Plane Object-based High-Level Map Representation for SLAM

Pavel Gritsenko[2], Igor Gritsenko[2], Askar Seidakhmet[2], and Bogdan Kwolek[1]

[1] AGH University of Science and Technology, 30 Mickiewicza, 30-059 Krakow, Poland
http://home.agh.edu.pl/~bkw/contact.html
[2] Al-Farabi Kazakh National University, Prospect al-Farabi 71, Almaty, Kazakhstan
lickro@mail.ru

Abstract. High-level map representation providing object-based understanding of the environment is an important component for SLAM. We present a novel algorithm to build plane object-based map representation upon point cloud that is obtained in real–time from RGB-D sensors such as Kinect. On the basis of segmented planes in point cloud we construct a graph, where a node and edge represent a plane and its real intersection with other plane, respectively. After that, we extract all trihedral angles (corners) represented by 3rd order cycles in the graph. Afterwards, we execute systematic aggregation of trihedral angles into object such as trihedral angles of the same plane-based object have common edges. Finally, we classify objects using simple subgraph patterns and determine their physical sizes. Our experiments figured out that the proposed algorithm reliably extracts objects, determines their physical sizes and classifies them with a promising performance.

1 Introduction

Recent progress in development of robotics has increased capabilities such as mobility and autonomy. Navigating in unknown environments requires a SLAM (Simultaneous Localization and Mapping) system or module. The SLAM problem can be posed as the metaphorical chicken-and-egg dilemma: a robot in order to determine its current location needs an accurate map. However, in order to incrementally build a map, it should have estimate of its position within the map. The SLAM problem is related to two questions: "where am I?" (localization), and "what does the environment look like?" (mapping). The currently utilized formulation of SLAM has its origins in the seminal work [1].

A robust SLAM algorithm is an essential component for any mobile robot to navigate through an unstructured environment. The SLAM problem has been one of the most popular research topics in mobile robotics for the last two decades and several approaches have been proposed [2]. In [3], authors argue that developments in the area of SLAM are entering the third era - "robust–perception age". The key requirements for the new age SLAM system are as follows: 1) Robust Performance; 2) High-Level Understanding; 3) Resource Awareness; 4) Task-Driven Perception.

In this paper, we propose an algorithm to build high-level map representation in order to extend capabilities of state-of-the-art SLAM algorithms to match the requirements mentioned above. At present, most of RGB-D SLAM systems relies on point clouds or truncated signed distance function (TSDF). The first crucial drawback is that these representations require substantial amount of memory. Even in case of mapping a simple environment, like an empty room, in both mentioned representations the memory requirements grow very fast with scene complexity and time, which makes impossible to use them in long-term operation and leads to noncompliance with the key requirements of third era SLAM systems – robust performance and resource awareness.

2 Relevant Work and Our Contribution

2.1 Relevant Work

Compact map requiring small amount of memory is an essential component of any SLAM system for long term operation and operation in a large environment. There are number of works that were devoted to constructing compact maps on the basis of geometric primitives like points, lines, and planes [4,5,6,7,8]. SLAM system using plane-to-plane correspondences was presented in [4], whereas the problem of unknown correspondences was investigated in [5]. One of the major shortcomings of plane-based methods [7,6,4,5] is insufficient number of non-parallel planes. This problem is partially solved in [8] by using additional laser scanner with large FOV, but with an additional cost and system complexity. In line-based SLAM it is hard to obtain line correspondences because the RGB-D data from 3D sensors like Kinect is noisy and includes missing depth values [6]. An exemplar point-based RGB-D mapping system is presented in [1]. It is based on seeking for three point-to-point correspondences using RGB and depth map stream to find an initial estimate of the pose using RANSAC, which is further handled and improved by ICP algorithm.

It is worth noting that despite remarkable progress in constructing a compact map, almost all SLAM systems include little semantic information to the map [3]. There are three main ways to construct high-level map representation and to provide semantics to a mobile robot [3]. The first kind of solutions treat the SLAM as a first step and then add semantics to produced map. The first attempt consisted in building classical geometric map using a 2D laser scanner [9]. Then associative Markov network was used to fuse the classified semantic places from each robot pose. A later work was quite similar, but it concentrated on 3D maps that were constructed off-line from RGB-D sequences [10]. The first on-line version was developed by Pronobis et al. [11]. After that, object recognition in videos has been supported by a monocular SLAM system [12]. The second group of approaches extract semantics in advance and then take advantage of prior knowledge including semantic classes or objects, their geometry, in order to improve the mapping quality. The first attempts focused on monocular SLAM with sparse features [13] and a dense map representation [14]. The first successful RGB-D SLAM was developed by Salas-Moreno et al. [15]. The third kind of

approaches combine extraction of semantics and SLAM and do it simultaneously. The first successful approach was achieved by Flint et al. [16], where a model leveraging the Manhattan world assumption has been utilized. In a later work, estimation of camera parameters and objects using both geometric and semantic information has been investigated [17]. Despite achieving improvement in the performance of object recognition, this approach is considered to be impractical for on-line robot operation because of execution time (about 20 minutes per image pair) and limited number of objects that can be processed. The complexity of the algorithm was gradually reduced in [18] using late fusion of semantic information and metric map. However, the system still works only in off-line. The first success in constructing on-line system with object recognition and adding semantic information to the scene has been achieved in [19] through the use of stereo cameras and a dense map representation.

2.2 Differences with Relevant Approaches

Our approach differs in several aspects from the relevant work. The first key difference is that our system does not decompose point cloud into set of separate primitives, but it aggregates planes into objects and builds a complex representation consisting of plane-based objects and point clouds representing objects of complex shapes, see Fig. 1. The second difference is that our algorithm recon-

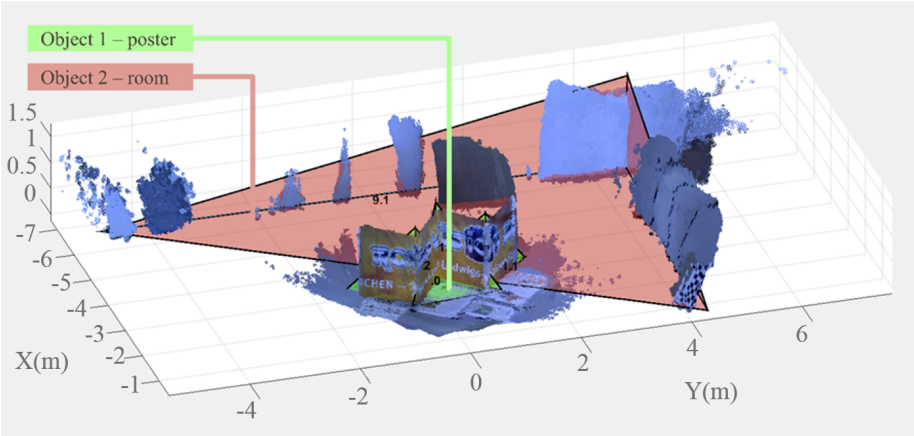

Fig. 1: Plane object-based high-level map representation on the point cloud reconstructed from 907 frames, represented by open TUM RGB-D dataset [20] (rgbd_dataset_freiburg3_structure_texture_far).

structs plane-based objects of any type consisting of any number of faces, and thus it differs from RANSAC, which extracts only objects of predefined shape. The third difference is that our high-level map representation is capable of reconstructing plane-based object of any size (room, floor, building) from sequence of frames, i.e. it adds object parts frame-by-frame. We will show that our map representation gradually reduces memory consumption by dozens of times with

respect to point cloud-based representation. The fourth difference is that our algorithm allows to use all state-of-the-art closed-form solutions [21,22,23,24,25,26] to determine the robot pose. Closed form solutions do not assume any motion model and are more reliable comparatively to the most popular state-of-the-art ICP algorithm [27]. The use of trihedral angles (corners) to represent plane-based objects stands for this ability. The trihedral angle allows to use all three types of primitives: point, line, plane, see also Fig. 1, as well as point-to-point [23], line-to-line [22], plane-to-plane [21], point-to-line [25], point-to-plane [26], and line-to-plane [24] correspondences and their closed-form solutions. From one hand any of such solutions can be used at a time. From the other hand, any combination of several solutions can be used [28]. This gradually increases the robustness of the proposed algorithm. Additionally, closed-form solutions are obtained in one iteration and require less computational resources than ICP. Therefore, the fourth difference refers to resource awareness.

It is worth noting that the fifth difference lies in classification of plane-based objects using a graph. The algorithm is based on graph patterns instead of ANN, RANSAC or other complex and computationally expensive algorithm. It is simple, fast and computationally very cheap, which makes possible real-time or close to real-time execution on ARM processors.

Another crucial point is that none of state-of-the-art representations provide high-level information about the environment to allow a robot to distinguish between objects, their sizes and types [3]. With the ability to extract objects and their physical sizes, our high-level map representation provides great amount of additional information, and provides additional opportunities for place recognition, loop closure, higher level of geometry understanding, i.e. semantics to achieve more user-friendly interaction between human and robot. In summary, this paper makes the following contributions:

- We present a novel plane object-based high-level map representation.
- We present an efficient algorithm to find real intersections (edges) between planes and to determine their sizes.
- We present an efficient algorithm to classify plane-based objects using subgraph patterns, which is fast and computationally effective.
- We demonstrate that the algorithm achieves real-time performance with mapping accuracy comparable to state-of-the-art algorithms.

3 Algorithm overview

Figure 2 shows step–by–step transition from simple point cloud to classified plane–based object representation with sizes (edge lengths). Actually the whole algorithm consists of seven major steps. In the first step, it acquires RGB-D data stream consisting of a pair of a color image and a depth map. After that, on the basis of the RGB-D stream and intrinsic calibration parameters we reconstruct colored point cloud. Such a colored point cloud becomes an initial map representation of the perceived scene. In the next step a plane segmentation from initial point cloud on the basis of M-estimator SAmple Consensus (MSAC) is executed.

Then, real planes intersections as well as their lengths are determined. On the basis of the segmented planes and their intersections a graph is constructed. This results in the second map representation, which consists of plane equations and point clouds representing objects of complex shape (not plane-based objects). Due to the substitution of point clouds representing plane equations the map representation becomes compact. The next step comprises extraction of trihedral angles that are represented by 3rd order cycles from the constructed graph (Fig. 3). After extracting the trihedral angles, their aggregation can be achieved assuming that trihedral angles of the same plane-based object have common edges. Finding common edges for trihedral angles is done using the constructed graph, which is fast and computationally effective. Through aggregating all trihedral angles having common edges we obtain abstract plane-based objects. This means that in this step we have a structure describing which planes belong to which object, the number of such objects, theirs sizes, but no types. Therefore, we call this representation as abstract plane-based objects, see Fig. 2. In the last step the classification is executed. The classification algorithm is based on matching graph patterns with a subgraph from the extracted graph. The resulting algorithm is simple, fast and effective.

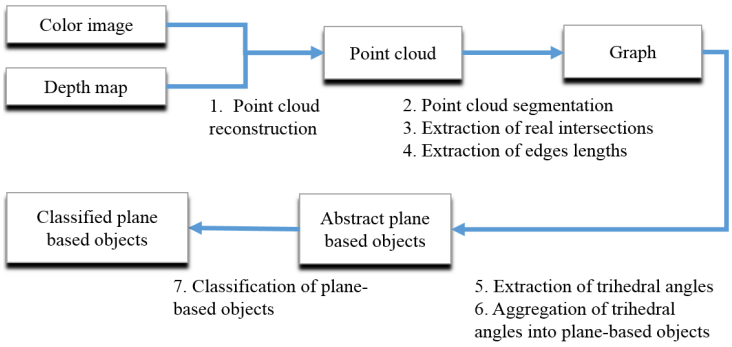

Fig. 2: Diagram representing transition from colored point cloud to classified plane object-based high-level map representation.

4 Proposed Algorithm

4.1 Determining physical object size

Let's consider how to extract real intersection of two planes and its length. Let's assume that we have planes: $P_1 = A_1x + B_1y + C_1z + D_1$ and $P_2 = A_2x + B_2y + C_2z + D_2$. According to the algorithm presented in Fig. 2, after point cloud segmentation we determine the plane intersections. Therefore, after this step we have planes and their inliers, i.e. array of points representing them. Let's denote P_1 inliers as $\{p_{plane1,1} \dots p_{plane1,i}\}$ and P_2 inliers as $\{p_{plane2,1} \dots p_{plane2,i}\}$. According to the definition of intersection that states that if planes P_1 and P_2 have real intersection then they should have common

inliers $\{p_{edge,1} \dots p_{edge,i}\}$, i.e. array of points belonging to P_1 and P_2 simultaneously, see (1):

$$\{p_{edge,1} \dots p_{edge,i}\} = \{p_{plane1,1} \dots p_{plane1,i}\} \cap \{p_{plane2,1} \dots p_{plane2,i}\} \quad (1)$$

However, $p_{edge,1} \dots p_{edge,i} = \emptyset$, because a point can be among inliers of only single plane. Thus, we reformulate definition and state that the points representing real intersection $p_{edge,1} \dots p_{edge,i}$ will be simultaneously close to planes P_1 and P_2. Therefore, we calculate point-to-plane distance for both sets of inliers to both planes on the basis of (2)–(5).

$$\{d_{plane1-plane1,1} \dots d_{plane1-plane1,i}\} = |A_1 p_{plane1,i,x} + B_1 p_{plane1,i,y} + C_1 p_{plane1,i,z} + D_1| * (A_1^2 + B_1^2 + C_1^2)^{1/2} \quad (2)$$

$$\{d_{plane1-plane2,1} \dots d_{plane1-plane2,i}\} = |A_2 p_{plane1,i,x} + B_2 p_{plane1,i,y} + C_2 p_{plane1,i,z} + D_2| * (A_2^2 + B_2^2 + C_2^2)^{1/2} \quad (3)$$

$$\{d_{plane2-plane2,1} \dots d_{plane2-plane2,i}\} = |A_2 p_{plane2,i,x} + B_2 p_{plane2,i,y} + C_2 p_{plane2,i,z} + D_2| * (A_2^2 + B_2^2 + C_2^2)^{1/2} \quad (4)$$

$$\{d_{plane2-plane1,1} \dots d_{plane2-plane1,i}\} = |A_1 p_{plane2,i,x} + B_1 p_{plane2,i,y} + C_1 p_{plane2,i,z} + D_1| * (A_1^2 + B_1^2 + C_1^2)^{1/2} \quad (5)$$

After that we determine all points that have the distance to both planes smaller than d_{thresh} and grater than d_{min}. d_{min} is used to deal with noise.

In order to define edge length d_{edge} we took two points from $p_{edge,1} \dots p_{edge,i}$ with maximum distance to each other. Actually, the distance between the points represents diagonal length $d_{diagonal}$ of the cylinder of radius d_{thresh}. Thus, the edge length is determined according to (6):

$$4 * d_{tresh}^2 + d_{edge}^2 = d_{diagonal}^2 \quad (6)$$

4.2 Aggregation of planes into object

After determining the plane intersections we can construct a graph whose node and edge represent a plane and its real intersection with other plane, respectively. This graph allows us to find in a fast manner all trihedral angles as 3rd order cycles in the graph, see Fig. 3. Afterwards, we make assumption that all trihedral angles of the same plane-based object have common edges.

This assumption allow us to extract plane-based objects. Trihedral angle is represented by three planes, see Fig. 3, so let's denote it by indexes of planes. For example, trihedral angle 2-3-7 has common faces with 2-9-7, 2-6-3, 3-7-5. Therefore, they belong to the same plane-based object. In the same time, the trihedral angle 2-9-7 has common edge with 2-9-6, and 5-9-7 so that all these (2-3-7, 2-6-3, 2-9-7, 3-7-5, 2-9-6, 5-9-7) trihedral angles belong to the same plane-based object. This way the aggregation is capable of adding trihedral angles, angle-by-angle.

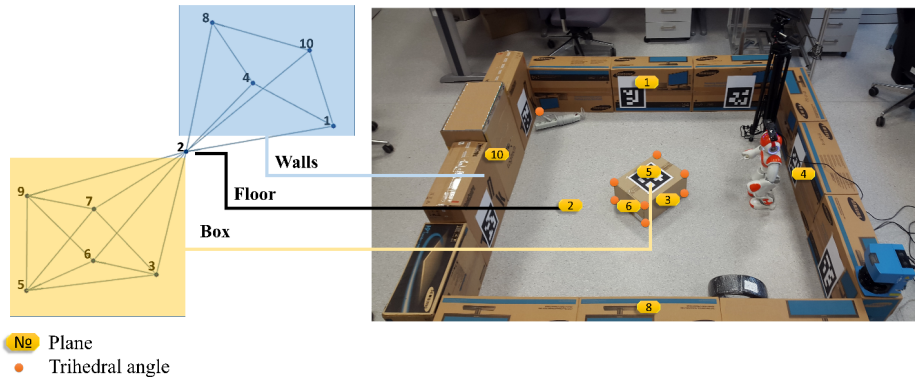

Fig. 3: The graph representing scene of the sequence 2 – a box in a simple room with four walls, which are observed by Nao humanoid robot.

4.3 Classification of objects

After all plane-based objects are extracted, a classification step can be executed. It is based on splitting the graph into sub-graphs (for instance plane two in Fig. 3), and then matching the predefined patterns, see Fig. 4.

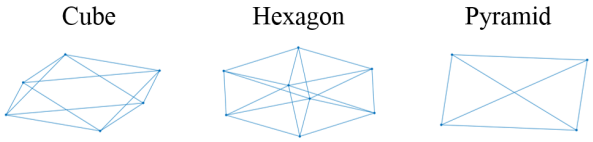

Fig. 4: Predefined patterns to match objects in the graph representation.

4.4 Merging plane object-based high-level map representation

Merging a pair of plane object-based high-level map representation is done using a graph representation. At first, data association is done and plane-to-plane correspondence is found on the basis of matching normals as well as distances between the considered planes. After that, a transformation matrix between frames is computed using closed-form solution for plane-to-plane correspondence [29]. The transformation matrix is further refined by a ICP algorithm. After that, the frames are aligned. Then using the plane-to-plane correspondence we merge graph representations. We assume that each edge length is the longest one from corresponding edges in pair of frames. This way the graph is reconstructed. Finally, we reinitialize the plane-based object structure and the classification, see Fig. 5.

5 Experimental Results

Our system is conceived to work with any RGB-D sensor. In the experiments we utilized Kinect 1 and ASUS Xtion sensors. They work at a frame rate of 30 Hz

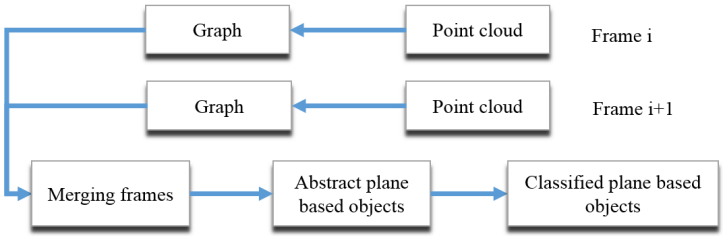

Fig. 5: Algorithm of merging plane object-based high-level map representations.

and provide color stream and depth map at a resolution of 640×480 pixels. The system has been tested on two sequences. The first sequence is available in open collection of Kinect (RGB+D) datasets with 6D ground truth recorded by the TUM CV group [20], (rgbd_dataset_freiburg3_structure_texture_far). It includes 907 frames with synchronized color image, depth map, accelerometer data and ground-truth for each step. The dataset has been recorded with frame–rate of 30 Hz and sensor resolution of 640×480 pixels. The ground-truth trajectory was obtained using high-accuracy motion-capture system consisting of eight high-speed tracking cameras (100 Hz). Figure 6 depicts results that were achieved in constructing plane object-based high-level map representation using hand-held Kinect. Figure 6a shows results of reconstruction of point cloud using color image, depth map and intrinsic calibration data that are provided with the dataset. Figure 6b depicts extraction of planes from the reconstructed point cloud using MSAC. Figure 6c shows results of systematic aggregation of trihedral angles into objects.

We merged point clouds that were reconstructed step-by-step from color images and depth map, using ground–truth trajectory. We determined the merged point clouds for frames number 200, 400 and 907.

As can be seen in Fig. 6a, there is substantial level of noise in the point cloud. Despite this the segmentation step is successful, see Fig. 6b. As can be seen in Fig 6b–c, the proposed algorithm for extraction of real intersections of planes from point cloud generated satisfactory results. Its is worth noting that the poster stands on the floor and its planes have real intersections with it. Despite that the floor has also intersections with the room planes, the system correctly extracted two different plane-based objects (poster and room) that are represented on Fig. 6c by different colors (green and red, respectively). However, we have no ability to estimate edge length error because there is no data in the TUM dataset for object sizes. We have determined object sizes for our dataset.

The second RGB-D sequence[1] has been recorded using the ASUS Xtion Pro Live that has been mounted on the Nao humanoid robot. In the discussed scenario the Nao robot made a circle of the radius 0.7 m in 24 steps. In each step, RGB, depth and ground-truth data were recorded, see Fig. 8. Ground-truth position has been determined using 2D LIDAR (SICK LMS 200 laser scanner), see Fig. 3. This dataset has been utilized in an experiment comprising constructing

[1] Available at: `http://bit.ly/ICCVG2018`

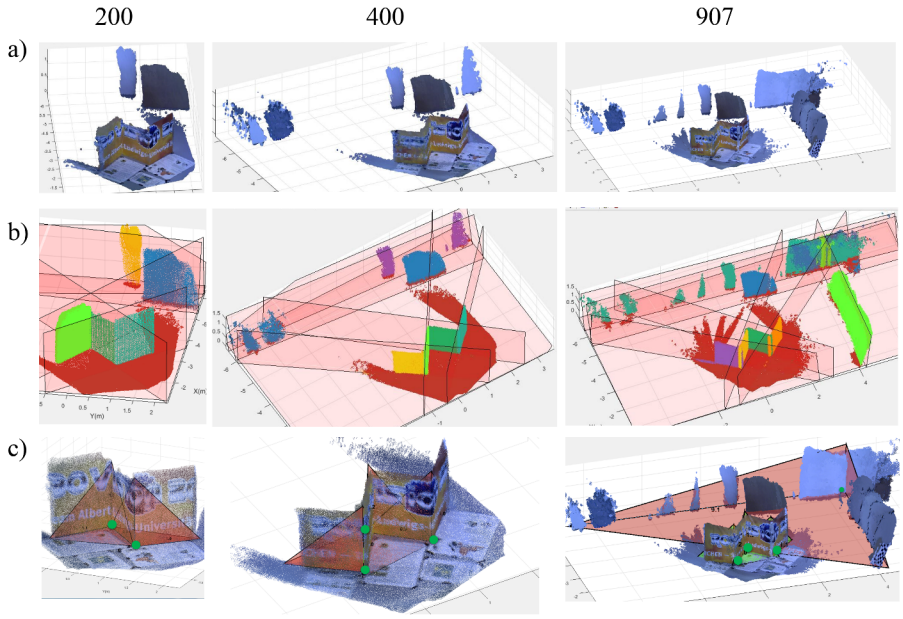

Fig. 6: An example of constructing plane object-based high-level map representation using hand-held Kinect. (a) Reconstructed point cloud using color image, depth map and intrinsic calibration data of the Kinect from TUM RGB-D dataset. (b) Results of extracting planes from reconstructed point cloud using MSAC. (c) Results of systematic aggregation of trihedral angles into objects. Green points of the graph stand for centers of trihedral angles. Edges of trihedral angles are represented by black line segments.

high-level map representations for each frame and merging them frame-by-frame. Figure 7 compares point cloud that was reconstructed frame-by-frame using state-of-the-art ICP algorithm with plane object-based high-level map representation that was reconstructed frame-by-frame on the basis of direct-estimate of plane-to-plane correspondence [29], see Fig. 7b.

It is worth noting that trihedral angle allows us to use all three types of basic geometric primitives: point, line and plane (Fig. 1). However, it is an open problem how to combine transformation matrices that are obtained on the basis of different closed form solutions for our high-level map representation. Actually, in the discussed experiment we have employed plane-to-plane [21] correspondence to find the transformation matrix between frames. Our system consumes about 60 times less amount of memory than state-of-the-art point cloud, see high level-map representation on Fig. 8a. The discussed figure depicts also the estimated object sizes as well as the estimated path with respect to ground-truth path.

Comparing estimated object sizes for box and walls (see Fig. 8) to its ground truth values (box height = 0.19 m, box width = 0.29 m, box length=0.38 m, room width=room length=1.88 m) we obtained the average error of 0.02 m.

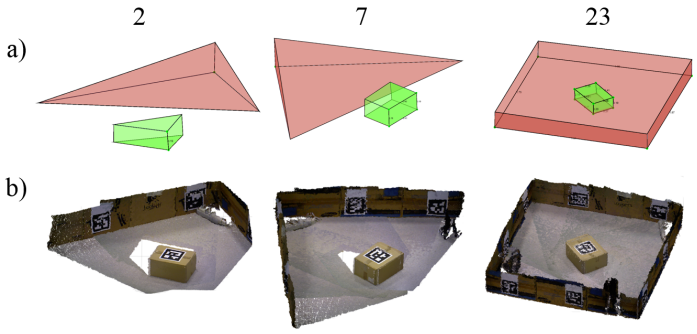

Fig. 7: Comparison of merging plane object-based high-level map representation using 6 DoF plane-to-plane correspondence (b) with merging point clouds reconstructed from color images and depth maps on each step using state-of-the-art ICP algorithm (a).

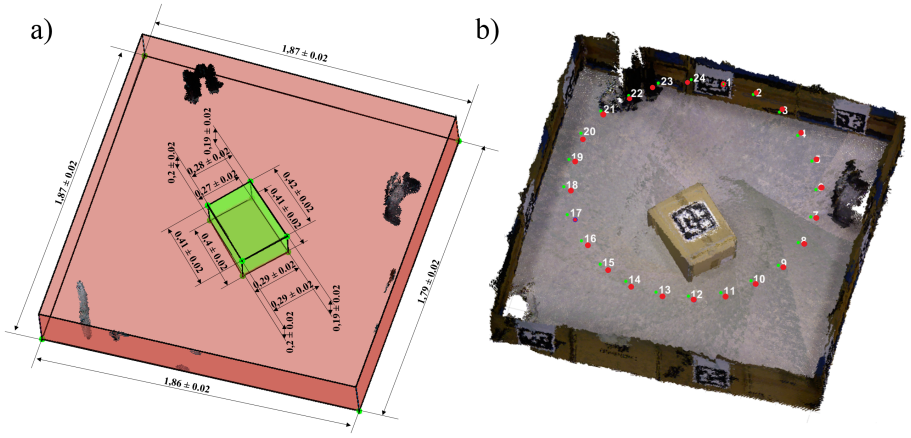

Fig. 8: Comparison of memory consumption of plane object-based high-level map representation for SLAM with point cloud. High-level map representation with estimated object sizes and errors (a). Ground truth trajectory (green points) in (b) has been determined using LMS 200 SICK laser scanner while the path (red points) has been estimated on the basis of visual odometry and our high level map representation.

6 Conclusions

We presented a novel algorithm for constructing plane object-based high-level map representation for SLAM. It achieves promising performance on point clouds acquired by Kinect and ASUS Xtion sensors and provides reliable extraction of plane-based objects, their dimensions and types. In future work we plan to extend

the system and use all combinations of primitives for calculating 6 DoF robot pose.

Acknowledgment. This work was partially supported by Polish National Science Center (NCN) under a research grant 2014/15/B/ST6/02808 as well as by PhD program of the Ministry of Science and Education of the Republic of Kazakhstan.

References

1. Henry, P., Krainin, M., Herbst, E., Ren, X., Fox, D.: RGB-D mapping: Using depth cameras for dense 3D modeling of indoor environments. In: Int. Symp. Experimental Robotics (ISER). (2010) 477–491
2. Bresson, G., Alsayed, Z., Yu, L., Glaser, S.: Simultaneous localization and mapping: A survey of current trends in autonomous driving. IEEE Trans. on Intelligent Vehicles **2**(3) (2017) 194–220
3. Cadena, C., Carlene, L., Carrillo, H., Latif, Y., Scaramuzza, D., Neira, J., Reid, I., Leonard, J.: Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. IEEE Trans. on Robotics **32**(6) (2016) 1309–1332
4. Weingarten, J., Siegwart, R.: 3D SLAM using planar segments. In: IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS). (October 2006) 3062–3067
5. Pathak, K., Birk, A., Vaskevicius, N., Poppinga, J.: Fast registration based on noisy planes with unknown correspondences for 3-D mapping. IEEE Trans. Robotics **26**(3) (June 2010) 424–441
6. Taguchi, Y., Jian, Y., Ramalingam, S., Feng, C.: Point-plane SLAM for hand-held 3D sensors. In: IEEE Int. Conf. on Robotics and Automation (ICRA). (May 2013) 5182–5189
7. Bodis-Szomoru, A., Riemenschneider, H., Van-Gool, L.: Efficient edge-aware surface mesh reconstruction for urban scenes. Comput. Vis. Image Understanding **66** (2015) 91–106
8. Trevor, A., Rogers, J., Christensen, H.: Planar surface SLAM with 3D and 2D sensors. In: IEEE Int. Conf. Robotics Automation (ICRA). (May 2012) 3041–3048
9. Mozos, M., Triebel, R., Jensfelt, P., Axel, R., Burgard, W.: Supervised semantic labeling of places using information extracted from sensor data. Robot. Auton. Syst. **55**(5) (2007) 391–402
10. Lai, K., Bo, L., Fox, D. In: Unsupervised feature learning for 3D scene labeling. IEEE (9 2014) 3050–3057
11. Pronobis, A., Jensfelt, P.: Large-scale semantic mapping and reasoning with heterogeneous modalities. In: IEEE Int. Conf. on Robotics and Automation. (May 2012) 3515–3522
12. Pillai, S., Leonard, J.: Monocular SLAM supported object recognition. In: Robot., Sci. Syst. Conf. (2015) 310–319
13. Civera, J., Galvez-Lopez, D., Riazuelo, L., Tardos, J., Montiel, J.: Towards semantic SLAM using a monocular camera. In: Intelligent Robots and Systems (IROS). (2011) 1277–1284
14. Dame, A., Prisacariu, V., Ren, C., Reid, I.: Dense reconstruction using 3D object shape priors. In: IEEE Conf. Comput. Vis. Pattern Recognit. (2013) 1288–1295

15. Salas-Moreno, R., Newcombe, R., Strasdat, H., Kelly, P., Davison, A.: SLAM+: Simultaneous localisation and mapping at the level of objects. In: IEEE Conf. Comput. Vis. Pattern Recognit. (2013) 1352–1359
16. Flint, A., Murray, D., Reid, I.: Manhattan scene understanding using monocular, stereo, and 3D features. In: Int. Conf. on Computer Vision. (2011) 2228–2235
17. Bao, S., Bagra, M., Chao, Y., Savarese, S.: Semantic structure from motion with points, regions, and objects. In: IEEE Conf. Comput. Vis. Pattern Recognit. (2012) 2703–2710
18. Kundu, A., Li, Y., Dellaert, F., Li, F., Rehg, J.: Joint semantic segmentation and 3D reconstruction from monocular video. In: Computer Vision – ECCV 2014, Springer Int. Publ. (2014) 703–718
19. Vineet, V., Miksik, O., Lidegaard, M., Niessner, M., Golodetz, S., Prisacariu, V., Kaehler, O., Murray, D., Izadi, S., Perez, P., Torr, P.: Incremental dense semantic stereo fusion for large-scale semantic scene reconstruction. In: IEEE Int. Conf. Robot. Autom. (2015) 75–82
20. Sturm, J., Engelhard, N., Endres, F., Burgard, W., Cremers, D.: A benchmark for the evaluation of RGB-D SLAM systems. In: Proc. of the Int. Conf. on Intelligent Robot Systems (IROS). (2012) 573–580
21. Grimson, W., Lozano-Perez, T.: Model-based recognition and localization from sparse range or tactile data. Int. J. of Robotics Research **3**(3) (1984) 3–35
22. Zhang, Z., Faugeras, O.: Determining motion from 3D line segment matches: A comparative study. Image and Vision Computing **9**(1) (1991) 10 – 19
23. Umeyama, S.: Least-squares estimation of transformation parameters between two point patterns. IEEE Trans. Pattern Anal. Mach. Intell. **13**(4) (1991) 376–380
24. Chen, H.: Pose determination from line-to-plane correspondences: Existence condition and closed-form solutions. IEEE Trans. on Pattern Analysis and Machine Intelligence **13**(6) (June 1991) 530–541
25. Nister, D.: A minimal solution to the generalised 3-point pose problem. In: IEEE Conf. Computer Vision and Pattern Recognition (CVPR). (June 2004) 560–567
26. Ramalingam, S., Taguchi, Y.: A theory of minimal 3D point to 3D plane registration and its generalization. Int. J. of Computer Vision **102** (2012) 73–90
27. Besl, P., McKay, N.: A method for registration of 3-D shapes. IEEE Trans. on Pattern Analysis and Machine Intelligence **14**(2) (Feb 1992) 239–256
28. Walker, M., Shao, L., Volz, R.: Estimating 3-D location parameters using dual number quaternions. CVGIP: Image Understanding **54**(3) (1991) 358–367
29. Khoshelham, K.: Direct 6-DoF pose estimation from point-plane correspondences. In: Int. Conf. on Digital Image Computing: Techniques and Applications (DICTA). (2015) 1–6