis Comput Graph* 19(10):1700–1707. doi:10.1109/TVCG.2013.74
 6. Qiu R, Zhou QY, Neumann U (2014) Pipe-run extraction and reconstruction from point clouds. In: Pajdla T, Schiele B, Tuytelaars T (eds). Fleet D. Springer International Publishing, Zurich. pp 17–30. doi:10.1007/978-3-319-10578-9_2
 7. Li Y, Wu X, Chrysathou Y, Sharf A, Cohen-Or D, Mitra NJ (2011) GlobFit: Consistently fitting primitives by discovering global relations. *ACM Trans Graph* 30(4):52–15212. doi:10.1145/2010324.1964947
 8. Schnabel R, Wahl R, Klein R (2007) Efficient RANSAC for point-cloud shape detection. *Comput Graph Forum* 26(2):214–226. doi:10.1111/j.1467-8659.2007.01016.x
 9. Drost B, Ilic S (2015) Local hough transform for 3d primitive detection. In: 2015 International Conference on 3D Vision. IEEE Publications, Lyon, pp 398–406. doi:10.1109/3DV.2015.52
 10. Sinha SN, Steedly D, Szeliski R, Agrawala M, Pollefeys M (2008) Interactive 3d architectural modeling from unordered photo collections. *ACM Trans Graph* 27(5):159–115910. doi:10.1145/1409060.1409112
 11. Mur-Artal R, Montiel JMM, Tardós JD (2015) ORB-SLAM: a versatile and accurate monocular slam system. *IEEE Trans Robot* 31(5):1147–1163. doi:10.1109/TRO.2015.2463671
 12. Uchiyama H, Taketomi T, Ikeda S, dM Lima JPS (2015) [POSTER] Abecedary tracking and mapping: a toolkit for tracking competitions. In: 2015 IEEE International Symposium on Mixed and Augmented Reality. IEEE Publications, Fukuoka, pp 198–199. doi:10.1109/ISMAR.2015.63

Submit your manuscript to a SpringerOpen[®] journal and benefit from:

- Convenient online submission
- Rigorous peer review
- Immediate publication on acceptance
- Open access: articles freely available online
- High visibility within the field
- Retaining the copyright to your article

Submit your next manuscript at ► springeropen.com
