2D Image Feature-Based Real-Time RGB-D 3D SLAM

Donghwa Lee[1], Hyongjin Kim[1], and Hyun Myung[1,2]

[1] Department of Civil and Environmental Engineering, KAIST,
291 Daehak-ro, Yuseong-gu, Daejeon 305-701,
Republic of Korea
[2] Robotics Program, KAIST,
291 Daehak-ro, Yuseong-gu, Daejeon 305-701,
Republic of Korea
{leedonghwa,hjkim86,hmyung}@kaist.ac.kr

Abstract. This paper proposes a real-time RGB-D (red-green-blue depth) 3D SLAM (simultaneous localization and mapping) system. Kinect style sensors give RGB-D data which contains 2D image and per-pixel depth information. 6-DOF (degree-of-freedom) visual odometry is obtained through the 3D-RANSAC (three-dimensional random sample consensus) algorithm with image features and depth information. For speed up extraction of features, parallel computation is performed on a GPU (graphics processing unit) processor. After a feature manager detects loop closure, a graph-based SLAM algorithm optimizes trajectory of the sensor and 3D map. Experimental results show the processing rate over 20 Hz.

Keywords: SLAM, 3D SLAM, RGB-D camera, image features, 3D-RANSAC.

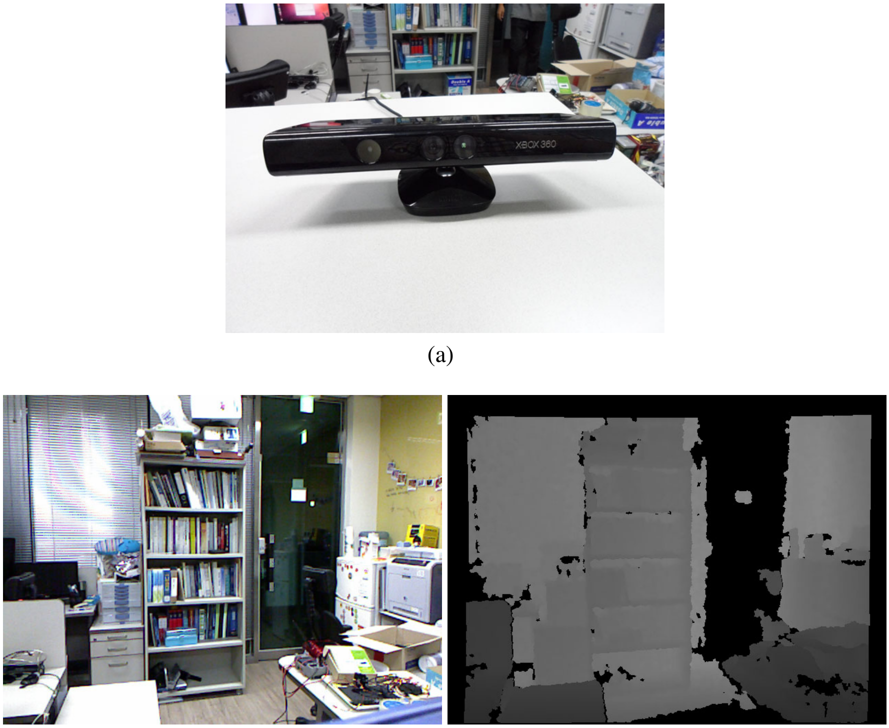
1 Introduction

There have been many researches for the SLAM (Simultaneous Localization and Mapping) problem over the past decade. The initial studies focused on two-dimensional environments, hence they was usually applied to mobile robots [1, 5, 10]. Recently, a variety of 3D SLAM algorithms supports 6-DOF (degree-of-freedom) pose optimization, therefore the SLAM technique is employed in various platforms like quadrotors, underwater robots, etc [7–9].

In the early 3D SLAM studies, expensive sensors like 2D and 3D-LRFs (laser range finders) were mainly used. But with the advent of cheap sensors like the Microsoft Kinect sensor, rapid development of the 3D SLAM area has begun [2–4, 6]. The Kinect sensor contains a depth sensor and a color camera (figure 1). The depth sensor obtains depth data using the IR (infrared) projection method [12]. Figures 1(b) and (c) show a color image and depth data from the Kinect sensor. The Kinect style sensors are called the RGB-D (red-green-blue depth) camera since they give the color image and the depth data concurrently.

Recently, the robotics and computer vision communities have focused on 3D SLAM techniques using the RGB-D camera data. Peter Henry *et al.* [6] used a FAST (Features from Accelerated Segment Test) and the ICP (Iterative Closest Point) method for visual odometry estimation. TORO (Tree-based netwORk Optimizer) SLAM algorithm

J.-H. Kim et al. (Eds.): *Robot Intelligence Technol. & Appl. 2012*, AISC 208, pp. 485–492.
DOI: 10.1007/978-3-642-37374-9_47 © Springer-Verlag Berlin Heidelberg 2013



(a)

(b) (c)

Fig. 1. RGB-D sensor system. (a) Microsoft Kinect sensor. (b) RGB color image. (c) Per-pixel depth data.

optimizes the full trajectory and 3D model. Loop closure detection also makes use of the FAST features. But this work did not operate in real-time. Microsoft Research presented KinectFusion [11] which maps 3D model at 30 Hz using the Kinect sensor and a GPU (Graphics Processing Unit) processor. The GPU processors are specialized in parallel computing, hence they processed the depth data for aligning and mapping 3D model in real-time. But this work has weakness in drift noise, since they did not use the loop closure detection and SLAM techniques. Felix Endres *et al.* [6] implemented and evaluated 3D SLAM with a variety of feature descriptors, the ICP algorithm, and the g2o (General framework for Graph Optimization) SLAM framework. On average, this work has the processing speed of 3 Hz.

In this paper, we propose RGB-D 3D SLAM system which has the processing rate over 20 Hz. The image feature detection is performed on the GPU processor. Visual odometry estimation uses the 3D-RANSAC (RANdom SAmple Consensus) algorithm with image features and the depth data. A feature manager detects loop closure, and then the iSAM (Incremental Smoothing And Mapping) graph-based SLAM algorithm optimizes the full trajectory. iSAM is a high-speed online SLAM core algorithm based on sparse linear algebra [7].

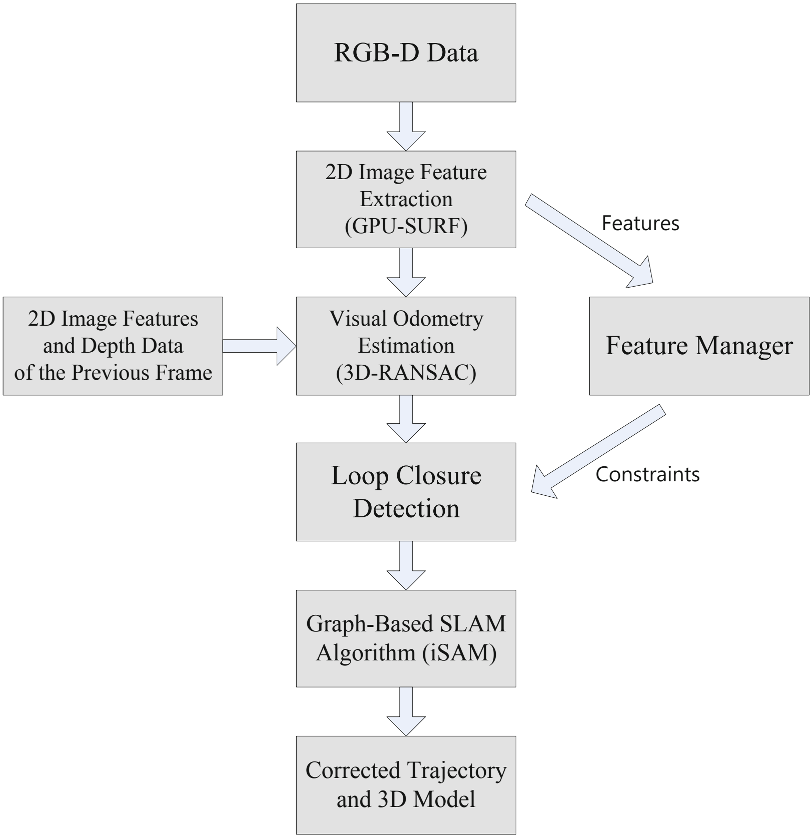


Fig. 2. Overview of the proposed RGB-D 3D SLAM system

The remainder of this paper is organized as follows. The second section presents the proposed 3D SLAM system. The third section provides experimantal results. Finally, the last section offers concluding remarks.

2 Proposed 3D SLAM System

Our approach utilizes only 2D RGB image and depth data from a RGB-D sensor. Processing steps of our system are illustrated in figure 2. First of all, 2D image features are extracted. Each feature can be located at a point in three-dimensional coordinate space with depth information. The features are used for 6-DOF visual odometry estimation with feature matching and 3D-RANSAC algorithm. Second, a feature manager gathers the whole features from the previous frames. Through comparison between the current and the preceding features, the current frame is matched to a past trajectory of the sensor. This matching procedure is called loop closure detection. Next, the full trajectory of the sensor is formed by a constraint graph with the visual odometry estimation and

Fig. 3. 2D image feature extraction and matching on a GPU processor. (Left) Previous image frame. (Right) Current image frame.

loop closure detection. After optimizing the constraint graph by the online graph-based SLAM algorithm, the corrected trajectory and the 3D map can be obtained. The whole steps are performed in real-time. Detailed explanation of this system is given in the next subsection.

2.1 Feature Extraction, Matching and 3D-RANSAC

For 6-DOF pose estimation, we extract 2D image features from the incoming color image of the sensor and match to the feature of the previous frame as shown in figure 3. We use SURF algorithm, which is less robust than SIFT algorithm, but its computational speed is faster. Although SURF has speed advantage, it is still hard to implement in real-time on CPU. Recently, GPU-based parallel computing has been applied to the feature extraction algorithms. The GPU-based algorithm allows real-time computational performance. In this system, the feature extraction procedure can handle all of the image data from the sensor in real-time (30Hz image frequency) with GPU-SURF algorithm in OpenCV 2.4.0.

In image feature procedure, feature matching algorithm has heavy computational load. The GPU computing has been also applied to the feature matching algorithm. We used GPU-based brute-force algorithm in OpenCV 2.4.0 for finding the correspondence of the features.

Using the depth information, each feature point has its position in three-dimensional coordinate space. After feature matching between the current and the previous frames, 3D-RANSAC algorithm estimates 6-DOF pose with correspondence and 3D position of the features. 3D-RANSAC algorithm in Point Cloud Library 1.5.1 was used in our implementation.

2.2 Feature Manager and Loop Closure Detection

For visual odometry estimation, it is necessary to keep the features of the previous image frame. But the constraints between the current frame and the frames of the past

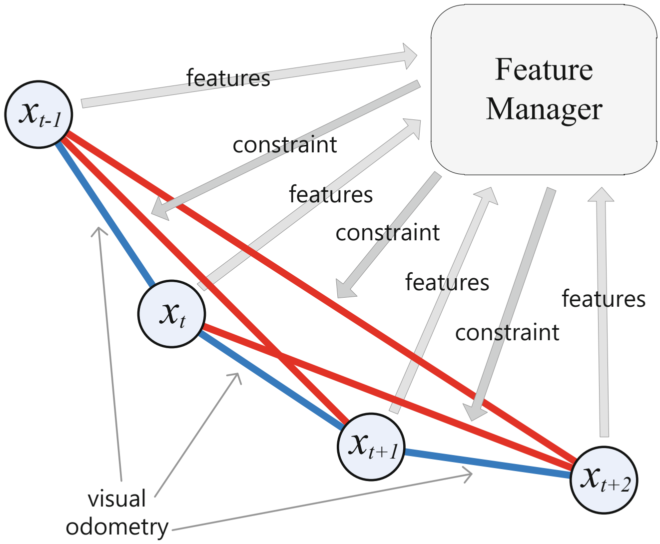


Fig. 4. Feature manager and loop closure detection

trajectory is necessary to optimize the full trajectory by graph-based SLAM algorithms. So, we designed a feature managing part named feature manager as shown in figure 4. After estimating visual odometry, the features of the current frame is sent to the feature manager. The feature manager checks duplication using the matching algorithm between the incoming features and the existing features gathered from the past frames. The features which have no correspondence to the past features are added to the feature manager as new features. Through the duplication check, related frames of the past trajectory are found, which is called loop closure detection. The features of the current frame are matched to the features from the related frames. And a 6-DOF pose constraint is obtained from the 3D-RANSAC algorithm.

2.3 Graph-Based SLAM Algorithm

In a graph-based SLAM algorithm, a graph form consisting of nodes and edges is required. The nodes represent the trajectory of a sensor and positions of landmarks. But, in the pose graph SLAM, the trajectory of a sensor is only included to the nodes. And an edge denotes a constraint between two nodes.

In our system, we construct the nodes using only the trajectory of the sensor. The edge information between the current and the previous frames is obtained by the 6-DOF visual odometry estimation. The other edges are determined with the loop closure detection procedure.

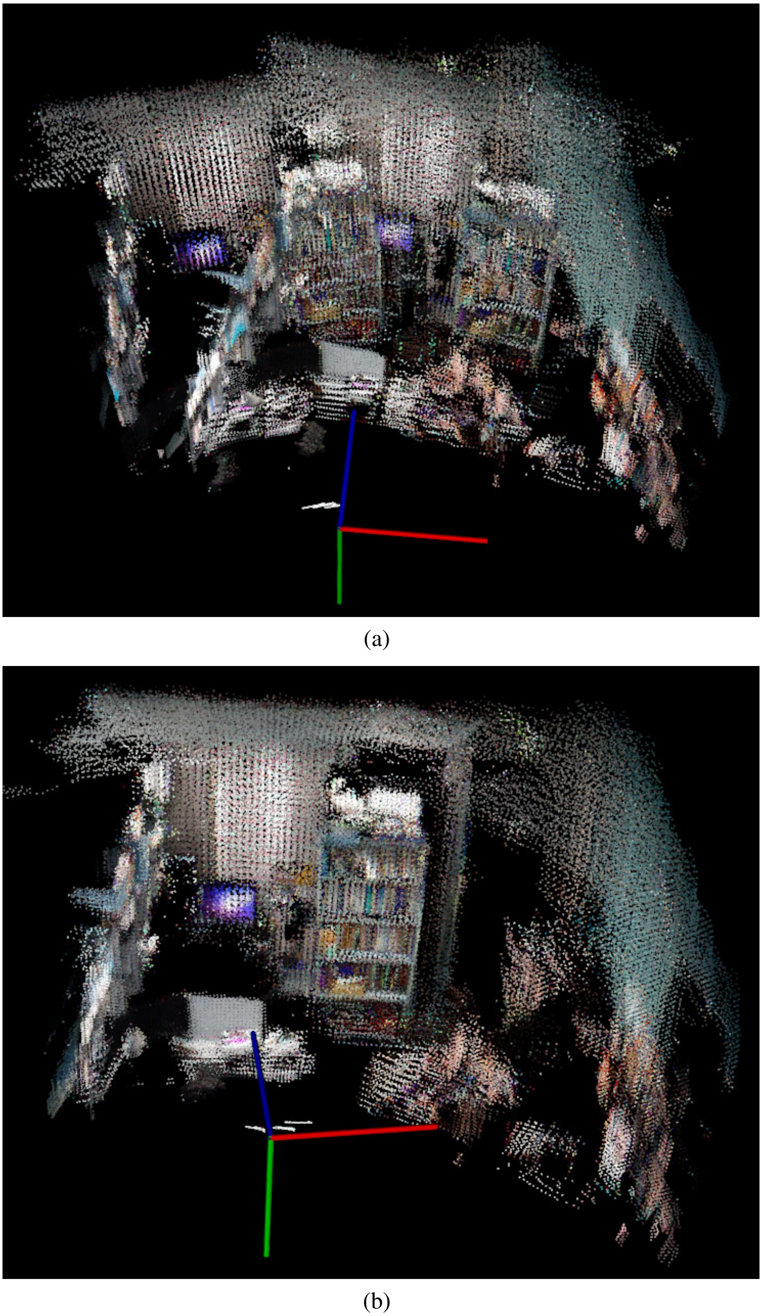


Fig. 5. Experimental results. (a) 3D model by only visual odometry. (b) Optimized 3D model by graph-based 3D SLAM algorithm.

Table 1. Average processing time of the system components

System components	Runtime (ms)
2D feature extraction (GPU-SURF)	17.2
Odometry estimation (Feature matching and 3D RANSAC)	5.9
Loop closure detection	8.1
SLAM algorithm (iSAM)	5.4
Total	36.6

Recently, a variety of graph-based SLAM algorithms has been introduced. In this paper, we use iSAM algorithm to optimize the sensor trajectory for real-time implementation. iSAM solves graph-based SLAM problem using sparse linear algebra and graphical models so that computational time is reduced drastically.

3 Experiments

We have conducted experiments in a room-size environment with the Microsoft Kinect RGB-D sensor. The Kinect uses a structured light for depth information, and its valid range is about 0.5m to 5m. The sensor gives a 2D RGB color image and depth data at 30 frames per second, both with 640×480 resolution. The 3D SLAM system is implemented on a Intel Core i7 CPU with 8 GB of memory. For accelerating the computation of the feature extraction with GPU, an Nvidia GT 560 Ti graphic card supporting the CUDA language is used.

Experimental results of the proposed 3D SLAM algorithm are presented in figure 5. Every node of the pose graph has 3D point cloud data which is transformed by the 6-DOF pose of each node and drawn in 3D space. Figure 5(a) shows 3D reconstruction results with only visual odometry data. The sensor trajectory is drifted by odometry estimation noise, so the result shows a misaligned 3D model. In figure 5(b), the nodes are optimized by the iSAM algorithm, hence the 3D model is aligned correctly.

Table 1 shows the average processing time of the proposed system in the experiments. The feature extraction part takes most of the time, 17.2 milliseconds, on average. The total processing time per frame is 36.6 milliseconds, therefore the rate of the proposed system is above 20 Hz.

4 Conclusion

This paper proposed a real-time RGB-D 3D SLAM system using only an RGB-D sensor. The visual odometry is obtained from the image features, the depth data and the 3D-RANSAC algorithm. The feature manager detects loop closure, and then the graph-based SLAM algorithm optimizes the full trajectory and the 3D model. The GPU processor accelerates operation speed of the system, and the average processing rate on a desktop PC is above 20 Hz.

The depth data is used only for the 3D position of the image features. We think that various applications of the depth data are possible while maintaining the speed of the system operation. Also, we will evaluate the trajectory and the 3D model with ground truth data in the future.

Acknowledgment. This work was supported partly by the R&D program of the Korea Ministry of Knowledge and Economy (MKE) and the Korea Evaluation Institute of Industrial Technology (KEIT). (The Development of Low-cost Autonomous Navigation Systems for a Robot Vehicle in Urban Environment, 10035354) This work was also partially funded by Samsung Electronics, Co. Ltd. This work was also financially supported partly by Korea Ministry of Land, Transport and Maritime Affairs (MLTM) as U-City Master and Doctor Course Grant Program.

References

1. Dissanayake, M., Newman, P., Clark, S., Durrant-Whyte, H., Csorba, M.: A solution to the simultaneous localization and map building (SLAM) problem. IEEE Transactions on Robotics and Automation 17(3), 229–241 (2001)
2. Endres, F., Burgard, W., Cremers, D.: A Benchmark for the Evaluation of RGB-D SLAM Systems. In: Proc. of 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS (August 2012)
3. Endres, F., Hess, J., Engelhard, N., Sturm, J., Cremers, D., Burgard, W.: An evaluation of the RGB-D SLAM system. In: Proc. of 2012 IEEE International Conference on Robotics and Automation (ICRA), pp. 1691–1696 (2012)
4. Engelharda, N., Endresa, F., Hessa, J., Sturmb, J., Burgarda, W.: Real-time 3D visual SLAM with a hand-held RGB-D camera. In: Proc. of the RGB-D Workshop on 3D Perception in Robotics at the European Robotics Forum, Vasteras, Sweden (2011)
5. Hahnel, D., Burgard, W., Fox, D., Thrun, S.: An efficient FastSLAM algorithm for generating maps of large-scale cyclic environments from raw laser range measurements. In: Proc. of 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), vol. 1, pp. 206–211 (2003)
6. Henry, P., Krainin, M., Herbst, E., Ren, X., Fox, D.: RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments. The International Journal of Robotics Research 31(5), 647–663 (2012)
7. Kaess, M., Ranganathan, A., Dellaert, F.: iSAM: Incremental smoothing and mapping. IEEE Transactions on Robotics 24(6), 1365–1378 (2008)
8. Krainin, M., Maturana, D., Fox, D.: Visual odometry and mapping for autonomous flight using an rgb-d camera. In: Proc. of The 15th International Symposium on Robotics Research, ISRR (2011)
9. Kummerle, R., Grisetti, G.: g2o: A General Framework for Graph Optimization. In: Proc. of 2011 IEEE International Conference on Robotics and Automation, ICRA (2011)
10. Michael, M., Sebastian, T., Daphne, K.: FastSLAM 2.0: An Improved Particle Filtering Algorithm for Simultaneous Localization and Mapping that Provably Converges. In: Proc. of The 18th International Joint Conference on Artificial Intelligence (2003)
11. Newcombe, R., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A., Kohli, P., Shotton, J., Hodges, S., Fitzgibbon, A.: KinectFusion: Real-time dense surface mapping and tracking. In: Proc. of 2011 10th IEEE International Symposium on Mixed and Augmented Reality (ISMAR), pp. 127–136 (2011)
12. Zhang, Z.: Microsoft Kinect Sensor and Its Effect. IEEE Multimedia 19(2), 4–10 (2012)