creative commons
COMMONS DEED

저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:

저작자표시. 귀하는 원저작자를 표시하여야 합니다.

비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.

변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 이용허락규약(Legal Code)을 이해하기 쉽게 요약한 것입니다.

Disclaimer

dCollection

Ph. D. DISSERTATION

A Forward-Viewing Mono-Camera Based SLAM System for Indoor Service Robots

실내 서비스로봇을 위한 전방 단안카메라 기반 SLAM 시스템

2017 년 8 월

서울대학교 대학원

전기 · 컴퓨터 공학부

이 태 재

Abstract

This dissertation presents a new forward-viewing monocular vision-based simultaneous localization and mapping (SLAM) method. The method is developed to be applicable in real-time on a low-cost embedded system for indoor service robots. The developed system utilizes a cost-effective mono-camera as a primary sensor and robot wheel encoders as well as a gyroscope as supplementary sensors. The proposed method is robust in various challenging indoor environments which contain low-textured areas, moving people, or changing environments. In this work, vanishing point (VP) and line features are utilized as landmarks for SLAM. The orientation of a robot is directly estimated using the direction of the VP. Then the estimation models for the robot position and the line landmark are derived as simple linear equations. Using these models, the camera poses and landmark positions are efficiently corrected by a novel local map correction method. To achieve high accuracy in a long-term exploration, a probabilistic loop detection procedure and a pose correction procedure are performed when the robot revisits the previously mapped areas. The performance of the proposed method is demonstrated under various challenging environments using dataset-based experiments using a desktop computer and real-time experiments using a low-cost embedded system. The experimental environments include a real home-like setting and a dedicated Vicon

motion-tracking systems equipped space. These conditions contain low-textured areas, moving people, or changing environments. The proposed method is also tested using the RAWSEEDS benchmark dataset.

Keywords : SLAM, Monocular vision, Vanishing Point, Line feature, Indoor Service Robot, Embedded system.

Student number : 2011-20914

Contents

List of Figures

List of Tables

1. Introduction

1.1 Background and Motivation

One of the goals in robotics is to develop a mobile robot that can act autonomously in the real world. A reliable localization is the most important prerequisite for this goal. The localization can be defined as estimating the position and orientation of the robot. The global positioning system (GPS) is the most widely used method for localization in outdoor environment. However, the popular robot localization technique in a GPS signal-denied environment, such as an indoor environment, is simultaneous localization and mapping (SLAM), in which embedded sensors are used on the robot. In SLAM, the sensors on the robot interact with the environment, and simultaneously estimate the robot's pose and surrounding environment. The SLAM provides an attractive solution because it does not need user-built maps or ad-hoc localization infrastructures. However, it is an inherently complex problem since an error of the robot pose leads to an error of the map and vice versa. Therefore, the SLAM problem has been one of the challenging issues in robotics community over the past decades.

Among various sensors for SLAM, this dissertation focuses on mono-camera-based method. A mono-camera is an attractive sensor because of its low cost, light weight, and low power consumption. In addition, it can

obtain richer visual representation from the environment. Especially for resource-limited robot platforms, using a mono-camera can be a successful solution. Regarding the viewing-direction of the camera, upward-viewing methods have several advantages in indoor environments. Visual features can be easily matched because there is no need to consider the scale invariance property of features. Also, the feature matching between adjacent images can be performed regardless of the rotation of the robot. On the other hand, forward-viewing methods have technical difficulties of feature tracking due to severe scale and viewpoint changes in the image domain. However, forward-viewing methods have a significant advantage in that the same camera can be used for the user interface, home monitoring, and obstacle avoidance [1], [2]. For example, the information obtained from the forward-viewing camera can be used for user recognition and for instructions. It can also be used to recognize obstacles in map building. The camera can also be used for surveillance and detection of home intrusions, fire or emergencies. Therefore, it is desirable to develop a forward-viewing camera-based SLAM algorithm despite the technical difficulties, which is the main topic of this dissertation.

In the literature, a large variety of solutions to the SLAM problem are available. During the last two decades, the SLAM has been intensively researched from understanding principles and problem formulation to developing core open-source libraries (The main surveys and tutorial papers for SLAM are summarized in Table 1.1). However, considering many

possible applications and environment in the real world, several major challenges still remain open [19], [20]. The application of SLAM in robotics can be classified into outdoor and indoor applications. Outdoor applications include vehicles, drones, military robots, field robots, and underwater robots. According to [21], indoor applications can be further divided into three categories: robots for factories and warehouses, robots for commercial spaces, and robots for individual homes (Fig. 1.1). Factories and warehouses are extremely controlled environments where automation engineers modified the environment for robots to work efficiently. The opposite end of this controlled environment is home. The home is not the space where robots provide professional services. The home is not manipulated for robots to work easily.

TABLE 1.1 SUMMARY OF SURVEYS AND TUTORIAL PAPERS IN SLAM

Topic	Reference
Formulation	[3]-[7]
SLAM back end	[8]
Observability and convergence analysis	[9]
Visual odometry	[10], [11]
Visual place recognition (Loop detection in visual SLAM)	[12]
Bundle adjustment for visual SLAM	[13], [14]
Main open-source SLAM libraries	GTSAM [15], g2o [16], Ceres [17], and iSAM [18]

Fig. 1.1 Classification for indoor service robots (categorization from [21]).

From the perspective of vision-based SLAM, the home environment is quite challenging. It contains plenty of less-textured areas. When the robot with a camera moves too close to the obstacles or objects, all tracked features are easily lost because of occlusion and severe scale changes in the image domain. This situation occurs frequently for home service robots such as robotic vacuums because they must move through every inch of the environment. Input images from robotic vacuums with forward-viewing mono-camera that move throughout every inch of the home only contain average of 151 corner features at 320 × 240 image resolution per image (please see Chapter 4.1 for a detailed explanation of the home dataset). As another dataset for evaluation in this work, Vicon dataset only contain 136 corners at 320 × 240 image resolution per image (please see Chapter 4.2 for a detailed explanation of the Vicon dataset). Comparing with the popular benchmark datasets, this is a very challenging situation for SLAM. The Kitti (outdoor, sequence 00) [22] and RAWSEEDS (indoor, sequence 25b Frontal)

[23], which are popular benchmark datasets, contain an average of 4443 corners at 1241 × 376 image resolution per image (731 for 320 × 240 image size ratio) and 657 corners at 320 × 240 image resolution per image, respectively. Table 1.2 shows the whole results. For corner detection, features from accelerated segment test (FAST) algorithm [24] with threshold 20 is used.

TABLE 1.2

NUMBER OF DETECTED CORNERS FOR WHOLE IMAGES OF VARIOUS DATASETS. ALL RESULTS ARE CONVERTED TO IMAGE RESOLUTION OF 320 × 240.

	Kitti 00 dataset [22]	Bicocca25b Dataset [23]	Home environment dataset (Chapter 4.1)	Vicon dataset (Chapter 4.2)
Average number of detected corners per image	731 (4443 for original image size)	657	151	136
Standard deviation of detected corners per image	304 (2045 for original image size)	406	143	122

Another challenge is that human activities make the home environment highly dynamic. There are moving people and objects that degrade the performance of SLAM resulting in incorrect pose estimation result and the occlusion of the static environment for SLAM. In addition, the environment can be changed during the SLAM process. For example, the locations of the

objects or the illumination can be changed. Figure 1.2 shows several examples of these challenging situations, which are extracted from the home dataset in this work.

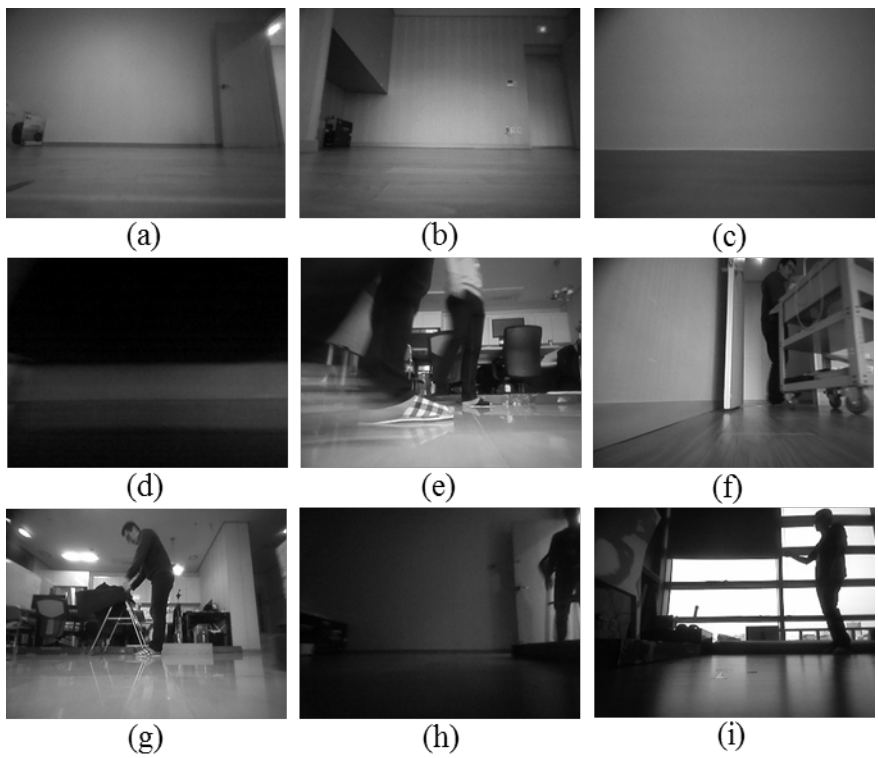


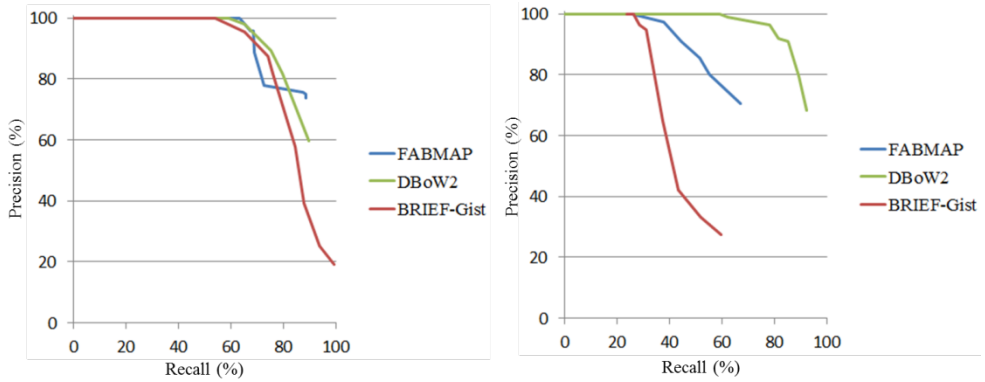
Figure 1.2 Challenging situations extracted the home dataset in this work. Images are captured from a robotic vacuum with a forward-viewing monocular vision sensor. (a)-(b) Less-textured areas. (c) No visual information when robot is too close to the wall. (d) No visual information when robot moved under the sofa. (e)-(f) Moving people and object. (g) Moving person who hang out the wash. (h) Changing illumination when a person turns off the light. (i) Changing environment where a person pull down a roller blind.

Another problem of SLAM in indoor environment especially in home is

reliable loop detection for loop closing. The SLAM problem aims at building a globally consistent pose estimation and environment reconstruction. The SLAM utilizes both ego-motion measurement and loop closing [20]. If there is no loop closing in SLAM, the problem reduces to odometry. The loop closing makes the SLAM more accurate especially in long-term exploration. The state-of-the-art vision-based SLAM algorithms mostly adopt visual bag-of-words (BoW) based loop detection methods. The ORB-SLAM [25] method uses DBoW2 [26] method and the LSD-SLAM [27] method uses OpenFabMap [28] for loop detection. The OpenFabMap is open source version of the original FABMAP [29] which is implemented on OpenCV library [30]. The DBoW2 and FABMAP method are the state-of-the-art visual BoW-based loop detection method, which utilizes BRIEF [31] and SURF [32] descriptor. When these methods are evaluated using the public datasets of Kitti (sequence 00) and RAWSEEDS (sequence Bicocca 25b), they show quite good performances. For comparison, a holistic image descriptor-based BRIEF-Gist method [33] is tested. Based on the ground truth robot poses from the dataset, loop detection results of all robot poses are classified into four classes: true positive (TP), true negative (TN), false positive (FP), and false negative (FN). The precision and recall are evaluated as follows:

$$precision = \frac{TP}{TP + FP}$$
$$recall = \frac{TP}{TP + TN} \tag{1.1}$$

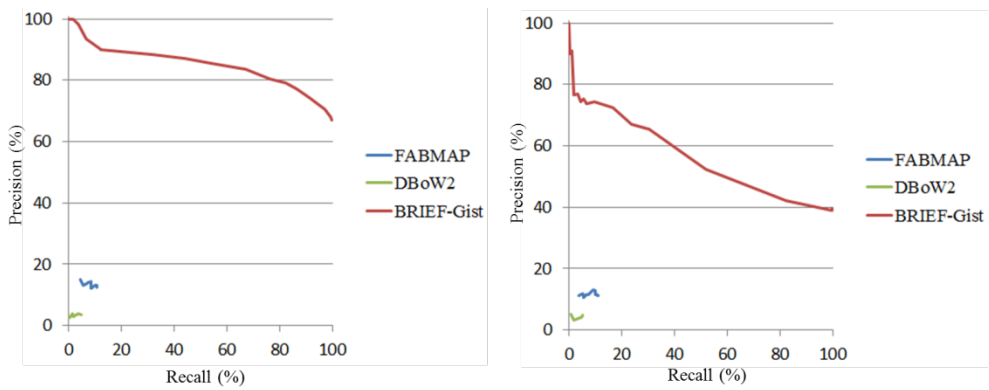
The resulting precision-recall curves are illustrated in Fig. 1.3. In Kitti dataset, DBoW2 and FABMAP methods show maximum recall rate of more than 60% with a precision of 100%. In RAWSEEDS dataset, DBoW2 and FABMAP methods show maximum recall rate of more than 60% and 30% with a precision of 100%, respectively. However, the performance of DBoW2 and FABMAP methods show severe performance degradation in home environment. The precision-recall curves in home environment are illustrated in Fig. 1.4 (Detailed explanations of performance evaluation in home environment are described in Appendix chapter). The main reason for the performance degradation of BoW-based loop detection methods in home environment is that images from the home environment contain very few features. The BoW-based methods find loops using the distributions of extracted descriptors of local features. When the input images contain very few features, the resulting indistinguishable distributions degrade the performance. Though the holistic image descriptor-based method which uses the whole image for loop detection shows better results than BoW-based methods in home environment, the development of reliable loop detection method for SLAM is still required.



(a) Kitti dataset (sequence 00)

(b) RAWSEEDS dataset (Sequence Bicocca 25b)

Figure 1.3 Precision-recall curves of various loop detection (place recognition) methods in Kitti dataset (sequence 00) and RAWSEEDS dataset (sequence Bicocca 25b)



(a) Home loop dataset 1 vs Home loop dataset 2

(b) Home loop dataset 1 vs Home loop dataset 3

Figure 1.4 Precision-recall curves of various loop detection (place recognition) methods in Home environment (Detailed explanations of methodologies and datasets are described in Appendix)

Another relatively unexplored issue is how to adapt SLAM algorithms to robotic platforms with computational constraints [20]. Many current SLAM algorithms are too expensive to run on embedded processors in real-time in

terms of computational speed and memory requirement. One example regarding the memory issue is visual bag-of-words (BoW) for relocalization, data association, and loop detection technique in current state-of-the-art SLAM [25], [27], [34]. To convert input image into BoW representations, the system has to load a huge size of vocabulary tree which is trained in advance. The representative DBoW2 [26] method using BRIEF descriptor [31] requires more than 250 MB memory for vocabulary tree even before the SLAM. In case of SURF descriptor [32] based methods, it requires much more memory. The direct image alignment-based SLAM methods [27], [36], [37] which reconstruct more than thousands points within a single image are also computationally expensive and memory consuming. More studies on visual SLAM that can be applicable to resource constrained platforms are needed.

1.2 Objectives

A globally consistent SLAM allows the robot to perform autonomous navigation and global path planning. Despite the long literature of approaches in vision-based SLAM problem, more research is needed in terms of robustness. In order to develop a reliable vision-based SLAM system applicable to consumer level indoor service robots, this dissertation addresses the following problems:

1) *Vision-based SLAM in challenging indoor environment*. The

performance of SLAM should be ensured in various environments. Especially, low-textured area and dynamic environment are challenging issues for vision-based methods. It is necessary to develop a robust SLAM system which can operable even in those challenging environments.

2) *Real-time SLAM in low-cost embedded processor*. The computational complexity and memory requirements of a SLAM must be considered for real-time operation. To be applicable to indoor service robots, the performance should be verified even in the low-cost embedded processor, in real-time.

1.3 Contributions

This dissertation presents a forward-viewing monocular vision-based SLAM method. A cost-effective mono-camera is adopted as a primary sensor, and robot wheel encoders and a gyroscope are utilized as supplementary sensors. The method is developed to be applicable on a low-cost embedded system for indoor service robot, especially for home environment. The proposed method is quite robust in various challenging indoor environments which contain low-textured areas, moving people, or changing environments. In order to get robust performance in challenging indoor environments, the proposed method adopts the vanishing point (VP) and orthogonal structure assumption. The proposed method can be robustly

executed even when the measurements of VP and lines are intermittently available. The main contributions of this paper are as follows:

1) Estimation models for robot orientation and translation are separately derived in simple equations by utilizing VP and line landmark, respectively.
2) An efficient local map correction method for estimating robot poses and line landmark positions is proposed.
3) A probabilistic loop closure detection method based on BRIEF-Gist [33] image descriptor is proposed.
4) The performance of the proposed method is quantitatively validated through various experiments including challenging indoor datasets and publicly available indoor benchmark datasets.
5) The proposed method is validated through real-time experiment on a low-cost NXP4330Q embedded board [38] and integrated in an autonomous robot navigation system.

1.4 Organization

This dissertation is organized as follows. In Chapter 2, the previous works on vision-based SLAM is presented. Chapter 3 describes the proposed SLAM method in detail. Chapter 4 shows the experimental results of the proposed method. Finally, the conclusion of this dissertation is presented in Chapter 5.

2. Previous Works

In the fields of robotics and computer vision, the visual SLAM has been studied intensively. Researchers have mainly used feature points as a landmark for SLAM. Current SLAM methods can be classified into two categories: filtering-based methods [39]-[46] and optimization-based methods [47]-[54], [25]. In filtering-based methods, every frame is processed by the filter to jointly estimate the camera pose and landmark positions. The accuracy of the estimates is maintained by covariances. A seminal work on monocular visual SLAM was proposed by Davison [39], who used the image patches around corner points as features, and formulated the SLAM problem using the extended Kalman Filter.

On the other hand, optimization-based methods estimate the camera pose and landmark positions in a deterministic way, usually through numerical optimizations called bundle adjustment operated in locally and globally. The representative work of this kind is Parallel Tracking and Mapping (PTAM) [47]. The PTAM method proposed the concept of tracking the camera pose and mapping the environment in two simultaneous threads. This method can handle hundreds of feature points in real-time by putting the time-consuming bundle adjustment into a separated thread. A number of subsequent studies have used the PTAM pipeline. More recently, Engel *et al.* [27] proposed a direct image-alignment-based SLAM called large-scale

direct monocular SLAM (LSD-SLAM). This method tracks and reconstructs high-gradient image points, which results in semi-dense depth maps in large-scale environments. Forster *et al.* [50] proposed a semi-direct monocular VO method called SVO. This method uses direct motion estimation for extracting initial point features and then continues using only these feature points. Recently, Mur-Artal *et al.* [25] proposed an oriented FAST and rotated BRIEF (ORB) point-feature-based monocular SLAM system called ORB-SLAM. This method uses the same ORB features for all SLAM tasks: tracking, mapping, relocalization, and loop closing. The ORB-SLAM method is referred to as the representative point feature-based method in monocular visual SLAM.

Numerous studies have been presented on arbitrary 3D line features-based SLAM [55]-[58], [35]. These methods used the filtering method to formulate the SLAM problem where line segments are parameterized with two end points [55]-[57] or as an infinite line [58] in small 3D space. Recently, Zhang *et al.* [35] proposed a 3D line-based stereo SLAM system. The method uses Plucker line coordinates for line parameterization. They have showed successful experimental results in corridor environments and outdoor environments.

In indoor environment, plenty of studies have used ceiling line features as the landmarks for SLAM. They usually require upward-viewing camera since data association, geometrical modeling, and implementation gets much easier. Also, they utilize corner features as additional landmarks for

SLAM. Jeong *et al.* [59] proposed the upward-viewing camera based SLAM which uses the ceiling line features and corner features as the landmarks for SLAM. Lee and Lee [60] proposed an upward-viewing camera-based SLAM method that uses ceiling line features and corner features for landmark. The algorithm runs in low-cost embedded system in indoor environments. Choi *et al.* [61] proposed the upward-viewing camera based SLAM where the boundaries between ceiling and walls are used as the landmarks for SLAM. Choi *et al.* [62] also proposed the upward-viewing camera based SLAM where the ceiling line features and corner features are used as the landmarks for visual SLAM. On the other hand, the proposed method uses a forward-viewing camera, but it has much technical difficulties in feature extraction and tracking. Nevertheless, a forward-viewing method has significant commercial advantages in that the same camera can be used for obstacle avoidance, home monitoring or as a user interface.

Several studies have applied VP information to attitude estimation and SLAM as is done in this study. These studies usually used the orthogonal structure assumption, which can make the problem easy and reduce the orientation drift. Monocular camera based orientation estimation methods are proposed in [63], [64]. These methods detect orthogonal VPs and estimate the three-axis orientation of the camera using the assumption of structure regularity. Lee *et al.* [65] proposed a laser sensor based algorithmic compass and its application to SLAM; the compass uses VPs as global

features in indoor environments. They have shown that the loop closure effect can be achieved using the VP as a global feature in a corridor environment. Zhang *et al.* [66] proposed a VP-based loop-closure method in a line-based SLAM. They used the property of structural regularity and showed the efficiency of the proposed method in corridor environment. Camposeco and Marc [67] proposed a visual-inertial odometry method which utilizes VPs to reduce angular drift in camera pose estimation. The method tracked the VP directions as a state of an Extended Kalman Filter and used these observations to update the orientation of the camera and the directions of the VP. The advantage of this method is that it allows the use of multiple vanishing directions. Zhou *et al.* [68] proposed visual SLAM using building structure lines called StructSLAM. In this method, the orthogonal structures and the directions of VPs are utilized for parameterizing the line landmarks. The results showed that a reduction in the orientation error in the line landmark induces much better SLAM results in indoor environments even without loop closure.

Inspired by the previous works, VP is utilized to reduce the orientation error by assuming the structural regularity of the indoor environment. In addition, line landmarks are aligned with the directions of the VPs. Unlike the previous works, the estimation model for the robot's orientation and translation are separately derived in simple equations using the VP and line landmark, respectively. Accordingly, we can estimate the robot's pose with reduced computation time.

3. Methodology

3.1 System Overview

In this work, it is assumed that the robot moves around the flat ground. As sensory inputs, images captured from a forward-viewing mono-camera and odometry from robot wheel encoders and gyroscope are used. The camera is 6.3 cm above the floor and slightly tilted 8.7° upward to provide various consumer services such as home monitoring, human-robot interactions or obstacle detection.

The architecture of the proposed SLAM algorithm is shown in Fig. 3.1. The proposed method runs three threads in parallel: tracking thread, mapping thread, and loop closing thread. The flowcharts of three threads are illustrated in Figs. 3.2, 3.3 and 3.4.

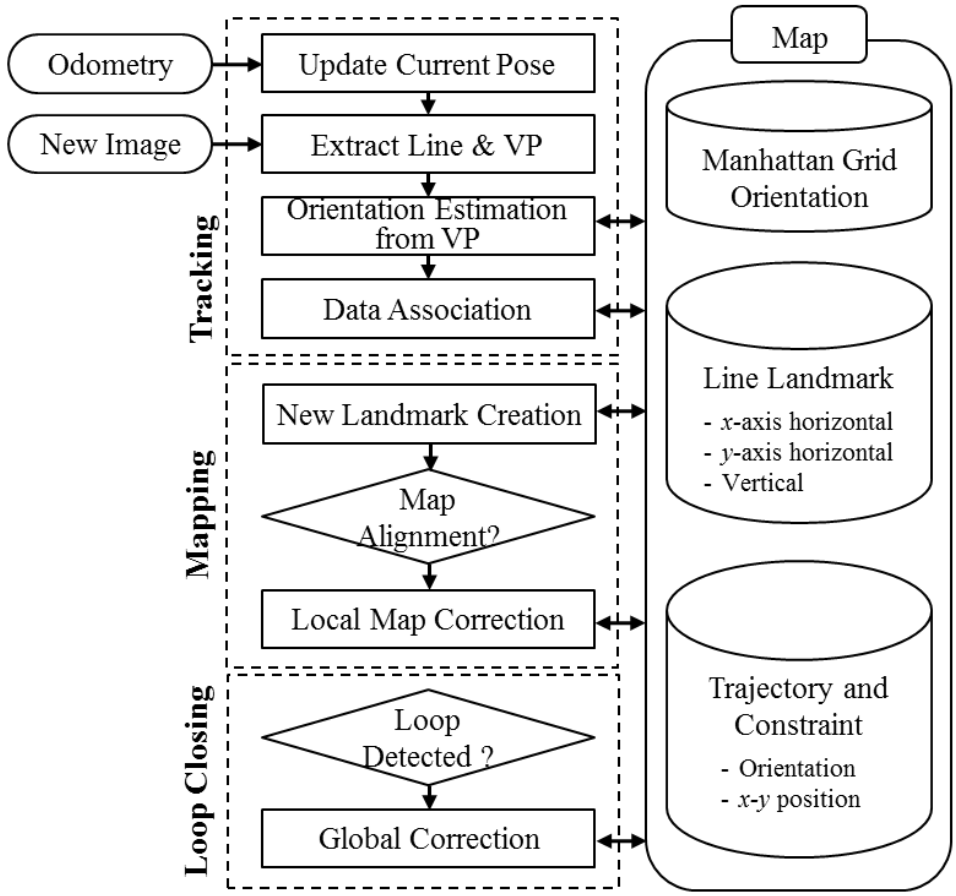


Fig. 3.1 Flowchart of the overall proposed SLAM algorithm.

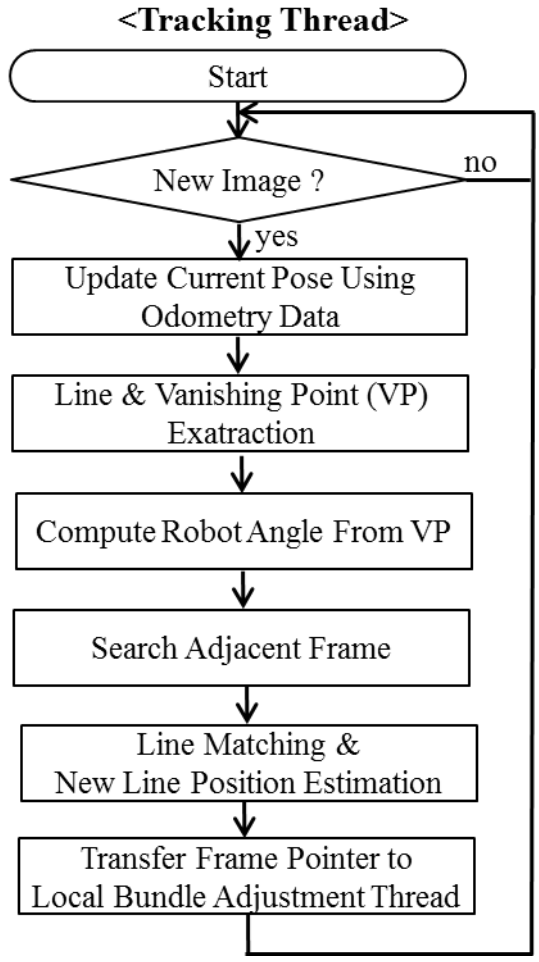


Figure 3.2 Flowchart of Tracking Thread

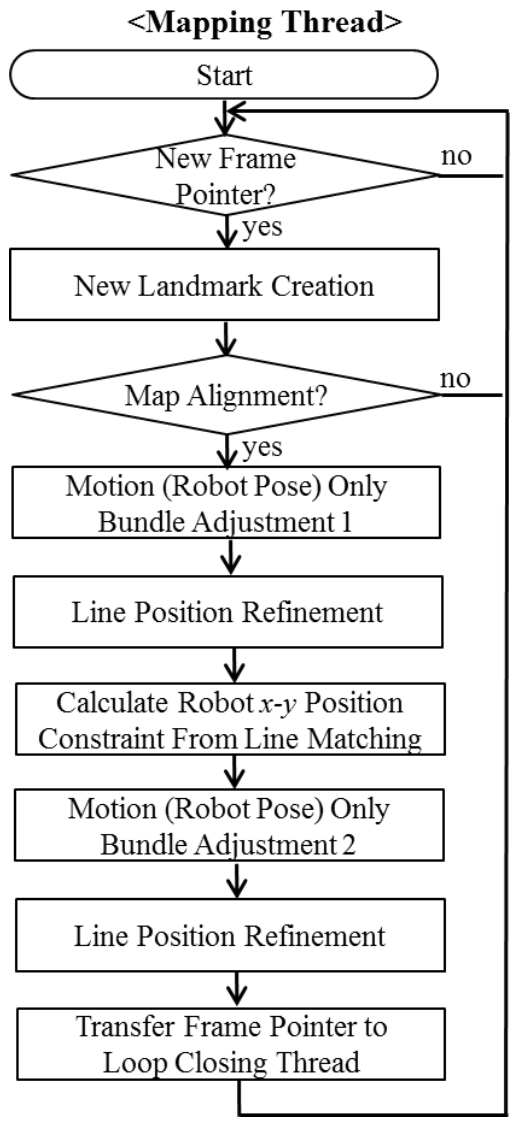


Figure 3.3 Flowchart of Mapping Thread

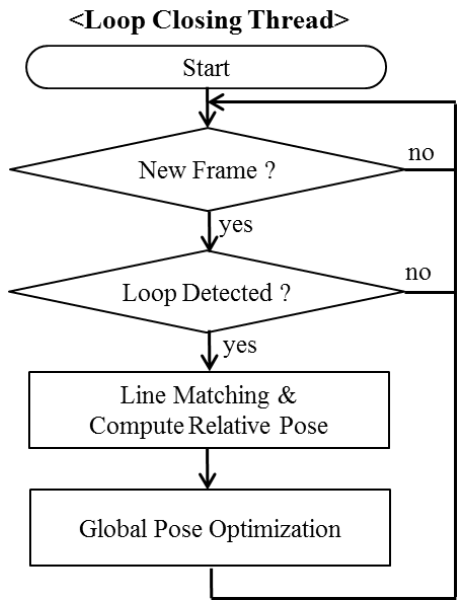


Figure 3.4 Flowchart of Loop Closing Thread

The tracking thread performs four main tasks as follows. Firstly, the robot pose is updated using odometry data. Secondly, line features and vanishing points (VPs) are extracted. Thirdly, the robot heading angle is estimated using VP extraction results which are utilized to reduce the accumulated robot orientation error in the local map correction in mapping thread. The orientation information of global environment is encoded in the extracted VP, and the estimated robot angle from VP can perform global correction of the estimated robot pose. Lastly, data association of lines are conducted. From the data association of lines, line landmark observations are made, and the position of new landmarks are estimated. After processing the above procedures, the frame pointer is transmitted to the local bundle adjustment thread. By utilizing the VP and line landmarks, the proposed method can be

more robust to dynamic environments, since they are not steadily extracted in moving people or objects.

The mapping thread creates new line landmarks using the estimated positions in tracking thread and inserts them in a map database. Afterwards, the mapping thread corrects the estimates of a bundle of robot poses and landmark positions. The local map correction is only conducted for recent N frames and observed line landmarks at the corresponding frames. Unlike the conventional local bundle adjustment, which corrects the robot poses (motion) and landmark positions (structure) simultaneously, the robot poses and landmark positions are refined separately twice to reduce the computational load. After processing the above procedures, the frame pointer is transmitted to the loop closing thread.

The loop closing thread finds for large loops for every input frames. A modified version of BRIEF-Gist [33] is used for loop detection. Once a loop is detected, relative pose is estimated between the current frame and the matched frame. Then, pose graph optimization is conducted using the incremental pose constraints from the estimated robot trajectory and the non-consecutive pose constraints from loop detection. After the global pose correction, the positions of observed line landmarks are re-estimated.

3.2 Manhattan Grid and System Initialization

Most indoor environments can be abstracted as blocks that are stacked together in three dominant directions, which is referred to as a Manhattan grid [69], [70]. This grid gives a natural reference frame for the viewer [71]. The advantage of adopting the Manhattan frame assumption is that both the robot orientation error and line landmark orientation error can be eliminated. Under this assumption, the extracted VPs in the image plane correspond to the three dominant directions of the Manhattan grid.

From the first pose of the robot, the world frame is set according to the initial pose of the robot. The *x-axis* points toward the forward direction of the robot, and the *z-axis* points toward the upward direction. At the system initialization step, the orientation of the Manhattan frame with respect to the world frame around its *z-axis* is averaged up to some predefined number. The initialization is conducted only when the variance of the orientation of the Manhattan frame with respect to the world frame is smaller than some pre-defined angle. In the initialization step, the robot poses are estimated using the odometry data. Figure 3.5 shows the typical example of relationship between Manhattan frame and world frame with same origin. Figure 3.6 shows the example of Manhattan frame configuration in a blueprint of typical home environment. The detailed method of estimating the angle between robot and the Manhattan frame is explained in the next

section. Usually, typical indoor environments can be modeled using one Manhattan frame, but sometimes multiple Manhattan frames are required (known as “Atlanta world” [72]). The proposed method resolves this situation by setting up several Manhattan frames environment as illustrated in Fig. 3.7. The additional Manhattan frame is configured when a dominant direction of the VP is changed in a partitioned grid area.

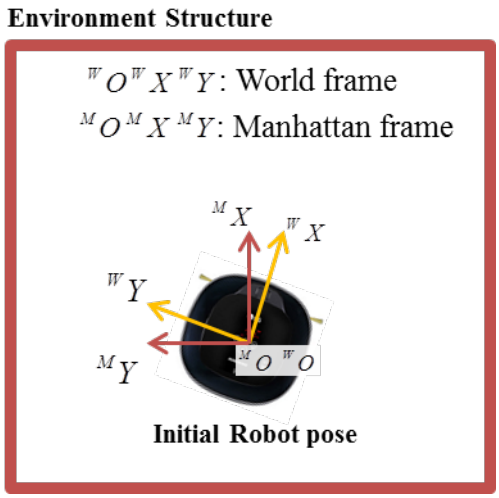


Figure 3.5 Typical example of relationship between Manhattan frame and world frame with same origin

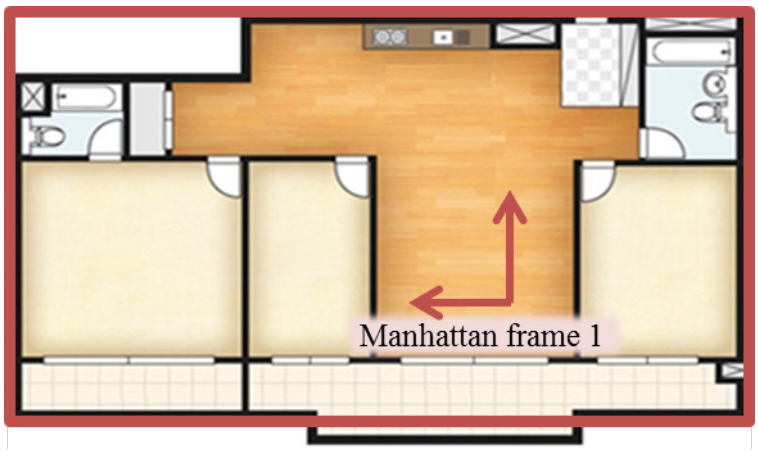


Figure 3.6 Example of Manhattan frame configuration in a blueprint of typical home environment

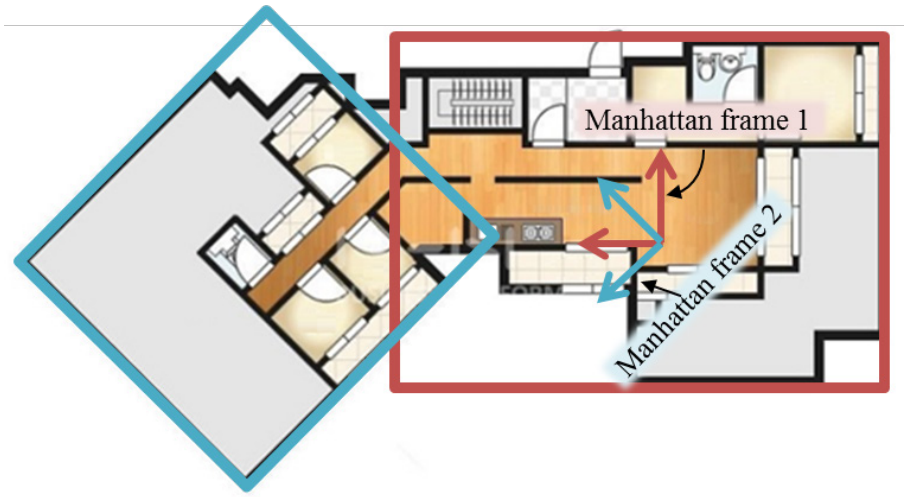


Figure 3.7 Indoor environment which can be modeled using multiple Manhattan grids

3.3 Vanishing Point Based Robot Orientation Estimation

To extract a VP, a histogram equalization is conducted only to images

with an average intensity lower than a predefined threshold for better extraction of line segments. Then the line segments are extracted using the line segment detector [73]. For the robust estimation of the VP, line segments with a length less than 15 pixels are eliminated, because short line segments tend to be noisy observations. Then the VP is extracted using the algorithm proposed by Zhang *et al.* [74]. Figure 3.8 shows some examples of VP extraction results in typical home environments. It can be seen that VPs can be robustly extracted even though the images contain moving people. In this work, the VPs of horizontal lines in the image are utilized to estimate the orientation between the robot and the Manhattan grid. These VPs are reliable when there are several lines appearing at the top of the image. Hence, images without any horizontal lines at the top region of the image are skipped.

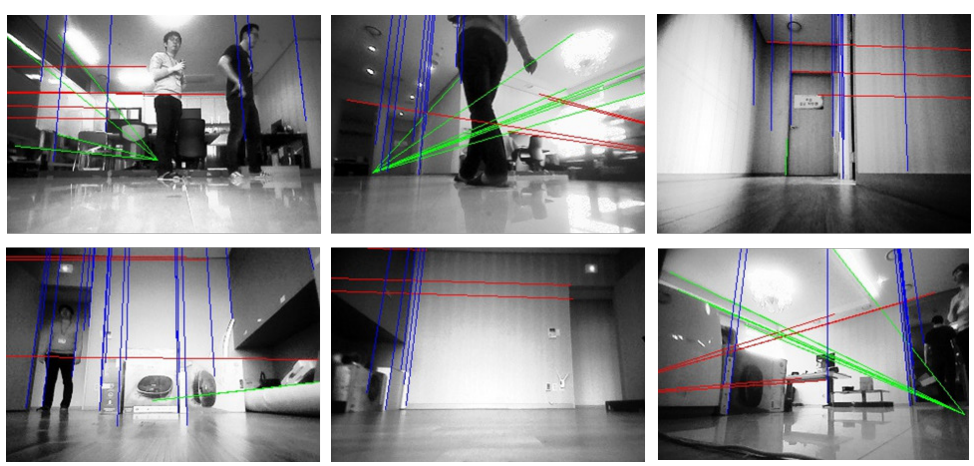


Fig. 3.8. Examples of VP extraction results in a typical home environment

Figure 3.9 shows a schematic for the camera and robot coordinate systems. From the extracted VP, three dominant orthogonal line direction vectors are obtained with respect to the camera frame. These three direction vectors are expressed as ${}^{C}\mathbf{vp}_1$, ${}^{C}\mathbf{vp}_2$, and ${}^{C}\mathbf{vp}_3$. Since a slightly tilted forward-viewing mono-camera at a fixed angle of θ_{tilt} is used in this work, the three direction vectors are transformed with respect to the robot's frame using simple rotational matrix computation as follows:

$$ {}^{R}\mathbf{vp}_i = {}^{R}_{C}R \cdot {}^{C}\mathbf{vp}_i = \begin{bmatrix} {}^{R}vp_{i,x} \\ {}^{R}vp_{i,y} \\ {}^{R}vp_{i,z} \end{bmatrix} = \begin{bmatrix} 0 & -\sin\theta_{tilt} & \cos\theta_{tilt} \\ 1 & 0 & 0 \\ 0 & \cos\theta_{tilt} & \sin\theta_{tilt} \end{bmatrix} \begin{bmatrix} {}^{C}vp_{i,x} \\ {}^{C}vp_{i,y} \\ {}^{C}vp_{i,z} \end{bmatrix}. \qquad (3.1) $$

Since the robot only rotates with respect to the *z-axis*, the orientation of the robot with respect to the Manhattan frame structure can be simply calculated using the estimated VP direction vectors. First, from the three VP direction vectors, the one VP is chosen whose direction is closest to the *y-axis* of the robot frame. This VP direction vector is called ${}^{R}\mathbf{vp}_{y-\text{dominent}}$ which is shown by a red arrow in Fig. 3.9. The direction vector ${}^{R}\mathbf{vp}_{y-\text{dominent}}$ is parallel to the Manhattan grid plane, which can be simply considered to be a wall in front of the robot. Second, the robot's orientation with respect to the Manhattan frame is computed by projecting the direction vector ${}^{R}\mathbf{vp}_{y-\text{dominent}}$ to the *x-y* plane of the robot's frame. Examples of vanishing point extraction with estimated angle between the robot and Manhattan frame are illustrated in Fig. 3.10. Finally, the robot orientation with respect

to the world frame is calculated. The angle between the Manhattan frame and the world frame is known from the initialization; therefore, this angle is added to the robot's orientation in the Manhattan frame, resulting in the robot's orientation in the world frame.

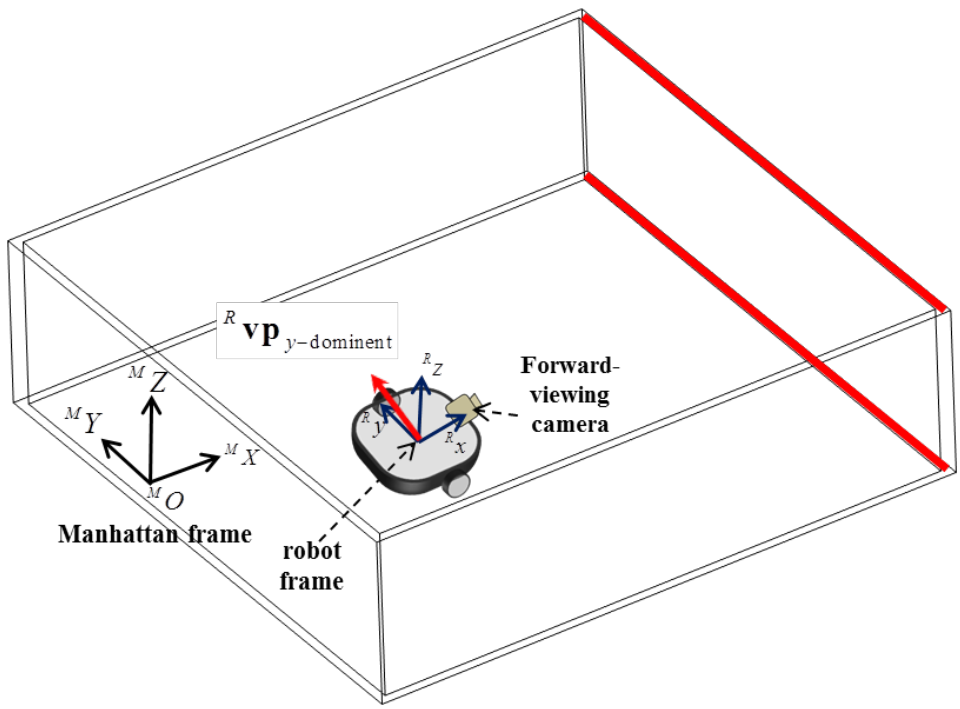


Figure 3.9 A schematic of mobile robot coordinate system in Manhattan grid.

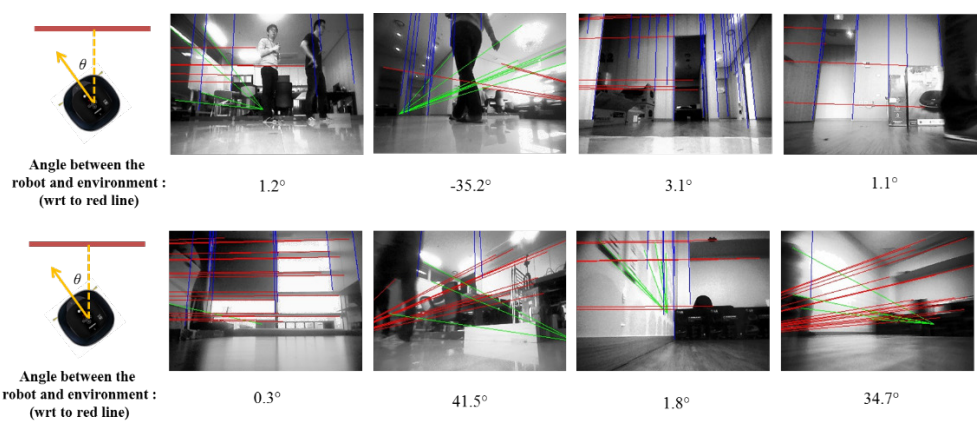


Figure 3.10 Examples of vanishing point extraction with estimated angle

between the robot and Manhattan frame.

3.4 Line Landmark Position Estimation

To parameterize and estimate the position of line landmarks, notations are presented. The robot's pose in 2D space at time step k is expressed as $\mathbf{x}_k = \begin{pmatrix} x_{r,k} & y_{r,k} & \theta_{r,k} \end{pmatrix}^T$. The camera is mounted in front of the robot as shown in Fig. 3.11. The pose of the camera in 2D space can be expressed using a simple transformation as follows:

$$\mathbf{x}_{c,k} = \begin{pmatrix} x_{c,k} \\ y_{c,k} \\ \theta_{c,k} \end{pmatrix} = \begin{pmatrix} x_{r,k} + d_1 \cos\theta_{r,k} - d_2 \sin\theta_{r,k} \\ y_{r,k} + d_1 \sin\theta_{r,k} + d_2 \cos\theta_{r,k} \\ \theta_{r,k} \end{pmatrix}. \tag{3.2}$$

Considering the tilting angle θ_t of the camera, the camera matrix for point projection can be expressed as:

$$\mathbf{P} = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} -\sin\theta_c & \cos\theta_c & 0 & x_c \sin\theta_c - y_c \cos\theta_c \\ -c_t \cos\theta_c & -c_t \sin\theta_c & s_t & c_t x_c \cos\theta_c + c_t y_c \sin\theta_c \\ s_t \cos\theta_c & s_t \sin\theta_c & c_t & -s_t x_c \cos\theta_c - s_t y_c \sin\theta_c \end{pmatrix}, \tag{3.3}$$

where (f_x, f_y) and (c_x, c_y) are the focal length and the principal point, respectively, of the camera in the image domain. The parameters s_t and c_t denote $\sin\theta_t$ and $\cos\theta_t$, respectively. Using this camera matrix, the line projection model can be expressed as follows [75]:

$$\tilde{\mathbf{l}} = \begin{pmatrix} l_1 \\ l_2 \\ l_3 \end{pmatrix} = \begin{pmatrix} (\mathbf{p}^{2T} \cdot \tilde{\mathbf{a}}) \cdot (\mathbf{p}^{3T} \cdot \tilde{\mathbf{b}}) - (\mathbf{p}^{2T} \cdot \tilde{\mathbf{b}})(\mathbf{p}^{2T} \cdot \tilde{\mathbf{a}}) \\ (\mathbf{p}^{3T} \cdot \tilde{\mathbf{a}}) \cdot (\mathbf{p}^{1T} \cdot \tilde{\mathbf{b}}) - (\mathbf{p}^{3T} \cdot \tilde{\mathbf{b}})(\mathbf{p}^{1T} \cdot \tilde{\mathbf{a}}) \\ (\mathbf{p}^{1T} \cdot \tilde{\mathbf{a}}) \cdot (\mathbf{p}^{2T} \cdot \tilde{\mathbf{b}}) - (\mathbf{p}^{1T} \cdot \tilde{\mathbf{b}})(\mathbf{p}^{2T} \cdot \tilde{\mathbf{a}}) \end{pmatrix} \tag{3.4}$$

where $\tilde{\mathbf{l}}$ is the projected line in an image plane; $(\tilde{\mathbf{a}}, \tilde{\mathbf{b}})$ are the two endpoints of the line; and $\mathbf{p}^{iT}$ is the *i*th row vector of $\mathbf{P}$.

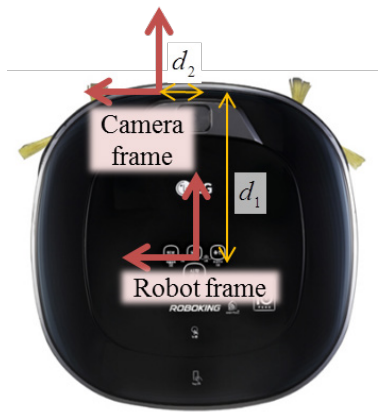


Figure 3.11 Robot frame and camera frame

As landmarks for SLAM, three types of line features are used in this work. These are composed of the vertical line, the *x-axis* horizontal line, and the *y-axis* horizontal line with respect to the Manhattan frame. Fig. 3.12 illustrates parameterization of three types of line landmarks. The position of the lines with respect to the world frame can be calculated using the rotation matrix computation between the Manhattan frame and the world frame. To parameterize the line landmarks, both endpoints are used.

For the vertical line, the endpoints $\tilde{\mathbf{a}}_v$ and $\tilde{\mathbf{b}}_v$ can be expressed as follows:

$$\begin{aligned}\tilde{\mathbf{a}}_v &= \begin{pmatrix} x_v & y_v & z_1 & 1 \end{pmatrix}^T \\ \tilde{\mathbf{b}}_v &= \begin{pmatrix} x_v & y_v & z_2 & 1 \end{pmatrix}^T,\end{aligned} \tag{3.5}$$

where x_v and y_v are variables which are estimated continuously by the SLAM. Variables z_1 and z_2 are simply calculated using the pin-hole camera model and observed line end points in image plane after the x_v and y_v are estimated. For *x-axis* horizontal line, end-points $\tilde{\mathbf{a}}_{h_x}$ and $\tilde{\mathbf{b}}_{h_x}$ can be expressed as follows:

$$\begin{aligned}\tilde{\mathbf{a}}_{h_x} &= \begin{pmatrix} x_1 & y_{h_x} & z_{h_x} & 1 \end{pmatrix}^T \\ \tilde{\mathbf{b}}_{h_x} &= \begin{pmatrix} x_2 & y_{h_x} & z_{h_x} & 1 \end{pmatrix}^T,\end{aligned} \tag{3.6}$$

where y_{h_x} and z_{h_x} are variables which are estimated continuously by the SLAM. Similarly, the end-points $\tilde{\mathbf{a}}_{h_y}$ and $\tilde{\mathbf{b}}_{h_y}$ of *y-axis* horizontal line can be expressed as follows:

$$\begin{aligned}\tilde{\mathbf{a}}_{h_y} &= \begin{pmatrix} x_{h_y} & y_1 & z_{h_y} & 1 \end{pmatrix}^T \\ \tilde{\mathbf{b}}_{h_y} &= \begin{pmatrix} x_{h_y} & y_2 & z_{h_y} & 1 \end{pmatrix}^T,\end{aligned} \tag{3.7}$$

where x_{h_y} and z_{h_y} are variables which are estimated continuously by the SLAM.

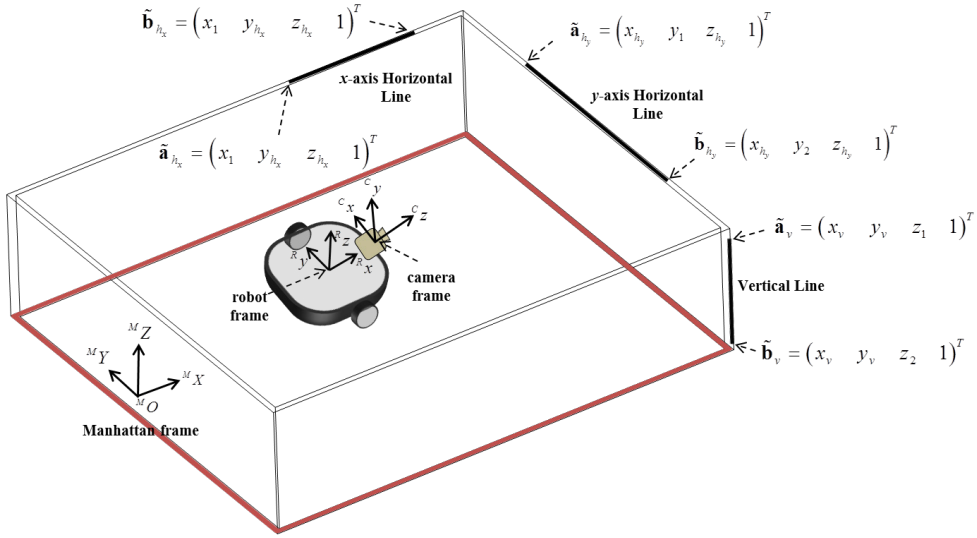


Figure 3.12 Line landmark parameterization

Using the line projection model of (3.4) and the previously defined line landmark parameterization from (3.5) to (3.7), the equation for position estimation of landmarks can be derived as a simple linear model. The projection equation of a vertical line is as follows:

$$(l_1 a_{v,2} - l_3 a_{v,1}) x_v + (l_1 b_{v,2} - l_3 b_{v,1}) y_v = (l_3 c_{v,1} - l_1 c_{v,2}), \tag{3.8}$$

where

$$\begin{aligned} a_{v,1} &= \mathbf{P}_{21}\mathbf{P}_{33} - \mathbf{P}_{23}\mathbf{P}_{31} \\ a_{v,2} &= \mathbf{P}_{11}\mathbf{P}_{23} - \mathbf{P}_{13}\mathbf{P}_{21} \\ b_{v,1} &= \mathbf{P}_{22}\mathbf{P}_{33} - \mathbf{P}_{23}\mathbf{P}_{32} \\ b_{v,2} &= \mathbf{P}_{12}\mathbf{P}_{23} - \mathbf{P}_{13}\mathbf{P}_{22} \\ c_{v,1} &= \mathbf{P}_{24}\mathbf{P}_{33} - \mathbf{P}_{23}\mathbf{P}_{34} \\ c_{v,2} &= \mathbf{P}_{14}\mathbf{P}_{23} - \mathbf{P}_{13}\mathbf{P}_{24} \end{aligned}. \tag{3.9}$$

Similarly, the projection equation of a *x-axis* horizontal line is as follows:

$$(l_2 a_{hx,1} - l_3 a_{hx,2}) y_{h_x} + (l_2 b_{hx,1} - l_3 b_{hx,2}) z_{h_x} = (l_3 c_{hx,1} - l_2 c_{hx,2}), \tag{3.10}$$

where

$$
\begin{aligned}
a_{hx,1} &= \mathbf{P}_{12}\mathbf{P}_{21} - \mathbf{P}_{11}\mathbf{P}_{22} \\
a_{hx,2} &= \mathbf{P}_{11}\mathbf{P}_{32} - \mathbf{P}_{12}\mathbf{P}_{31} \\
b_{hx,1} &= \mathbf{P}_{13}\mathbf{P}_{21} - \mathbf{P}_{11}\mathbf{P}_{23} \\
b_{hx,2} &= \mathbf{P}_{11}\mathbf{P}_{33} - \mathbf{P}_{13}\mathbf{P}_{31} \\
c_{hx,1} &= \mathbf{P}_{11}\mathbf{P}_{34} - \mathbf{P}_{14}\mathbf{P}_{31} \\
c_{hx,2} &= \mathbf{P}_{14}\mathbf{P}_{21} - \mathbf{P}_{11}\mathbf{P}_{24}
\end{aligned}
\quad . \tag{3.11}
$$

Lastly, the projection equation of a *y-axis* horizontal line is as follows:

$$
(l_2 a_{hy,1} - l_3 a_{hy,2}) x_{h_y} + (l_2 b_{hy,1} - l_3 b_{hy,2}) z_{h_y} = (l_3 c_{hy,1} - l_2 c_{hy,2}) , \tag{3.12}
$$

where

$$
\begin{aligned}
a_{hy,1} &= \mathbf{P}_{11}\mathbf{P}_{22} - \mathbf{P}_{12}\mathbf{P}_{21} \\
a_{hy,2} &= \mathbf{P}_{12}\mathbf{P}_{31} - \mathbf{P}_{11}\mathbf{P}_{32} \\
b_{hy,1} &= \mathbf{P}_{13}\mathbf{P}_{22} - \mathbf{P}_{12}\mathbf{P}_{23} \\
b_{hy,2} &= \mathbf{P}_{12}\mathbf{P}_{33} - \mathbf{P}_{13}\mathbf{P}_{32} \\
c_{hy,1} &= \mathbf{P}_{12}\mathbf{P}_{34} - \mathbf{P}_{14}\mathbf{P}_{32} \\
c_{hy,2} &= \mathbf{P}_{14}\mathbf{P}_{22} - \mathbf{P}_{12}\mathbf{P}_{24}
\end{aligned}
\quad . \tag{3.13}
$$

Since equations (3.8), (3.10) and (3.12) are linear, the position of line landmarks can be easily estimated from the line matching results from different robot locations. When the same line is matched over several images, the position of the line can be simply calculated by linear least-square method.

For further performance improvement in position estimation of line landmarks, an additional constraint is imposed. Sometimes, when robot moves forward with a forward mono-camera, the position of feature points or lines are estimated closer to robot than the actual position. This is due to the difficulty in acquiring sufficient disparity for accurate triangulation of

the feature. In this case, the performance of SLAM could degrade. So, it is assumed that the position of the matched line features exist farther than 1.5 m in front of the camera. The example of this inequality constraint on *y*-axis horizontal line is illustrated in Fig. 3. 13. Using this inequality constraint and line projection equations, the problem can be easily solved by quadratic programming.

For computational efficiency, the image patches with small size (e.g. 11 × 11) around the midpoint of extracted line are used for data association of line features. The example of line matching is shown in Fig. 3.14.

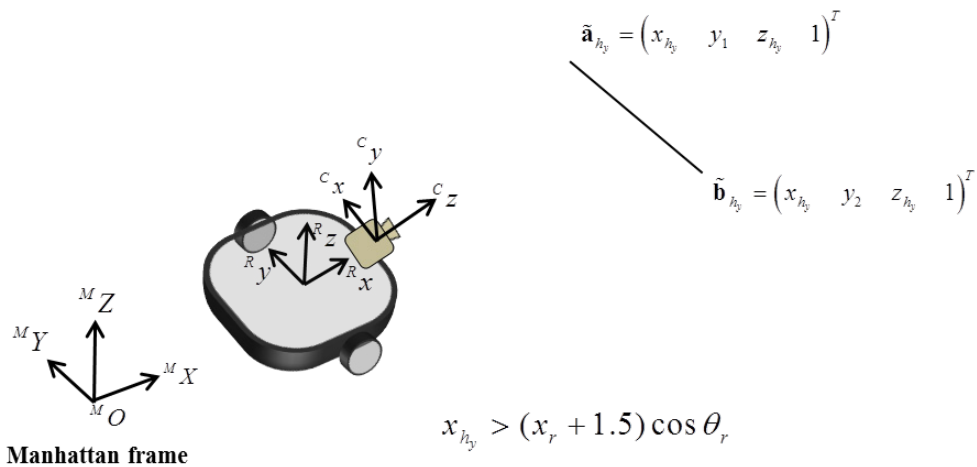


Figure 3.13 The example of inequality constraint of y-axis horizontal line for estimation of line position

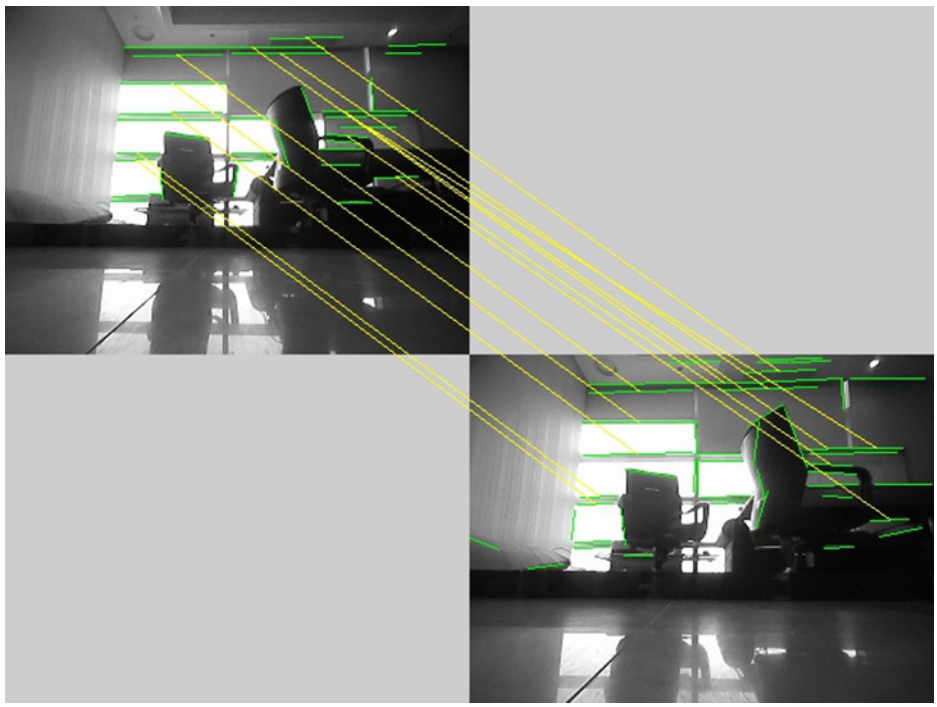


Figure 3.14 Example of line matching result

3.5 Camera Position Estimation

From the extracted line landmark with a known position, the camera position $\mathbf{x} = \begin{pmatrix} x_c & y_c \end{pmatrix}^T$ can be estimated. These estimates of the camera x-y positions are used in the local map correction step in the mapping thread. Using the previously defined landmark parameterization and the line projection model, the equations for estimating the camera position can be derived. Using the vertical line projection, the camera position can be expressed as follows:

$$(a_v \cdot l_1 - b_v \cdot l_3)x_c + (c_v \cdot l_1 - d_v \cdot l_3)y_c = e_v \cdot l_3 - f_v \cdot l_1 \qquad (3.14)$$

where

$$
\begin{aligned}
a_v &= f_x f_y s_t \sin\theta + f_x c_y c_t \sin\theta - f_y c_x \cos\theta \\
b_v &= f_y \cos\theta \\
c_v &= -f_x f_y s_t \cos\theta - f_y c_y c_t \cos\theta - f_y c_x \sin\theta \\
d_v &= f_y \sin\theta \\
e_v &= -f_y (x_v \cos\theta + y_v \sin\theta) \\
f_v &= (f_x f_y s_t y_v + f_x c_y c_t y_v + f_y c_x x_v)\cos\theta \\
&\quad + (f_y c_x y_v - f_x f_y s_2 x_v - f_x c_y c_t x_v)\sin\theta
\end{aligned}
\quad . \qquad (3.15)
$$

Similarly, using the *x-axis* horizontal line projection, the camera position can be expressed as follows:

$$
(a_{hx} \cdot l_3 - b_{hx} \cdot l_2) y_r = c_{hx} \cdot l_2 - d_{hx} \cdot l_3 \qquad (3.16)
$$

where

$$
\begin{aligned}
a_{hx} &= f_x s_t \\
b_{hx} &= f_x f_y c_t - f_x c_y s_t \\
c_{hx} &= (f_x c_y s_t - f_x f_y c_t) y_{h_x} \\
&\quad + (f_x f_y s_t \sin\theta + f_x c_y c_t \sin\theta - f_y c_x \cos\theta) z_{h_x} \\
d_{hx} &= -f_x s_t y_{h_x} - f_x c_t \sin\theta z_{h_x}
\end{aligned}
\quad . \qquad (3.17)
$$

Finally, using the *y-axis* horizontal line projection, the camera position can be expressed as follows:

$$
(a_{hy} \cdot l_3 - b_{hy} \cdot l_2) x_r = c_{hy} \cdot l_2 - d_{hy} \cdot l_3 , \qquad (3.18)
$$

where

$$
\begin{aligned}
a_{hy} &= -f_x s_t \\
b_{hy} &= f_x c_y s_t - f_x f_y c_t \\
c_{hy} &= (f_x f_y c_t - f_x c_y s_t) x_{h_y} \\
&\quad - (f_y c_x \cos\theta + f_x f_y s_t \cos\theta + f_x c_y c_t \cos\theta) z_{h_y} \\
d_{hy} &= f_x s_t x_{h_y} + f_x c_t \cos\theta z_{h_y}
\end{aligned}
\quad . \qquad (3.19)
$$

Using these three types of extracted line landmarks, the camera position can be estimated using the linear-least squares method.

3.6 Local Map Correction

The local map correction process corrects the estimates of a bundle of camera poses and landmark positions. The correction is only conducted for recent N frames and the observed line landmarks at the corresponding frames. When measurements from the camera (i.e., robot orientation from VP and robot x-y position measurement) are not valid due to low-textured areas, occlusions, or changing environments, the local map correction process is skipped. Subsequently, when valid measurements are available, local map correction is conducted, which includes the skipped frames. The flowchart of the local map correction process is shown in Fig. 3.15.

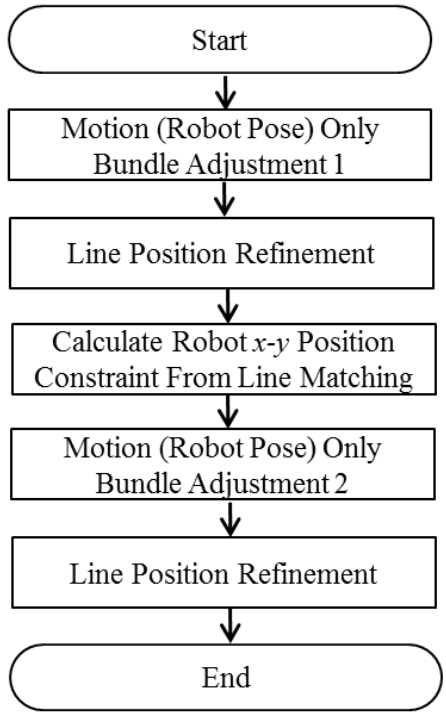


Figure 3.15 Flowchart of local map correction process.

The motion only bundle adjustment 1 corrects the camera poses using the odometry mesurements and the VP-based orientation measurements in tracking thread. The cost function to be minimized is formulated as follows:

$$E(\mathbf{x}_{c,s}^{w},\cdots,\mathbf{x}_{c,k}^{w}) = \sum_{i} \| (\mathbf{z}_{o,i} - (\mathbf{x}_{c,i}^{w} \ominus \mathbf{x}_{c,i-1}^{w})) \|_{\Sigma_o}^{2} + \sum_{i} R_i (z_{VP,s,i}^{w} - (\theta_i - \theta_s))^2 \quad (3.20)$$

where $\mathbf{x}_{c,s}$ is the *s*th 2D camera poses; $\mathbf{x}_{c,k}$ is the current camera pose; $\mathbf{z}_{o,i}$ is the odometry measurements which is the relative pose between the *(i-1)*th camera pose and *i*th camera pose; $\ominus$ is the inverse pose composition operator; $z_{VP,s,i}$ is the VP-based orientation measurements with respect to the *s*th camera orientation; θ_i is the *i*th camera orientation estimation; Σ_{odo}

is the covariance matrix of odometry measurement; and R_i is the covariance of VP-based orientation measurement. The variable *s* indicates the start index of the local map correction. The covariance of the VP-based orientation is inversely proportional to the ratio of inliers in estimating the VPs. The inverse pose composition operator is a relative transformation between the two camera poses $\mathbf{x}_i$ and $\mathbf{x}_j$ defined as follows:

$$\mathbf{z} = \mathbf{x}_i \ominus \mathbf{x}_j = \begin{bmatrix} (x_i - x_j)\cos\theta_j + (y_i - y_j)\sin\theta_j \\ -(x_i - x_j)\sin\theta_j + (y_i - y_j)\cos\theta_j \\ \theta_i - \theta_j \end{bmatrix} . \quad (3.21)$$

The Levenberg-Marquart algorithm using a g2o framework [16] is executed to minimize equation (3.21). We can obtain the estimates of the bundle of camera poses from the motion-only bundle adjustment 1 step; therefore, the accuracy of the corresponding landmark position estimation can be improved. In this regard, line position is re-estimated using the method proposed in Chapter 3.4.

The next step is to calculate the *x*-*y* position of cameras using the method in Chapter 3.5 followed by motion only bundle adjustment 2. The differences between the motion only bundle adjustments 1 and 2 are the *x*-*y* camera pose constraints. The cost function to be minimized is formulated as follows:

$$\begin{aligned} E(\mathbf{x}^w_{c,s}, \cdots, \mathbf{x}^w_{c,k}) = & \sum_i \| (\mathbf{z}_{o,i} - (\mathbf{x}^w_{c,i} \ominus \mathbf{x}^w_{c,i-1})) \|^2_{\Sigma_o} \\ & + \sum_i R_i (z^w_{VP,s,i} - (\theta_i - \theta_s))^2 + \sum_i \| (\mathbf{z}^w_{xy,i} - \mathbf{x}^w_{xy,i}) \|^2_{\Sigma_{xy}} \end{aligned} , \quad (3.22)$$

where $\mathbf{z}_{xy,i}$ is the *x*-*y* position measurement at the *i*th camera obtained from the line matching results; $\mathbf{x}_{xy,i}$ is the current estimation of the *x*-*y* position at *i*th camera; and Σ_{xy} is the covariance matrix of the *x*-*y* position measurement. Σ_{xy} is determined proportional to the inverse of the residual error for estimating the *x*-*y* position.

Finally, the line position is again estimated using the method proposed in Chapter 3.4. During the final line position estimation step, if the residual error divided by the number of observations in the least-squares estimation is larger than a pre-defined threshold, the line landmark is discarded.

3.7 Loop Closing

The Loop closing thread finds large loops for every input frames. As referred in the Chapter 1.1, visual bag-of-words based loop detection algorithms require huge memory to load vocabulary tree. As an alternative, whole image descriptor, i.e., holistic image descriptor is used for loop detection. Among various holistic image descriptors, BRIEF-Gist [33] method is utilized considering for fast computation.

3.7.1 Extracting Multiple BRIEF-Gist descriptors

The major weak point of holistic image descriptor approaches is that it is vulnerable to viewpoint changes. Let's assume that two images captured from two different robot poses are compared. If two robot poses are exactly same, the original BRIEF-Gist would be sufficient. However, when robot is revisiting the same place, it would be more probable that the two robot poses are slightly different which causes viewpoint changes. There would be infinitely many cases of two robot poses, only four cases are considered in this dissertation as shown in Fig. 3.16 for simplicity. The overlapped frustum region of viewpoint in each image plane should be considered. It can be easily expected that the recall performance can be increased when multiple BRIEF-Gist descriptors from multiple image regions are extracted. In this dissertation, three BRIEF-Gist descriptors are extracted from a single image, i.e. left 80% of the image, right 80% of the image, and original image. When comparing two images captured from two places, the extracted three BRIEF-Gist descriptors are compared each other. Fig. 3.17 shows the process of extracting and comparing scene similarity of the proposed method.

For further improvement of performance, the images are filtered by testing possession of sufficient information for place recognition in image before extracting the multiple BRIEF-Gist descriptors. When robot moves in a home environment, input images contain insufficient features or textures for place recognition in numerous cases. For example, when robot moves forward in home environment, no visual information can be captured if the

camera is located in front of wall or obstacles. Blur is another problem especially when robot rotates. Fig. 3.18 shows typical examples of image inputs when robot moves the home environment. When these images are used for place recognition, the precision of the algorithm degrades drastically. These images can be simply discarded using the number of extracted lines in image from tracking thread. When the number of extracted lines does not exceed the certain threshold, the image is discarded. In this dissertation, the threshold is set to 30.

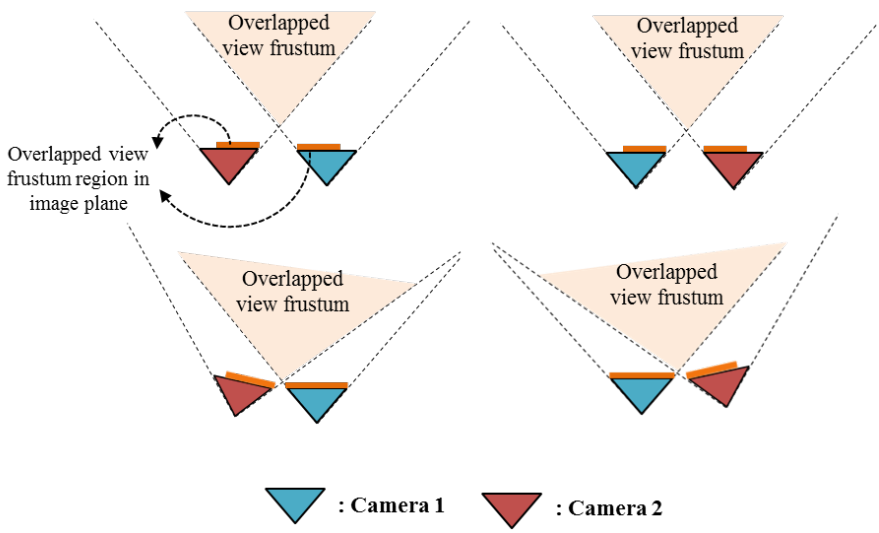


Figure 3.16 The relative pose cases when revisiting of the same place occurred

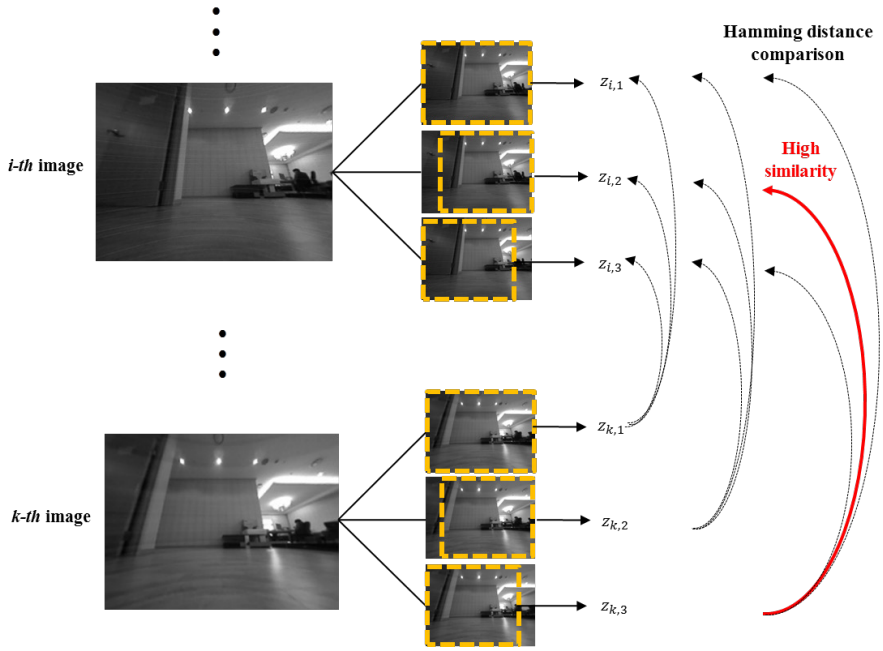


Figure 3.17 Process of extracting and comparing scene similarity by extracting multiple scene descriptors of the proposed method

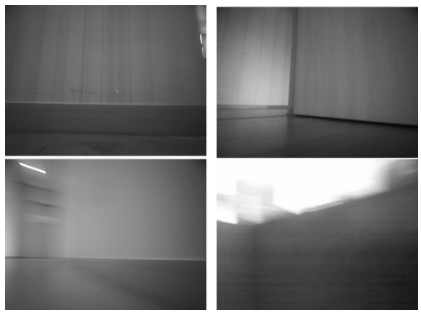


Figure 3.18 Typical examples of image inputs with less features and textures for place recognition in home environment

3.7.2 Data Structure for Fast Comparison

The conventional holistic image descriptor based place recognition

algorithms is done by comparing the previously seen places in an incremental manner. The computation times for comparison increase as the number of previously seen places increase.

A data structure is proposed with faster comparison between current image and previous images using a simple linked list structure. When the binary descriptor spaces are hierarchically divided, the nearest descriptor from the current descriptor can be quickly accessed. However, since the prior distributions of descriptors are unknown, the data structure is incrementally built when images are captured. When the new image is captured, the extracted BRIEF-Gist descriptor is linked to a nearest seed descriptor or registered as a new seed descriptor according to the distance between them. The proposed data structure is just a simple clustering of input descriptors with fixed seeds. When detecting the previously seen place, the current descriptor is firstly compared with the seed descriptors. If the distance between the current descriptor and a certain seed descriptor is low enough, the current descriptor checks all linked descriptor to the seed. In this way, the descriptor cluster with large distance can be simply passed over with just one comparison. Fig. 3.19 shows the example of data structure of the proposed method where $H(\cdot,\cdot)$, $r_{cluster}$, th, $\mathbf{z}_i$ be the Hamming distance, threshold of Hamming distance for deciding the same cluster, the threshold of Hamming distance for detecting the same place, and the extracted BRIEF-Gist from the image, respectively.

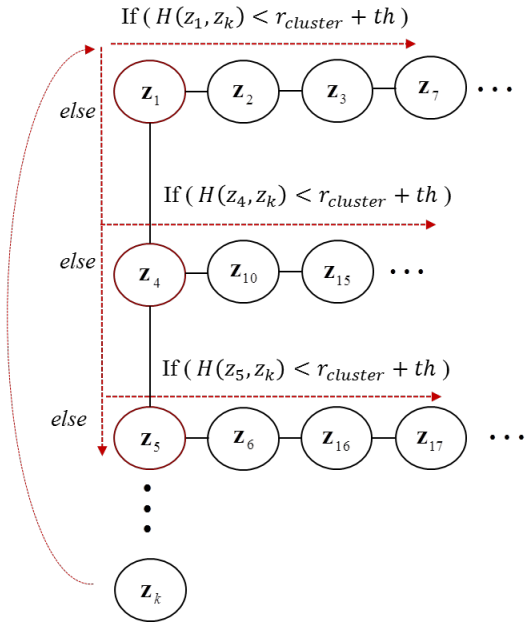


Figure 3.19 The data structure for fast comparison of the proposed method

3.7.3 Bayesian Filtering based Loop Detection

For the final detection of large loop, a Bayesian filtering framework is utilized. The Bayesian filtering method increases the tolerance of the loop detection to noisy responses generated from the raw matching scores [76]-[78]. The proposed method is inspired by the method of [77]. As proposed in [77], the recursive Bayes filter form of the probability that the current location L_k is the same as some previously seen location L_i can be expressed as follows:

$$P(L_k = L_i \mid Z^k) = \frac{P(Z^k \mid L_k = L_i) \cdot P(L_k = L_i \mid Z^{k-1})}{P(Z_k \mid Z^{k-1})} = \frac{P(d_{ki} \mid L_k = L_i) \cdot P(L_k = L_i \mid Z^{k-1})}{P(d_{ki})} \tag{3.23}$$

where d_{ki} is the difference between the whole image descriptors at time *i* and the current time index *k*; Z_k is the image descriptor observation at time *k*; and Z^k is the sequence of observations made at each time step thus far. The probability of $P(d_{ki} \mid L_k = L_i)$ and $P(d_{ki})$ is calculated using a frequency-based method where the probability distribution is estimated from several samples from environment in advance. Unlike the previous works, robot rotation is considered when estimating the probability of $P(L_k = L_i \mid Z^{k-1})$. Assuming that the current location is similar to the previous location, the probability distribution of $P(L_k = L_i \mid Z^{k-1})$ can be approximated to $P(L_{k-1} = L_i \mid Z^{k-1})$. However, when the robot rotates, the scene of sequential input image changes drastically, and the *a posteriori* probability of the previous step becomes meaningless. Therefore, when the robot rotates, *a priori* probability is assumed to uniform distribution. The odometry information is used to decide whether the robot is rotated more than 10° from the previous step. Consequently, the probability that the current place L_k is the same as some previously seen place L_i is calculated as follows:

$$P(L_k = L_i \mid Z^k) \simeq \begin{cases} \dfrac{1}{k-1} \cdot \dfrac{P(d_{ki} \mid L_k = L_i)}{P(d_{ki})} & \textit{if robot is rotated from previous step} \\ \dfrac{P(d_{ki} \mid L_k = L_i) \cdot P(L_{k-1} = L_i \mid Z^{k-1})}{P(d_{ki})} & \textit{else} \end{cases} \tag{3.24}$$

After normalizing the probability distribution, if the maximum of $P(L_k = L_i \mid Z^k)$ for previous places exceed a threshold, the current place and the corresponding previous place are regarded as the same place; otherwise, the current place is regarded as a new place and added to the map database.

3.7.4 Global Pose Correction

To close the loop, a pose graph optimization is performed after the loop detection. Before the global pose optimization, the relative position is computed between the matched frame from the loop detection and the current frame using the method proposed in Section III-D. The pose graph optimization is conducted for all poses between the matched frame from the loop detection and the current frame. The cost function to be minimized for pose graph optimization is as follows:

$$\begin{aligned} E(\mathbf{x}^w_{matched}, \ldots, \mathbf{x}^w_{curr}) = & \sum_{i=matched}^{curr} \| (\mathbf{z}_{vo,i} - (\mathbf{x}_i \ominus \mathbf{x}_{i-1})) \|^2_{\Sigma_{VO}} \\ & + \| (\mathbf{z}_{matched,curr} - (\mathbf{x}_{matched} \ominus \mathbf{x}_{curr})) \|^2_{\Sigma_{LD}} \end{aligned}, \tag{3.25}$$

where $\mathbf{x}_{curr}$ is the pose of the current frame; $\mathbf{x}_{matched}$ is the pose of the matched frame from the loop detection; $\mathbf{z}_{incre,i}$ is the relative incremental pose of $\mathbf{x}_i$ with respect to the $\mathbf{x}_{i-1}$ frame from the current estimates;

$\mathbf{z}_{matched,curr}$ is the relative pose of $\mathbf{x}_{curr}$ with respect to $\mathbf{x}_{matched}$; Σ_{incre} is the covariance matrix of the incremental pose estimation; and Σ_{LD} is the covariance matrix of the relative pose from the loop detection. The covariance matrices are determined proportional to the inverse of the co-visibility of line landmarks. In case there is low co-visibility between incremental poses because of rotation or a low-textured area, we set the upper bound to determine the covariance matrix. The optimization is performed by Levenberg-Marquart algorithm using the g2o library [16]. After the optimization, each map line is transformed according to the correction of each corresponding frame that observes it.

4. Experiments

In the experiments, datasets-based experiments on a desktop computer and real-time experiments on an embedded system are performed. For various experiments, home datasets and Vicon datasets are captured. In the Vicon datasets, Vicon motion capture system is used for ground truth robot trajectory in an indoor environment. As a large scale indoor test, experiments are also conducted using the public RAWSEEDS benchmark dataset [23]. For real-time embedded experiments, the proposed algorithm is implemented in NXP4330Q embedded board, and conducted experiments in a home environment. In our comparative experiments, three different approaches are compared as given below.

First, a 2D version of ORB-SLAM [25] method is implemented. There are several differences between the original ORB-SLAM and the implemented 2D version of ORB-SLAM. First, the camera poses are parameterized in the 2D space. Second, the odometry data is used to calculate the initial pose of the robot in the tracking thread. Third, the odometry data are used as an edge in the local bundle adjustment step. The odometry-based edge, which is called *EdgeSE2* in the g2o framework [16] is generated between consecutive camera poses in the *SE(2)* space. Fourth, the relocalization mode in the tracking thread is eliminated by means of the odometry data. Originally, the relocalization mode is conducted when the

tracked features are lost. Especially, when the robot rotates rapidly or the robot travels in a low-textured environment, the tracked features are easily lost; this is followed by the relocalization mode. When the relocalization mode is activated, the SLAM process is stopped. However, with the aid of odometry, we can proceed with the SLAM process. This algorithm is denoted as *ORB-SLAM 2D* in the following experiments.

Second, VP-based line SLAM with standard bundle adjustment method has been implemented. The overall algorithm is very similar to the proposed method. The line extraction, VP extraction, data association, line parameterization, line observation model, and line initialization methods are the same as those in the proposed method. In the tracking thread, the VP-based robot orientation estimation procedure is skipped. In the mapping thread, the standard local bundle adjustment is conducted. The cost function to be minimized is formulated as follows:

$$E(\mathbf{x}_{c,s},\cdots,\mathbf{x}_{c,k},\mathbf{l}_l,\mathbf{l}_{l+1},\cdots) = \sum_i \| (\mathbf{z}_{o,i} - (\mathbf{x}_i \ominus \mathbf{x}_{i-1})) \|^2_{\Sigma_{odo}} + \sum_i \sum_j \rho(\left\| \mathbf{z}_{i,j} - h(\mathbf{x}_i, \mathbf{l}_j) \right\|^2_{\Sigma_{line}}) \quad (3.26)$$

where $\mathbf{l}_j$ is the position of the landmark, which is one of the lines (i.e., the vertical line, the *x-axis* horizontal line, or the *y-axis* horizontal line); $\mathbf{z}_{i,j}$ is the measurement of line landmark in the image plane; and $h(\cdot)$ is the line projection model. In the loop-closing thread, the robot's pose for the current frame is estimated using the matched line landmarks from the matched

frame by minimizing the cost function

$$E(\mathbf{x}_{current}) = \sum_{j} \left\| \mathbf{z}_j - h(\mathbf{x}_{current}, \mathbf{l}_{j,matched}) \right\|^2_{\Sigma_{line}}, \qquad (3.27)$$

where $\mathbf{l}_{j,matched}$ is the position of line landmark where it has been matched in the current frame through data association with the matched frame from the loop detection. This version of implementation is intended to examine the effect of the proposed estimation method for VP-based robot orientation and the local map correction method compared with the standard optimization-based method. This algorithm was denoted as *VP standard BA* in the following experiments.

Third, the VP-based orientation correction method without line landmark estimation is implemented. This version estimates VP-based orientation only in the tracking thread and motion-only bundle adjustment 1 in the mapping thread with no loop-closing thread. This version is intended to examine the performance of the proposed VP-based orientation estimation method only. This method is denoted as *VP-only* in the following experiments.

Apart from the SLAM experiments, loop detection performances of various methods including the proposed method (Chapter 3.7) are presented in Appendix. The experiments are conducted in home environment.

4.1 Home Environment Dataset

Home datasets are acquired in a typical home environment as shown in

Fig. 4.1. The robot explored the experimental environment while collecting images, with a resolution of 320 × 240 pixels, from a forward-viewing mono-camera, along with robot odometry data. Acquiring the datasets are performed four times with two different start positions for the robot. The characteristics of each home datasets are summarized in Table 4.1. In datasets 3 and 4, the illumination conditions are changed ten times during the experiments by turning the lights on or off. The intervals between the changes in illumination condition are above three hundred frames. In addition, 20% of the input images contain moving people or objects. The sample images of the dynamic environment situations are illustrated in Fig. 1.3 (d)-(i). Two different start positions are illustrated in Fig. 4.2; This figure also shows the furniture disposition of the environment. The robot has moved in a trajectory greater than 400 m, and the total numbers of collected images are 8393, 8780, 7734, and 8431 respectively, for datasets 1, 2, 3, and 4.

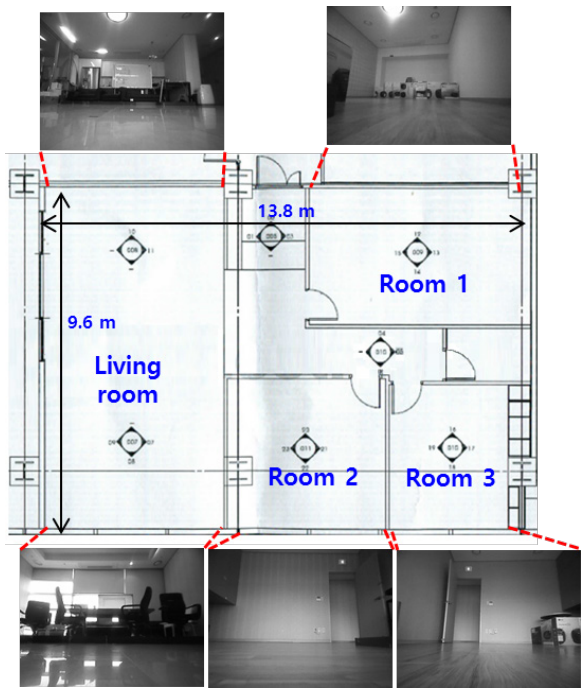


Figure 4.1 Blueprint of home environment and example images

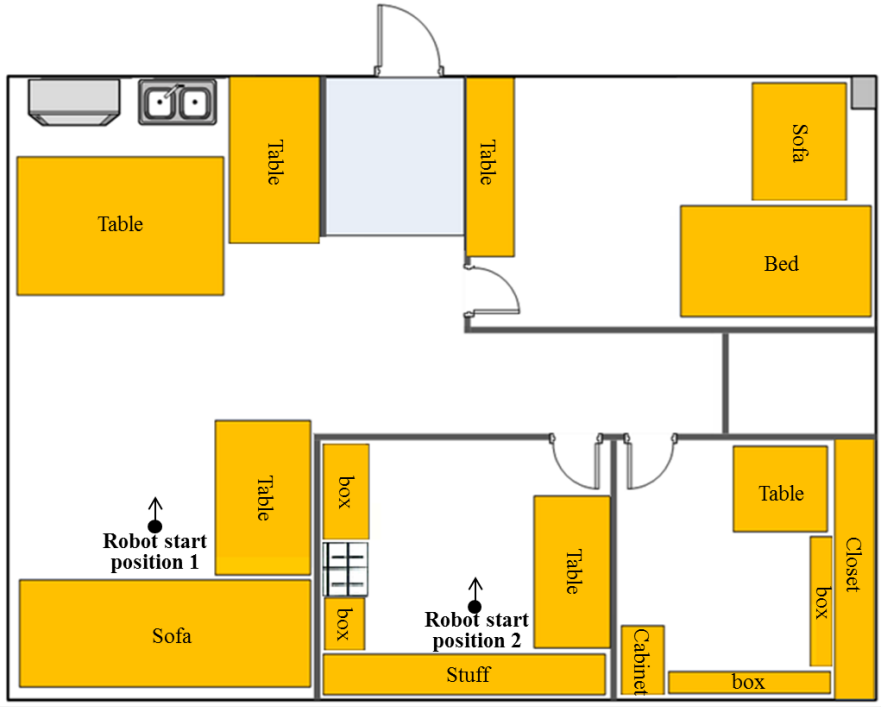


Figure 4.2 Furniture disposition of the home environment

TABLE 4.1
HOME DATASET CHARACTERISTICS

Dataset	Start position	Environment	# of images	# of images containing moving people or objects
1	1	Static	8393	None
2	2	Static	8780	None
3	1	Dynamic	7734	1859
4	2	Dynamic	8431	1466

For robotic platform, the robotic vacuum shown in Fig. 4.3 is used. An upward-viewing mono-camera is used for autonomous navigation [60] while acquiring the datasets. As a path planning strategy, Boustrophedon method [79] is used to cover the whole environment, and ultrasonic sensors are used to avoid obstacles. At the end of the drive, the robot is manually returned to the starting point using the remote controller. Ideally, the estimated final pose of the robot must be the origin, that is, $(0,\ 0,\ 0°)^T$. The distance between the estimated final position and the origin is used to quantitatively measure the accuracy of the SLAM algorithm, and called this the closed-loop error.

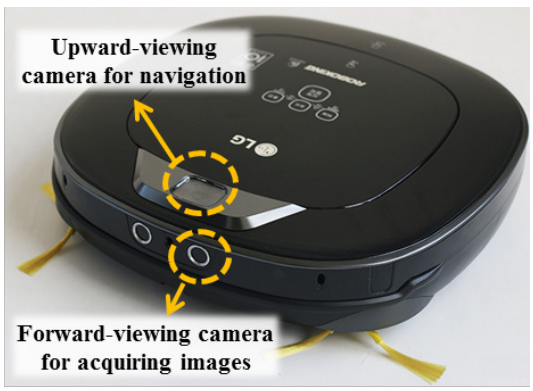


Figure 4.3 Robot platform for acquiring home dataset

The algorithms are tested on a desktop computer with Intel Core i7-2600. Fig. 4.4 shows the result of the proposed SLAM from the home dataset 3 (dynamic environment). The purple lines represent the reconstructed line landmarks, and the yellow line represents the SLAM-based robot trajectory. Fig. 4.5 and Fig. 4.6 shows the estimated robot trajectories of various methods for home dataset 1 (static environment) and 3 (dynamic environment). As the robot is manually returned to the starting point at the end of the drive, the estimated end position should ideally be the same as the start position. For the *ORB-SLAM 2D* case, it is clear that a large error drift has occurred. Estimated end position of the robot is far from the start position which should have been the same as the start position. This is mainly due to error drift in the low-texture areas of the environment and difficulty in feature tracking when robot moves every inch of the environment. Furthermore, no large loop closure between early stage and last stage has occurred. Loop closure occurred only twice between the 1035 frame and the 1206 frame, and between the 3060 frame and the 6320 frame. Clearly, the orientation estimation of the proposed method is more accurate than that of the *VP Standard BA* case. The difference lies in the fact that the robot's orientation is corrected directly from the VP in the proposed method. The robot's orientation is corrected by extracting the landmarks in the *VP Standard BA* method. Although the orientation error in the line landmark estimation can be eliminated from the VP, the accuracy of the *VP Standard*

BA method reduces when the extraction of the line landmark is limited in low-textured areas. In case of *VP-only*, the orientation error is effectively eliminated, but the integrated translation error exists.

For a quantitative analysis, the closed-loop error is measured, as defined in the previous paragraph. Table 4.2 shows the measured closed-loop error of the various methods for the four home datasets. The proposed method shows the lowest closed-loop error compared with the other methods. The accuracy of the VP-based methods is similar in both the static environment and dynamic environments.

With respect to the computational speed, the average running time for processing a single frame of each thread is measured and summarized in Table III. The difference between the proposed method and *VP Standard BA* is mainly in the mapping thread. The proposed local map correction method is 3.6 times faster than the standard bundle adjustment while showing better accuracy. The *ORB-SLAM 2D* shows the slowest result. Originally, ORB-SLAM runs usually around 30 Hz for 640 × 480 resolution images. But in the experiments, 320 × 240 resolution images with low-textured areas are used, and camera poses are parameterized in 2D space. This resulted in relatively faster results than the original ORB-SLAM. After finishing the SLAM, total memory usages are measured. The average memory usages for four datasets are summarized in Table 4.4 for four methods. The memory usage for the *VP-only* method is the lowest because it does not estimate the landmark. The proposed method and the *VP Standard BA* method required

an average of 114.1 MB and 113.9 MB, respectively, for conducting the whole SLAM process, whereas the *ORB-SLAM 2D* method required an average of 1160.5 MB. At the start of the SLAM process, *ORB-SLAM 2D* requires more than 250 MB to load the vocabulary tree. In the experiments, the *ORB-SLAM 2D* method have reconstructed an average of 25,892 point features, whereas the proposed method and the *VP Standard BA* method reconstructed an average of 1,072 and 1,058 line features, respectively.

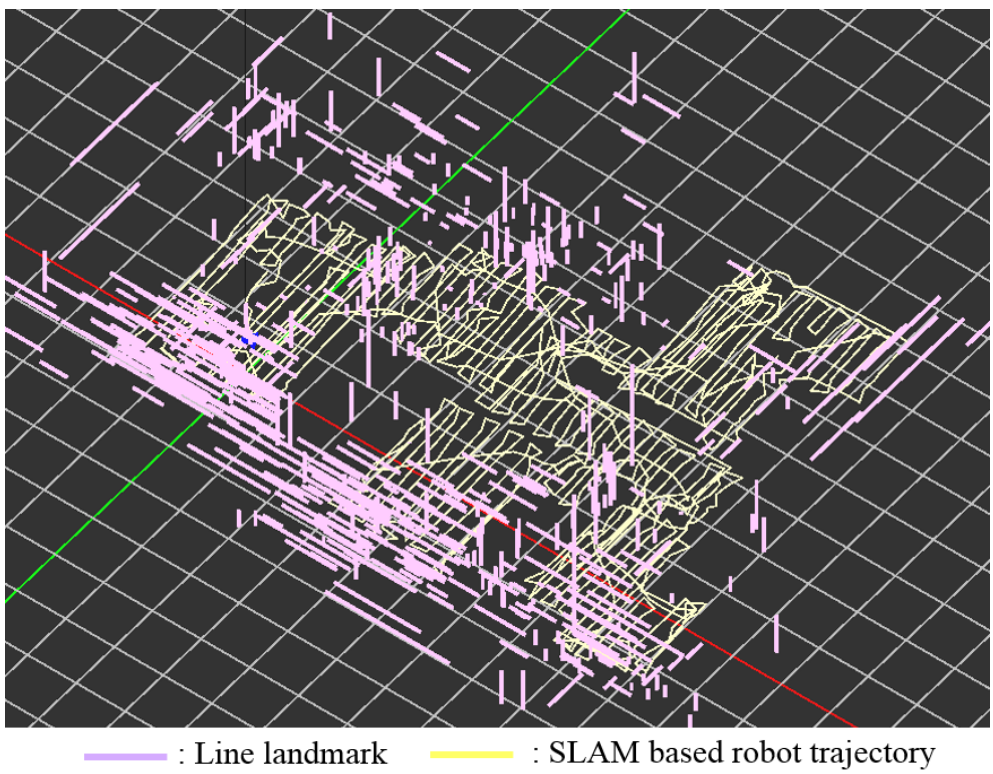


Figure 4.4 Result of the proposed SLAM using home dataset 3 (dynamic). Purple lines represent reconstructed line landmarks, and yellow line represents the SLAM based robot trajectory.

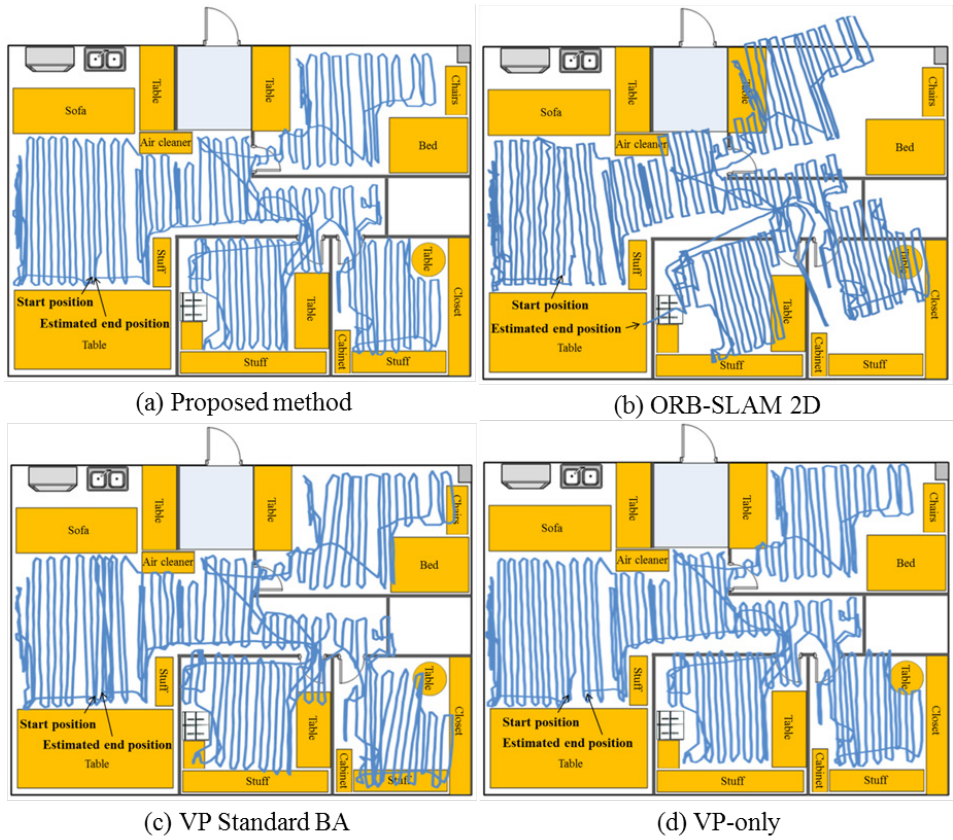


Figure 4.5 Estimated robot trajectories of various methods using the home dataset 1 (static environment).

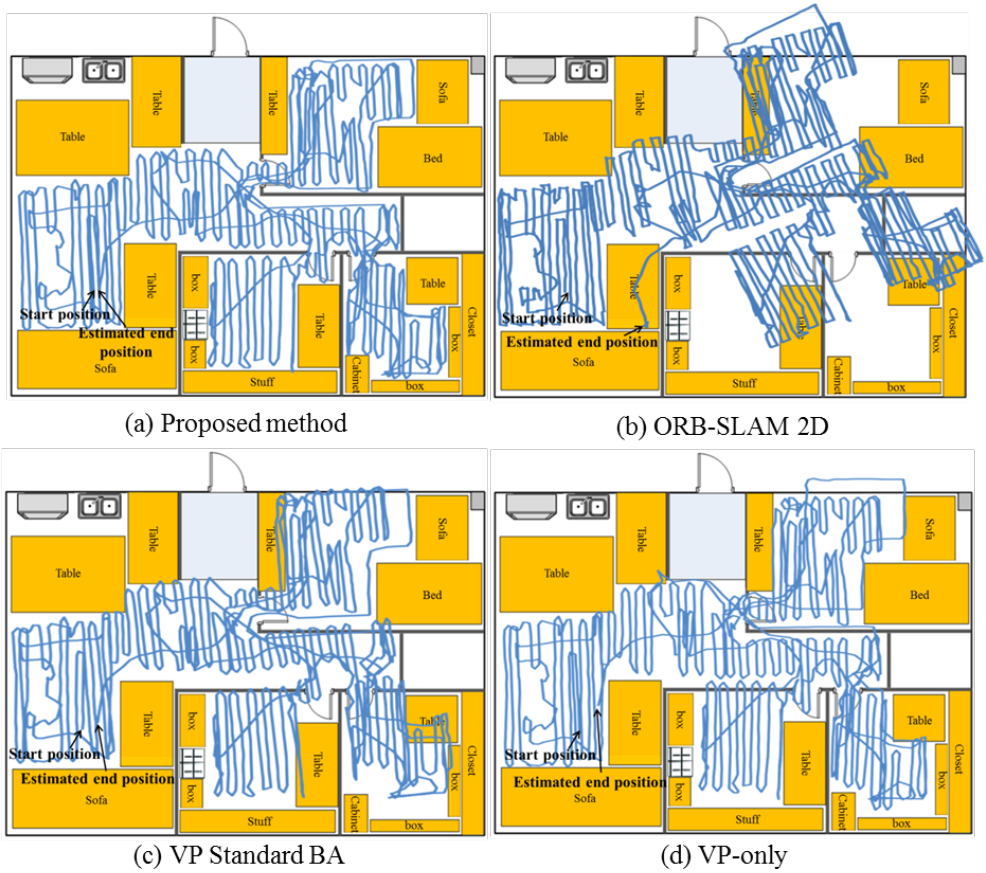


Figure 4.6 Estimated robot trajectories of various methods using the home dataset 3 (dynamic environment).

TABLE 4.2
CLOSED-LOOP ERROR OF VARIOUS METHODS IN HOME DATASET

Dataset	Proposed method (cm)	ORB-SLAM 2D (cm)	VP Standard BA (cm)	VP-only (cm)
1	6.6	147.0	52.0	93.5
2	7.0	67.6	42.2	129.8
3	13.0	226.1	59.0	106.2
4	6.1	98.4	47.4	91.3
Avg.	8.2	134.8	50.2	105.2
Std.	3.2	69.1	7.1	17.7

TABLE 4.3
TIMING RESULTS OF VARIOUS METHODS PER EACH THREADS IN HOME DATASET

Version	Thread	Avg. (ms)	Std. (ms)
Proposed Method	Trackng	6.96	5.03
	Mapping	5.71	4.19
	Loop Closing	4.72	1.38
ORB-SLAM 2D	Trackng	16.41	7.67
	Local Mapping	41.92	34.22
	Loop Closing	3.08	26.37
VP Standard BA	Trackng	6.92	5.01
	Mapping	20.31	11.20
	Loop Closing	4.81	1.72
VP-only	Trackng	3.02	0.96
	Mapping	0.3	0.14

TABLE 4.4
AVERAGE MEMORY USAGE OF VARIOUS METHODS IN HOME DATASET

Proposed method	ORB-SLAM 2D	VP Standard BA	VP-only
114.1 MB	1160.5 MB	113.9 MB	2.9 MB

4.2 Vicon Dataset

For further evaluation, four indoor environment datasets are made with Vicon motion capture system. The motion capture system tracks the position of infrared reflective markers with high accuracy, that is, with less than 0.5

mm error. The Vicon motion capture system is illustrated in Fig. 4.7. Blue violet-colored acrylic partitions are placed for robot to drive within the recognizable area of motion capture system. Total eight Vantage V5 motion capture cameras are used in the experiments to generate ground truth trajectory of the robot. Vantage V5 motion capture camera which is shown in Fig. 4.8 emits infrared (IR) lights and receives the reflected IR lights from the markers. Fig. 4.9 illustrates the example of motion capture from Vicon tracker program. Fig. 4.10 shows the robot platform with markers for the experiment.

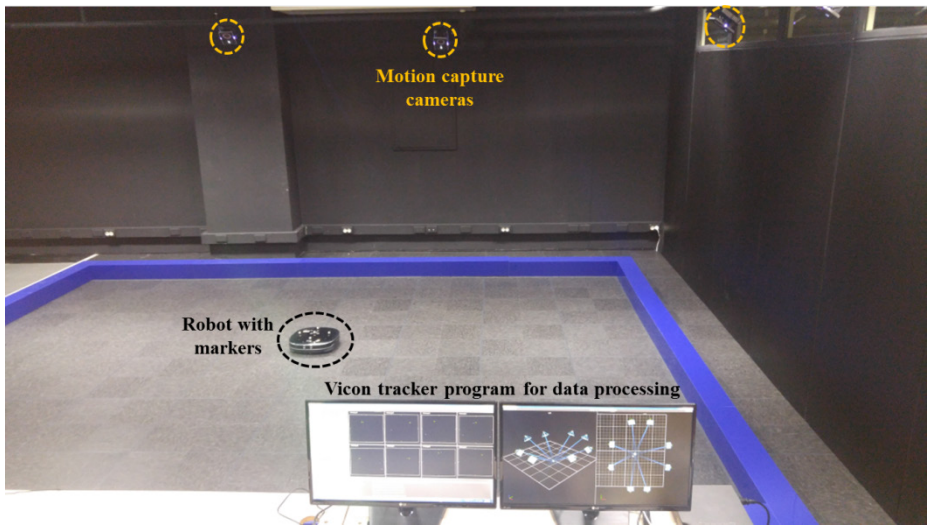


Figure 4.7 The experimental environment for Vicon dataset

Figure 4.8 Vantage V5 motion capture camera

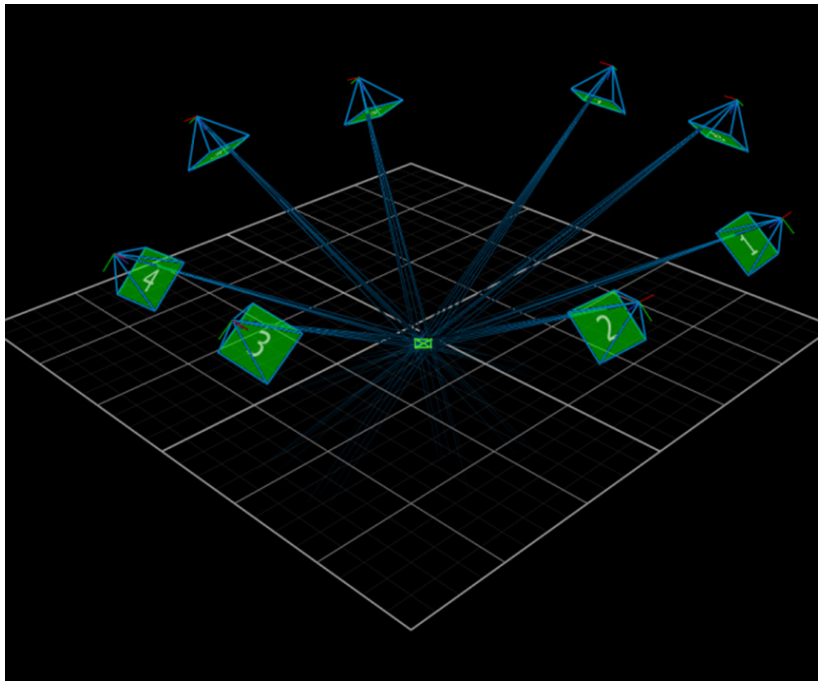


Figure 4.9 The example of motion capture from Vicon tracker program

Figure 4.10 Robot platform with markers for motion capture

Total four datasets are generated which are called Vicon dataset 1 to 4 in the latter part of this dissertation. The experimental environment of Vicon datasets are illustrated in Fig. 4.11 and Fig. 4.12. In the Vicon dataset 1 and 2, the images contain no moving people, while more than 30% of input images contain moving people in dataset 3 and 4. The procedure of generating the dataset is same as that of home environment dataset. At each datasets, the robot has moved in a greater than 85 m trajectory, and the total numbers of collected images are 1641, 1684, 1708, and 1655 respectively, for the datasets 1, 2, 3, and 4, respectively. These datasets are quite challenging because of the lack of features in most areas. The sample images of Vicon dataset are illustrated in Fig. 4.13. When corner features are detected for whole datasets using the FAST algorithm [24] with threshold 20, it only contains average of 136 corners per image. This value is even lower than that of the home environment dataset.

The resultant robot trajectories are illustrated in Fig. 4.14-4.17 for Vicon dataset 1 to 4, respectively, aligned with the ground truth trajectory from motion capture system. Since the trajectories for both dataset are not long, large errors have not occurred. The absolute position errors for whole robot poses are compared in Table 4.5. The error of the proposed method is lower than other methods in all cases. The errors for the whole trajectories for four datasets are illustrated in Fig. 4.18-4.21.

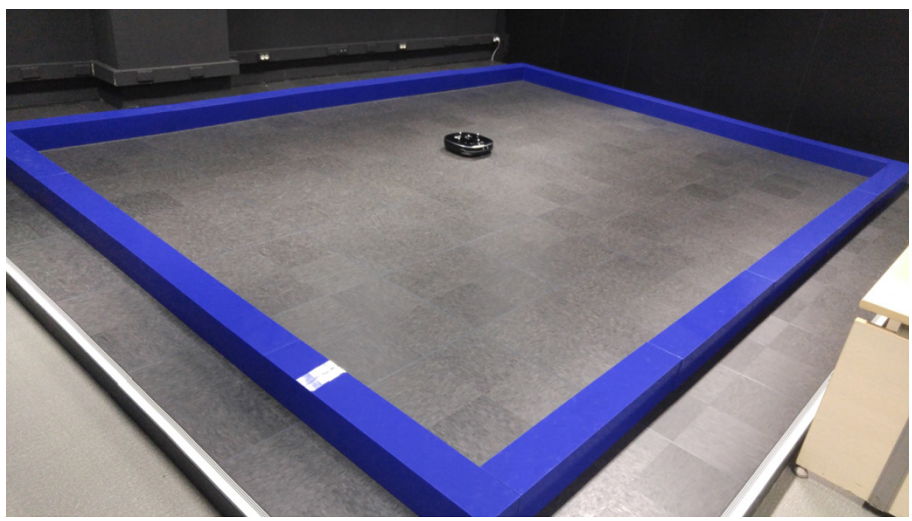


Figure 4.11 Experimental environments for Vicon dataset1 (static) and 3 (dynamic)

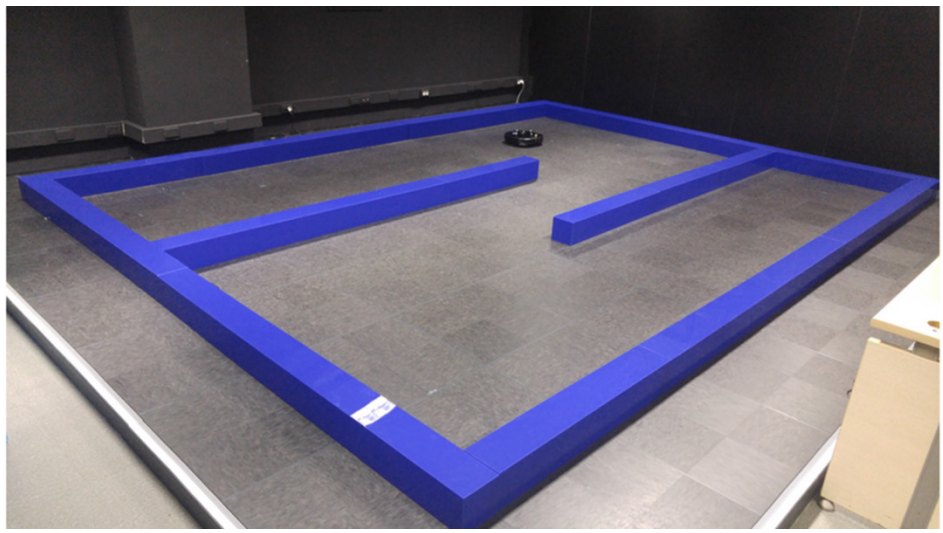


Figure 4.12 Experimental environments for Vicon dataset2 (static) and 4 (dynamic)

Figure 4.13 Sample images of Vicon dataset

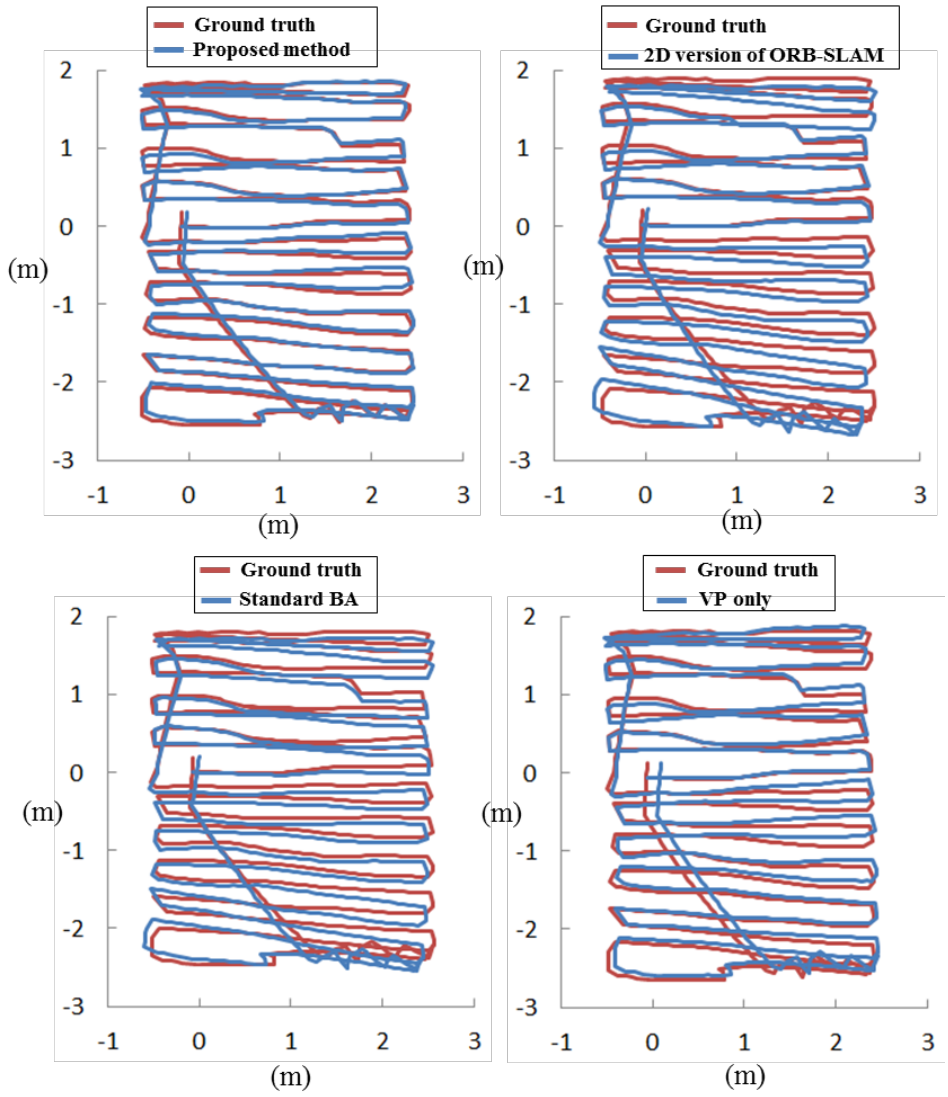


Figure 4.14 Estimated robot trajectory of various methods for Vicon dataset 1 (static) aligned with the ground truth trajectory from Vicon motion capture system.

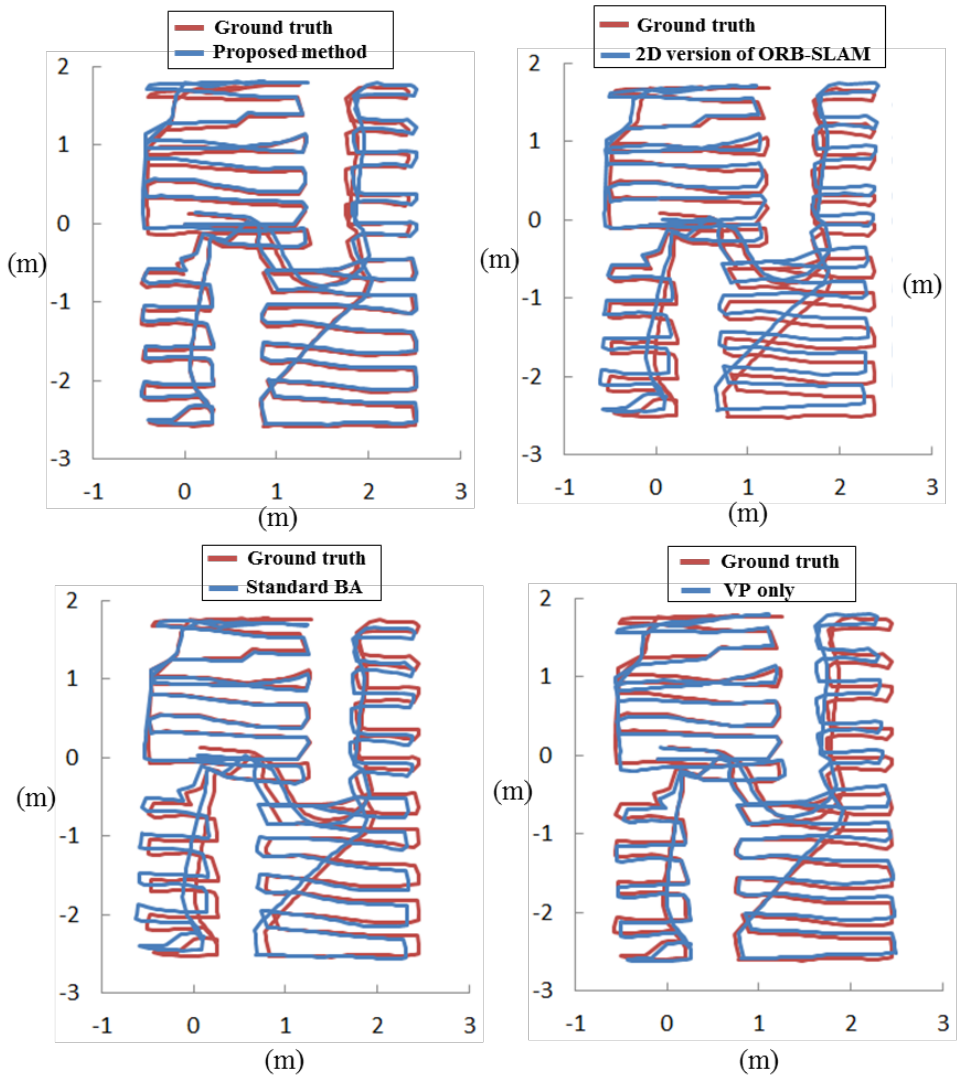


Figure 4.15 Estimated robot trajectory of various methods for Vicon dataset 2 (static) aligned with the ground truth trajectory from Vicon motion capture system

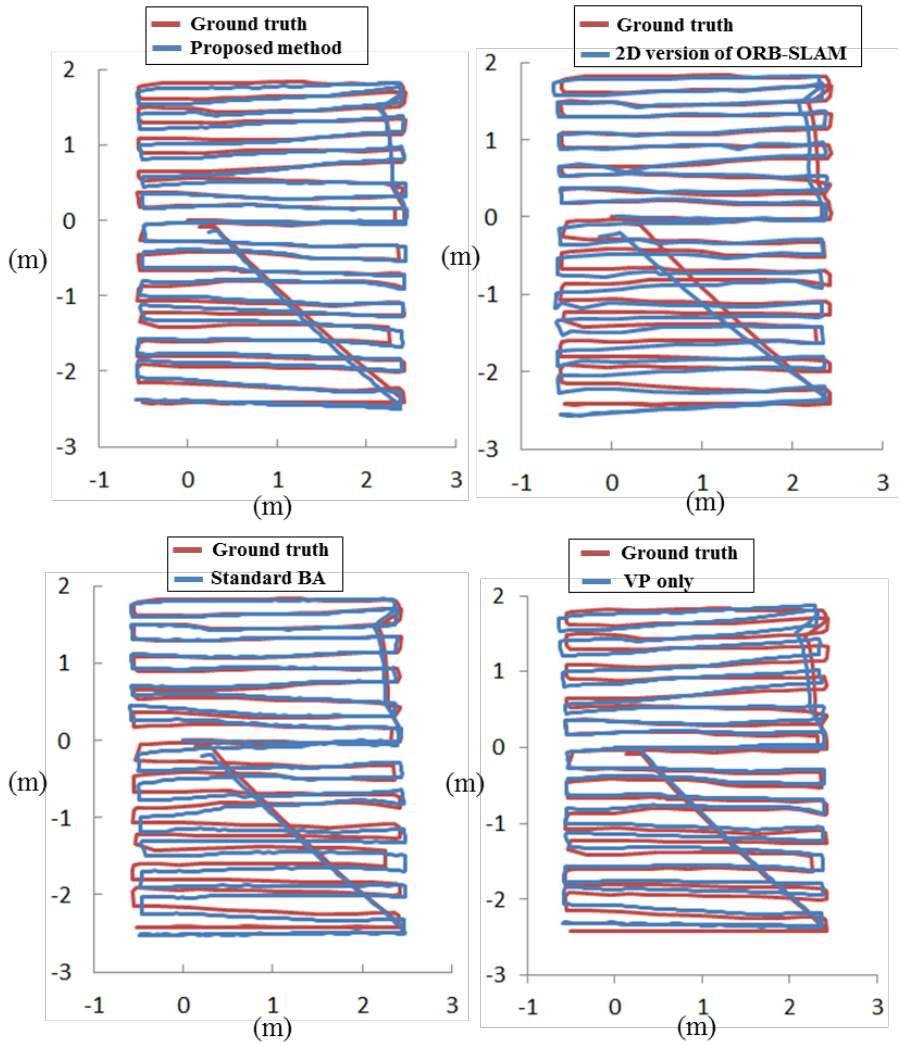


Figure 4.16 Estimated robot trajectory of various methods for Vicon dataset 3 (dynamic) aligned with the ground truth trajectory from Vicon motion capture system.

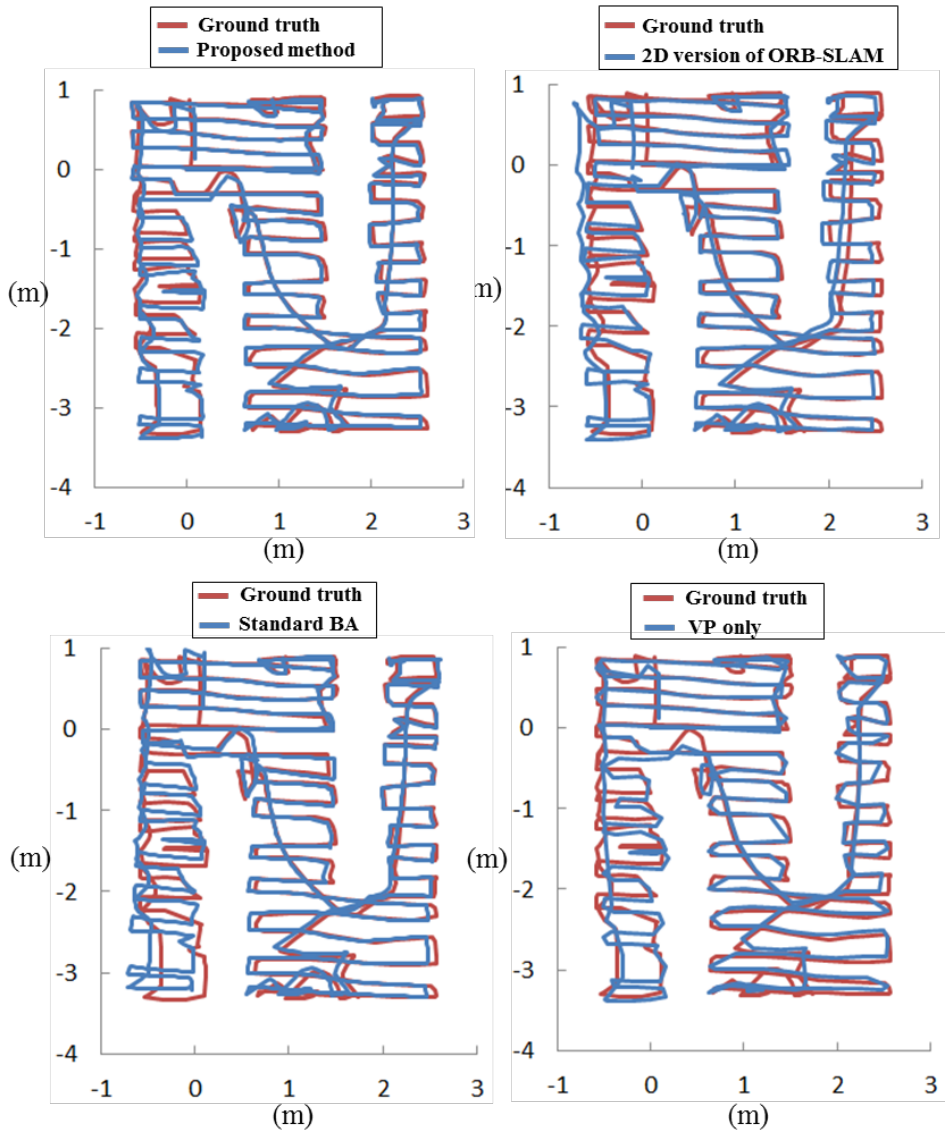


Figure 4.17 Estimated robot trajectory of various methods for Vicon dataset 4 (dynamic) aligned with the ground truth trajectory from Vicon motion capture system

TABLE 4.5
ABSOLUTE POSITION ERROR OF VARIOUS METHODS IN VICON DATASET

Dataset	Error	Proposed method (cm)	ORB-SLAM 2D (cm)	VP Standard BA (cm)	VP-only (cm)
1 (static)	Avg.	4.7	8.0	7.2	6.9
	Max.	15.9	30.4	17.9	20.9
	Std.	2.8	4.4	4.1	2.4
2 (static)	Avg.	4.6	10.0	7.1	4.7
	Max.	12.2	38.1	20.8	11.6
	Std.	3.4	6.6	4.9	2.9
3 (dynamic)	Avg.	3.7	9.2	6.5	5.5
	Max.	8.9	28.2	13.5	11.2
	Std.	1.7	5.1	2.6	2.6
4 (dynamic)	Avg.	5.5	12.1	8.7	6.1
	Max.	10.2	40.4	26.7	10.7
	Std.	2.4	6.5	4.9	1.6

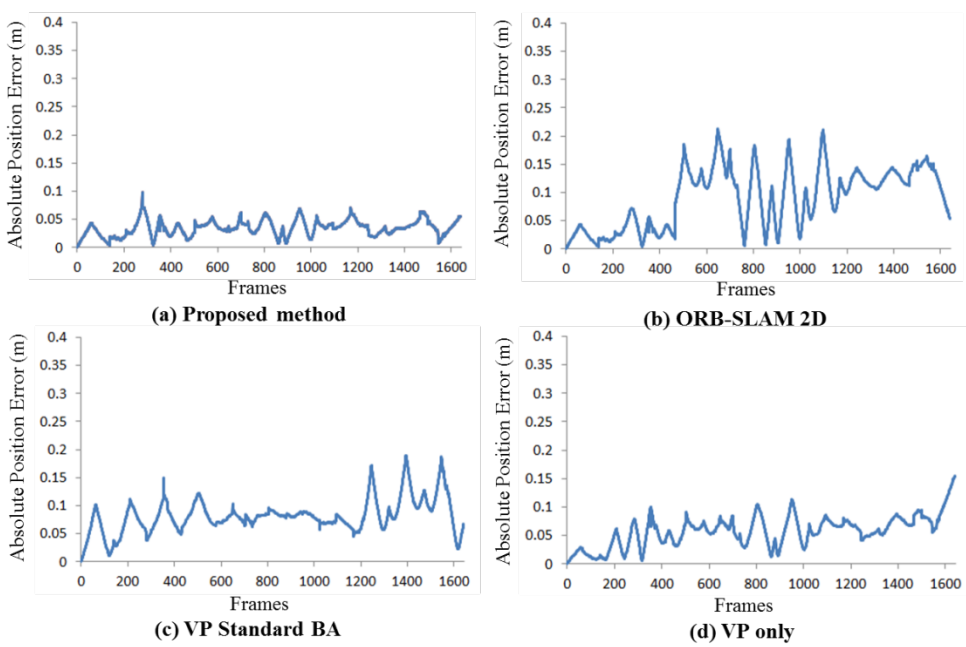


Figure 4.18 Absolute position errors for the whole trajectories for Vicon

dataset 1 (static)

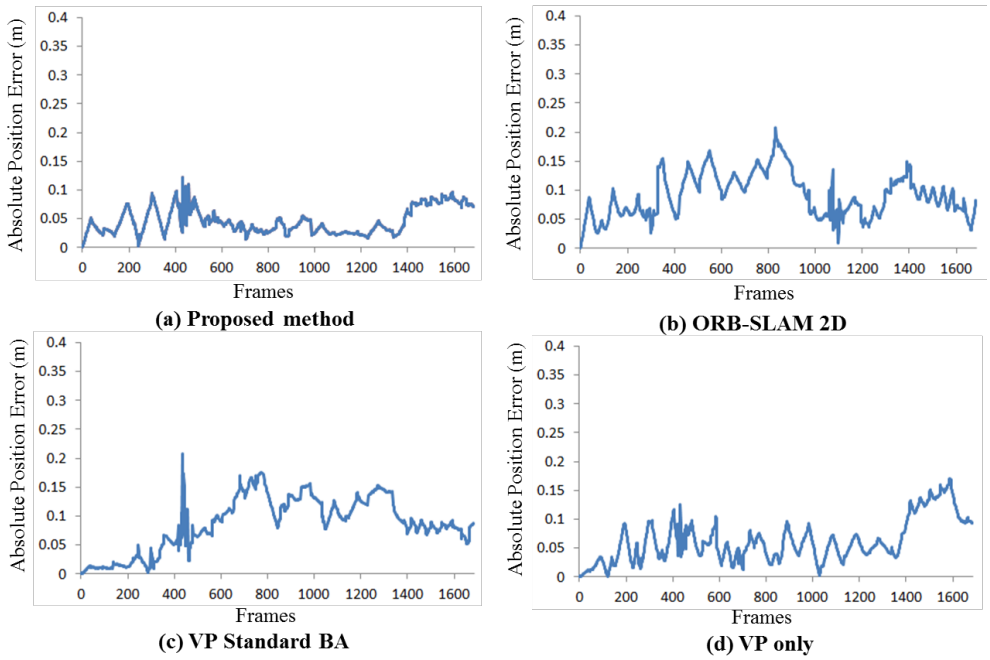


Figure 4.19 Absolute position errors for the whole trajectories for Vicon dataset 2 (static)

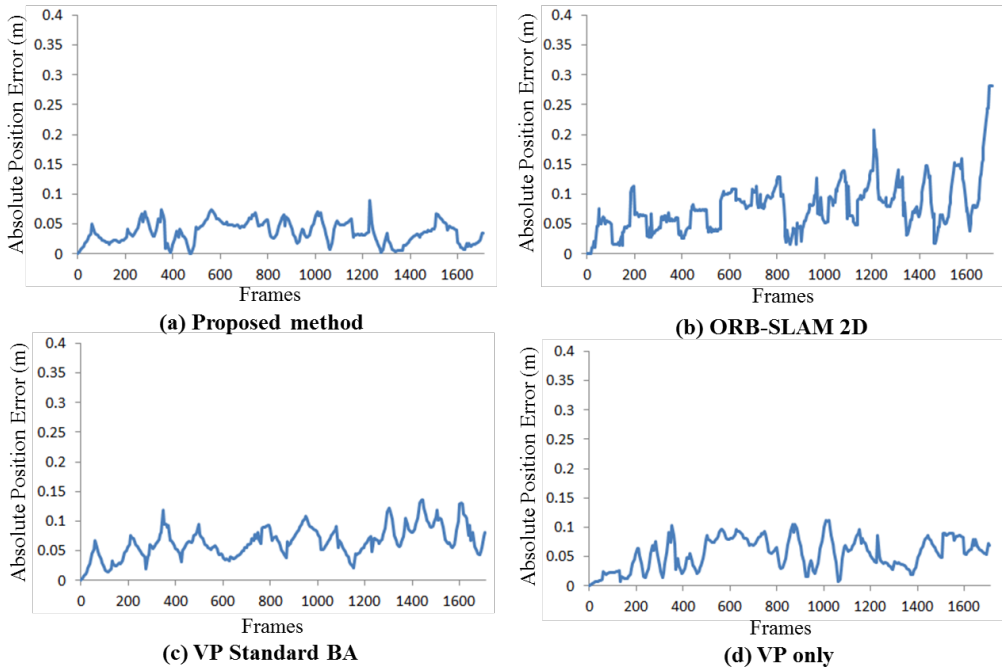


Figure 4.20 Absolute position errors for the whole trajectories for Vicon dataset 3 (dynamic)

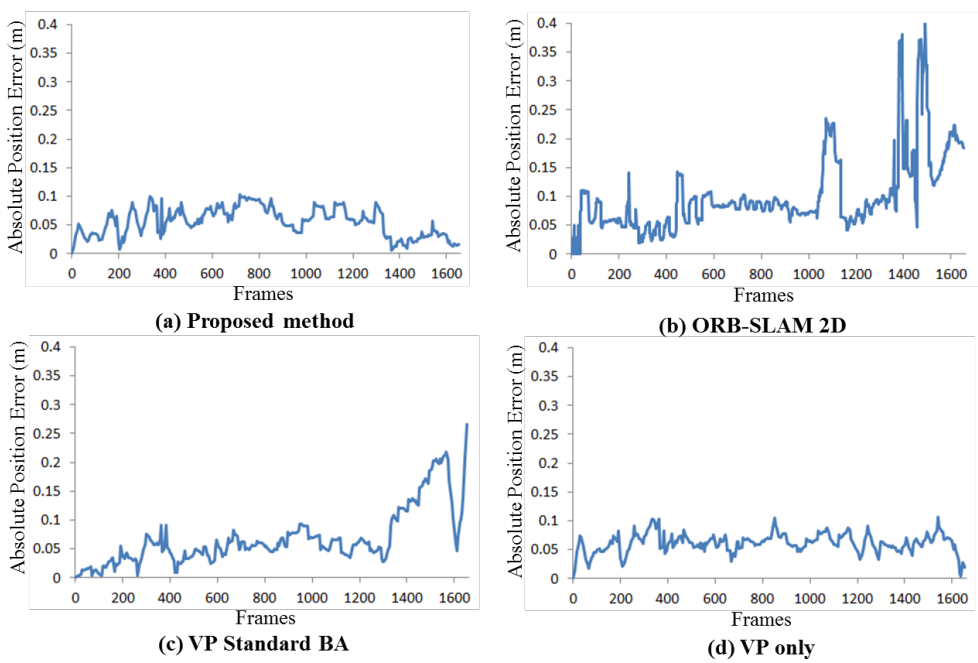


Figure 4.21 Absolute position errors for the whole trajectories for Vicon dataset 4 (dynamic)

For further evaluation, mapping performance of the proposed SLAM is tested. The real coordinates of the map lines is measured by hand with respect to the world frame (same as the first robot pose). After the SLAM using Vicon dataset 3, fifteen map lines are randomly sampled that are not eliminated during the SLAM process. Figure 4.22 shows these map lines marked as blue dotted ellipse among the extracted lines in the images. To measure the estimation errors in y-axis horizontal lines, differences of x and z coordinates between the measured coordinates and the estimated coordinates from the SLAM are used. Likewise, the estimation errors in x-axis horizontal lines and vertical lines are measured using y and z coordinates and x and y coordinates, respectively. The mapping errors are calculated for each axis. Table 4.6 shows the results. The average of mapping errors for x-axis, y-axis and z-axis coordinates are -0.14 m, 0.09 m and 0.08 m, respectively. Although the standard deviation of the mapping errors (0.40 m, 0.14 m, 0.26 m for x-axis, y-axis and z-axis coordinates) are relatively large, the proposed method also utilizes odometry data and loop closure technique, which results in accurate robot localization performances (under 5 cm of average absolute position errors for Vicon dataset).

Figure 4.22 Extracted map lines for measuring the mapping error (marked as blue dotted ellipse)

TABLE 4.6
ESTIMATION ERRORS IN MAP LINES FOR EACH AXIS

	Error in *x*-axis	Error in *y*-axis	Error in *z*-axis
Avg. (m)	-0.14	0.09	-0.08
Std. (m)	0.40	0.14	0.26

4.3 Benchmark dataset in large scale indoor environment

To evaluate the accuracy in large scale indoor environments, experiments are conducted using the RAWSEEDS benchmark dataset (Biccoca25b) [23].

The dataset is collected in an office building by a wheeled robot with multiple sensors including laser range finders, cameras, inertial measurement unit, and wheel encoders. The dataset also provide ground truth data of the robot's pose for evaluation. In this experiment, the image sequences from the frontal camera and odometry data are used. The sample images from the frontal camera are illustrated in Fig. 4.23. The dataset consists of 52,695 images, and the whole path is about 774 m long. Even though this dataset contains low-textured areas and dark corridors in some area, it has sufficient features in comparison with the home dataset or Vicon dataset. In addition, the path contains several loops and revisited regions. This implies that large error drift can be reduced by applying loop detection and loop closing algorithms.

The resultant robot trajectories are illustrated in Fig. 4.24; the trajectories are aligned with the ground truth trajectory. Absolute position errors for the whole robot trajectories are compared in Table 4.7. The average of the absolute position error for whole trajectory of the proposed method, ORB-SLAM 2D, VP Standard BA, and VP-only are 0.71, 2.51, 0.88, and 2.43 m, respectively. Interestingly, the VP-only method is quite accurate with no effort of executing complicated landmark estimation procedures. The errors for the whole trajectories for various methods are illustrated in Fig. 4.25.

Figure 4.23 Sample images of RAWSEEDS (Bicocca25b) dataset from the frontal camera

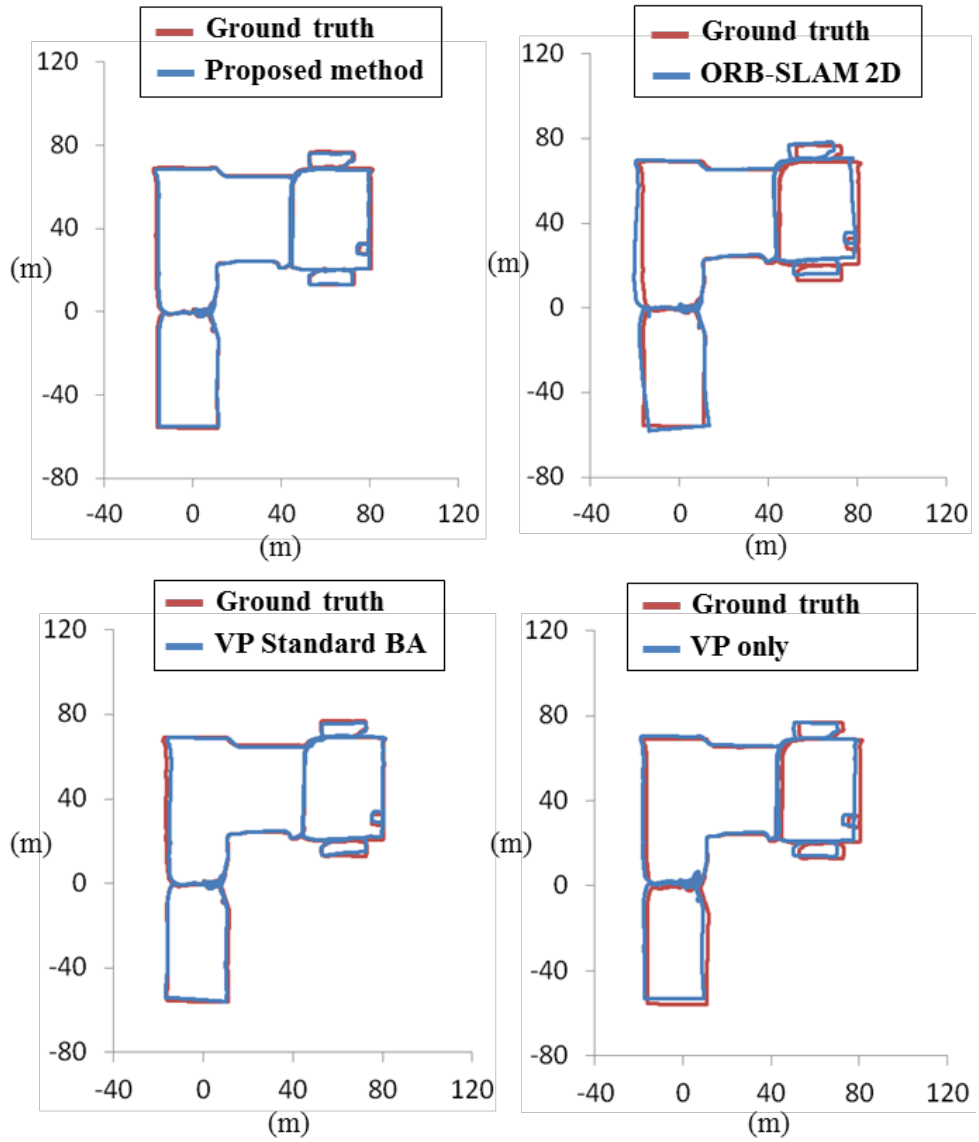


Figure 4.24 Estimated robot trajectories of various methods for RAWSEEDS (Bicocca 25b) dataset aligned with the ground truth trajectory

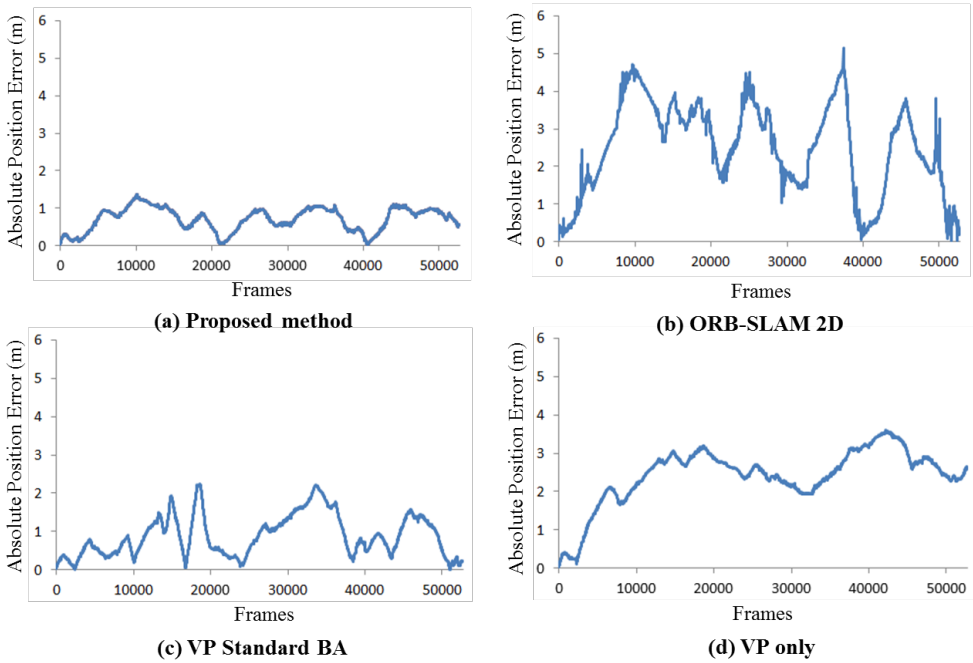


Figure 4.25 Absolute position errors for the whole trajectories for RAWSEEDS (Bicocca 25b) dataset.

TABLE 4.7
ABSOLUTE POSITION ERROR OF VARIOUS METHODS IN RAWSEEDS DATASET (BICOCCA 25B)

	Proposed method	ORB-SLAM 2D	VP Standard BA	VP-only
Avg. (m)	0.71	2.51	0.88	2.43
Max. (m)	1.36	5.14	2.24	3.61
Std. (m)	0.31	1.21	0.54	0.73

4.4 Embedded Real-Time SLAM in Home Environment

The proposed SLAM algorithm has been implemented in a low-cost embedded board of NXP4330Q [38] which is equipped with Cortex A9 processor and 512 MB memory. Fig. 4.26 illustrates the robot platform equipped with NXP4330Q board. The proposed method is integrated in an autonomous robot navigation system. When robot explores the environment, the obstacle grid map is constructed by using the localization result from the proposed SLAM and detected obstacles from the ultrasonic sensor. This obstacle grid map is used in path planning and motion planning for autonomous robot navigation. The Boustrophedon method [79] is used for path planning strategy. All services including data acquisition, the proposed SLAM, navigation, and motion planning are simultaneously executed in NXP4330Q board in real time. Considering the limited computational resources, images are captured when robot is moved more than 30 cm, or rotated more than 30° compared to the previous frame. The driving and rotation velocities of the mobile robot are 0.35 m/s and 30 °/s, respectively.

At the end of the drive, the robot is manually returned to the starting point with the remote controller for measure the closed-loop error. As the same as home dataset based experiments in previous section, real time SLAM experiments are performed four times. The scenarios of the experiments are

the same as acquiring the home datasets.

Figure 4.27 illustrates the generated obstacle grid map in the real-time experiment sequence 3 (dynamic environment). The yellow grid indicates the area where the robot has driven. The blue and pink grid indicates the detected wall and obstacle from the ultrasonic sensor. Comparing with the blueprint of the environment in Fig. 4.2, the map is accurately built using the proposed method. Table 4.8 shows the measured closed-loop error of the real time SLAM experiments for the four sequences. The average and standard deviation of closed-loop errors are 8.7 cm and 1.8 cm, respectively. The accuracy is similar to the dataset-based experimental results where the algorithm is executed in desktop PC. For the computation time, the average time for embedded processor to process one frame for tracking, mapping, and loop closing thread are 243.2, 211.8, and 138.1 ms, respectively.

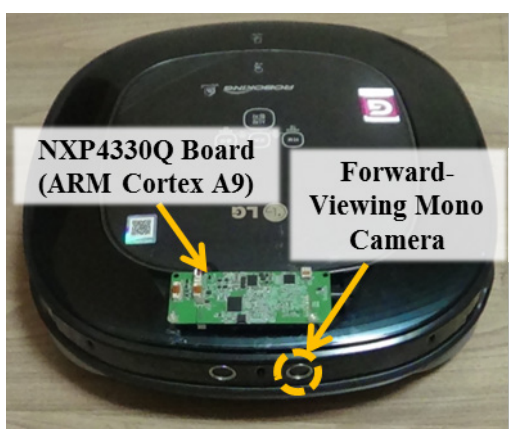


Figure 4.26 Robot platform equipped with NXP4330Q board for real time SLAM experiment

TABLE 4.8
CLOSED LOOP ERROR OF REAL-TIME SLAM EXPERIMENTS ON EMBEDDED SYSTEM IN HOME ENVIRONMENT

Sequence	Start position	Environment	Closed loop error (cm)
1	1	Static	9.6
2	2	Static	6.0
3	1	Dynamic	8.4
4	2	Dynamic	10.9
	Avg.		8.7
	Std.		1.8

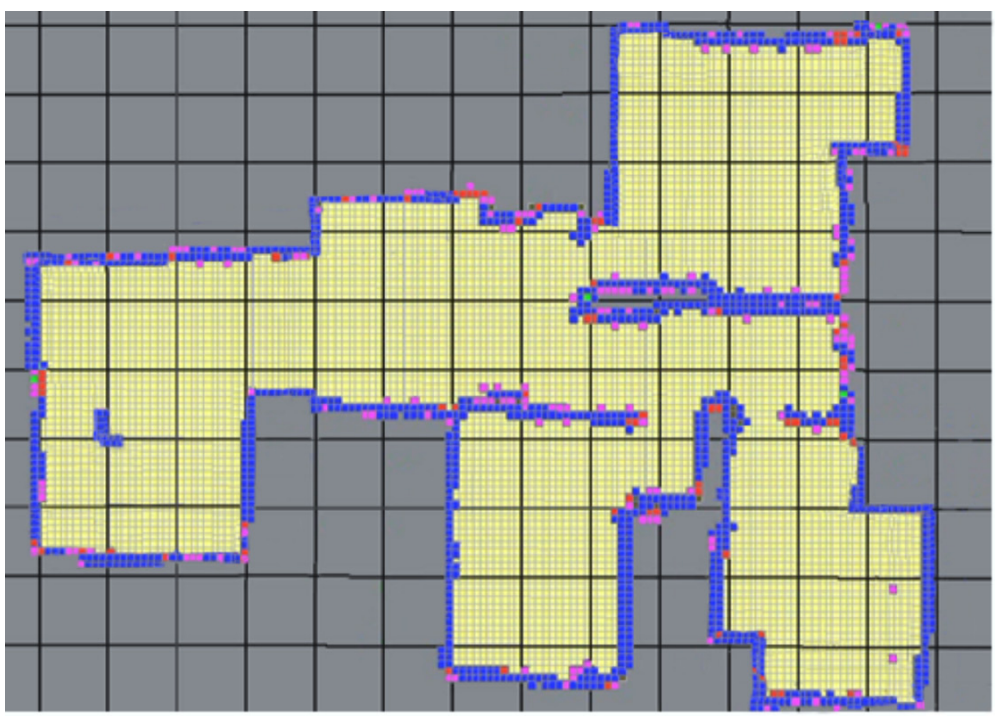


Figure 4.27 Generated obstacle grid map for autonomous navigation using the proposed SLAM on NXP4330Q board in home environment (sequence 3).

5. Conclusion

This dissertation presented a vision-based simultaneous localization and mapping (SLAM) for indoor service robots. A cost-effective forward-viewing mono-camera is adopted as a primary sensor, and robot wheel encoders and a gyroscope were utilized as supplementary sensors. The performance of current vision-based SLAM algorithms can be easily degraded in low-textured or dynamic challenging environments. Though the drift of error in these low-textured areas is inevitable, it can be reduced when vanishing point is utilized as a global feature. In this work, vanishing point and line feature based SLAM which can operate even in the low-textured indoor environments was proposed. The proposed vanishing point-based orientation estimation method makes SLAM problem into a simpler form. The equation for position estimation of the robot and landmark can be formulated in a linear form, and the proposed bundle adjustment corrects the estimates of a bundle of robot poses and landmark positions. The proposed method is computationally effective, so it can be executed on low-cost embedded system on real time. The experiments are conducted in both PC and embedded system. These include typical home environment, featureless indoor environment with motion capture system, and large scale indoor environment from a benchmark SLAM dataset. The proposed method is robust in various challenging indoor environments which contain low-

textured areas, moving people, or changing environments. The method is expected to be applicable on a low-cost embedded system for indoor service robots.

Appendix:

Performance Evaluation of Various Loop Detection Methods in Home Environment

There has been enormous works on the visual loop detection for SLAM. The visual loop detection is also referred as visual place recognition in computer vision and robotics community. The state-of-the-art vision-based SLAM algorithms mostly adopt visual bag-of-words (BoW) based loop detection methods. The FABMAP [29] method match the appearance of the current scene to a past place by converting the image into BoW representations built on SURF local features. The DBoW2 method [26] also uses BoW representations built on FAST corner detector combined with BRIEF descriptor. Inspired by the text retrieval system, DBoW2 uses term frequency-inverse document frequency (TF-IDF) score [80] for observation likelihood. As another approach for loop detection, holistic image matching approaches compute a single descriptor like Gist [81] for the whole image. BRIEF-Gist method [33] uses a BRIEF descriptor [31] for fast holistic image descriptor. SeqSLAM method [82] uses sum of absolute differences between contrast enhanced, low-resolution images. The matching is conducted on image sequences using a continuous Dynamic Time Warping technique.

To evaluate the performance of various loop detection methods, the experiments are conducted in a typical home environment as shown in Fig. A.1. The robot has explored the experimental environment while collecting the images from a forward-viewing camera. Images are captured when robot is moved more than 30 cm, or rotated more than 30 degree compared to the previous frame. This is conducted three times resulting three datasets, and these datasets are denoted as home loop dataset 1 to 3 in the following. The home loop dataset 1 and 2 are acquired when lights are turned on in daytime. The home loop dataset 3 is acquired when lights are turned on in nighttime. The images are collected at a resolution of 320 × 240 pixels. The example images for home loop datasets are illustrated in Fig. A.2. When comparing the two datasets in the experiments, first dataset is used to build the map for loop detection, and the second dataset is used to compare it with the map for loop detection. For robot platform, a robotic vacuum is used as shown in Fig. A.3. For quantitative evaluation of the loop detection methods, the ground truth robot pose is needed. When the ground truth robot pose is known, the detected two poses can be evaluated as true or false. However, it is hard to get ground truth robot pose in home environment. Instead, the SLAM based localization result is used as ground truth robot poses. The robotic vacuum used in the experiments is equipped with the upward-viewing camera based SLAM system [60]. Then, the loop detection result of all robot poses can be classified into four classes: true positive (*TP*), true negative (*TN*), false positive (*FP*), and false negative (*FN*). The precision and recall are

evaluated as follows:

$$precision = \frac{TP}{TP + FP}$$
$$recall = \frac{TP}{TP + TN} \quad (1)$$

In the experiments, DBoW2 [26], FABMAP [29], SeqSLAM [82], and BRIEF-Gist [33] algorithms are compared. Additionally, three versions of modified BRIEF-Gist based loop detection algorithms are tested. First, multiple BRIEF-Gist descriptor extraction method as proposed in Chapter 3.7.2 is tested (*BRIEF-Gist multiple extraction*). Second, original BRIEF-Gist with Bayesian filtering based loop detection method as proposed in Chapter 3.7.4 is tested (*BRIEF-Gist multiple extraction*). Third, multiple BRIEF-Gist descriptor extraction with Bayesian filtering based loop detection method is tested (*BRIEF-Gist multiple extraction + Bayesian filtering*). The last one is the final loop detection method that is applied to the proposed SLAM system. Figure A.4 and A.5 show the precision-recall plots of various loop detection methods for home loop dataset 1 vs 2 and dataset 1 vs 3, respectively. Overally, the performances for dataset 1 vs 3 are lower than that of dataset 1 vs 2, since illumination condition has changed. The DBoW2 and FABMAP methods show extremely low performance compared to other methods. The main reason for the performance degradation of BoW-based loop detection methods in home environment is that images from the home environment contain very few features. The BoW-based methods find loops using the distribution of extracted

descriptors of local features. When the input images contain very few features, the resulting indistinguishable distributions degrade the performance. On the other hand, holistic image descriptor-based methods (BRIEF-Gist and SeqSLAM) show much better results than BoW-based methods, since these methods do not rely on local feature extraction. When extracting multiple descriptor technique (*BRIEF-Gist multiple extraction*) or Bayesian filtering technique (*BRIEF-Gist Bayesian filtering*) is applied to the original BRIEF-Gist method, the performance gets better. When both techniques are combined (*BRIEF-Gist multiple extraction + Bayesian filtering*), it shows the most superior performance.

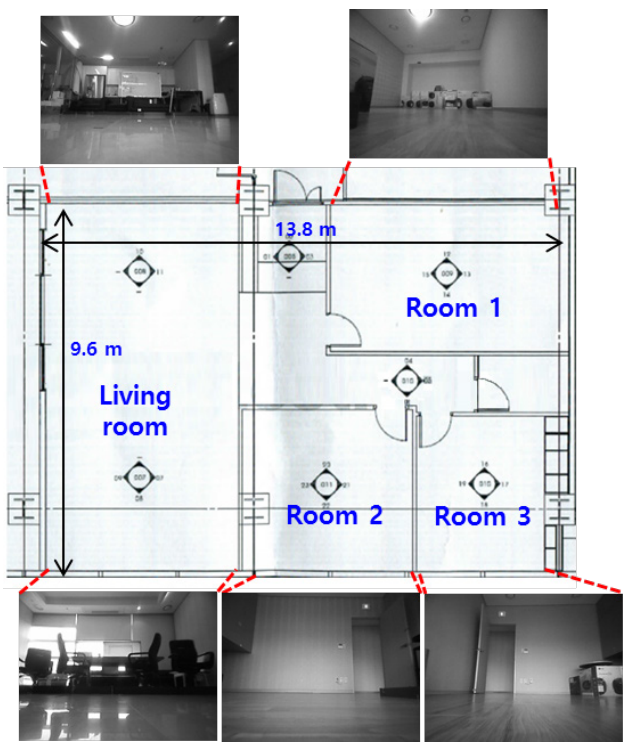
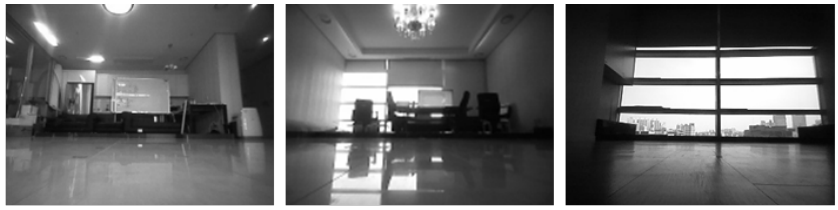


Figure A.1 Experimental environment for performance evaluation of various loop detection methods



(a) Home loop dataset 1 (Lights on in daytime)

(b) Home loop dataset 2 (Lights on in daytime)

(c) Home loop dataset 3 (Lights on in nighttime)

Figure A.2 Illumination conditions for home loop dataset 1 to 3.

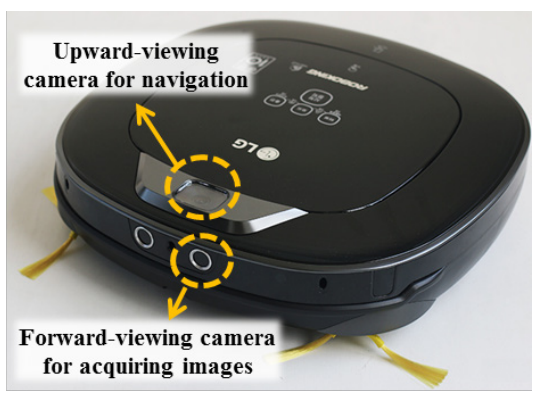


Figure A.3 Robot platform for acquiring home sequence dataset

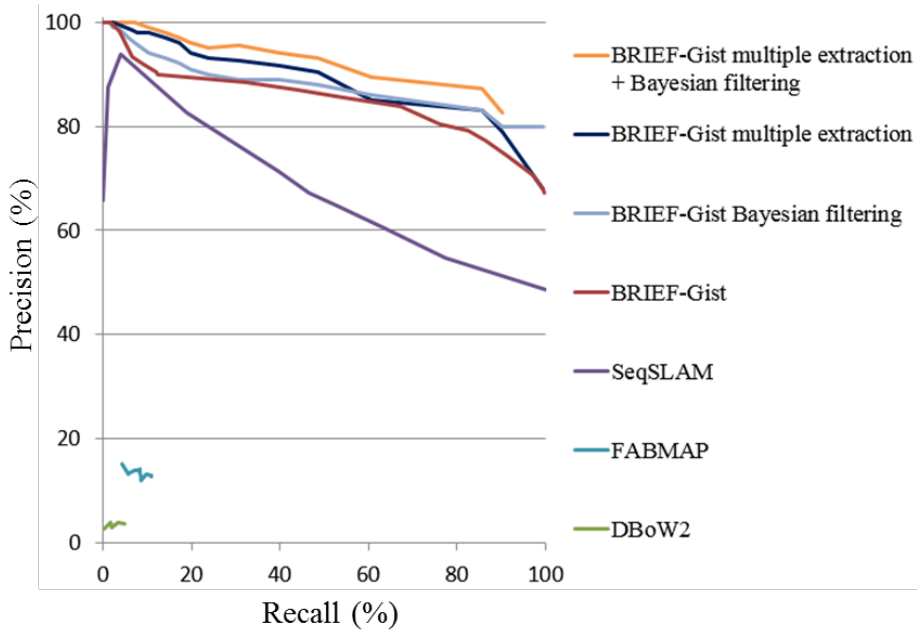


Figure A.4 Precision-recall curves of various loop detection methods for home loop dataset 1 vs home loop dataset 2

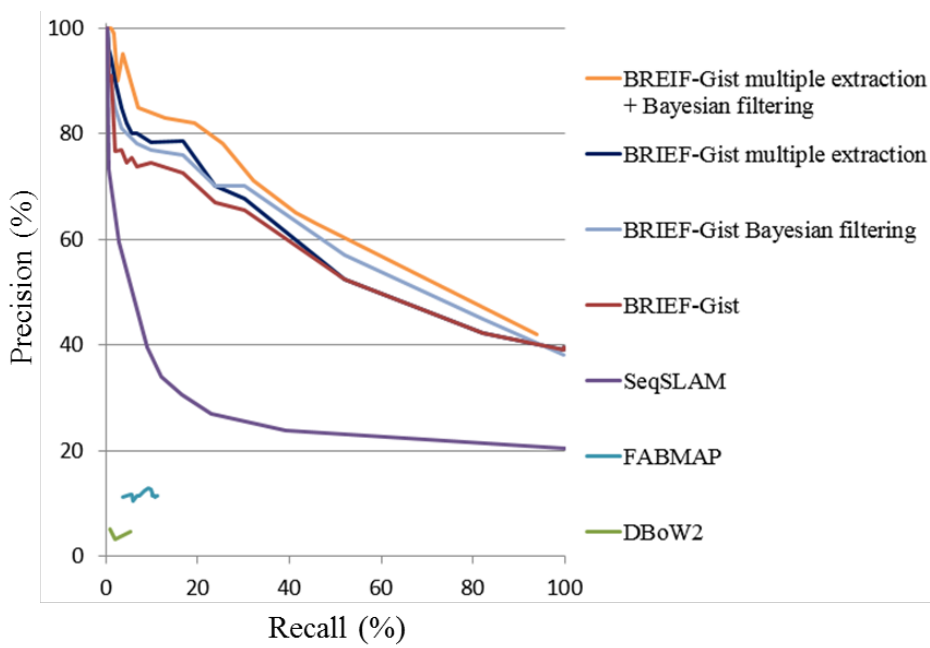


Figure A.5 Precision-recall curves of various loop detection methods for home loop dataset 1 vs home loop dataset 3

Reference

[1] T. Lee, B. Jang and D. Cho, “A novel method for estimating the heading angle for a home service robot using a forward-viewing mono-camera and motion sensors,” *International Journal of Control, Automation, and Systems*, vol. 13, no.3, pp. 709-717, 2015.

[2] T. Lee, D. Yi and D. Cho, “A monocular vision sensor-based obstacle detection algorithm for autonomous robots,” *Sensors*, vol. 16, no. 3, 2016.

[3] M. Dissanayake, P. Newman, S. Clark, H. F. Durrant-Whyte, and M. Csorba, “A solution to the simultaneous localization and map building (SLAM) problem,” *IEEE Transactions on Robotics and Automation*, vol. 17, no. 3, pp. 229–241, 2001.

[4] H. Durrant-Whyte, and T. Bailey, "Simultaneous localization and mapping: part I," *IEEE Robotics & Automation Magazine*, vol. 13, no. 2, pp. 99-110, 2006.

[5] H. Durrant-Whyte, and T. Bailey, "Simultaneous localization and mapping: part II," *IEEE Robotics & Automation Magazine*, vol. 13, no. 3, pp. 108-117, 2006.

[6] J. Aulinas, Y. Petillot, J. Salvi, and X. Llad´o, “The SLAM Problem: A survey,” in *Proc. of International Conference on Catalan Association Artificial Intelligence*, pp. 363–371, 2008.

[7] S. Thrun, W. Burgard, and D. Fox, Probabilistic Robotics. Cambridge, MA, USA: MIT Press, 2005

[8] G. Grisetti, R. Kummerle, C. Stachniss, and W. Burgard, "A Tutorial on Graph-Based SLAM," *IEEE Intelligent Transportation Systems Magazine*, vol. 2, no. 4, pp. 31-43, 2010

[9] G. Dissanayake, S. Huang, Z. Wang, and R. Ranasinghe "A review of recent developments in Simultaneous Localization and Mapping," in *Proc. of IEEE International Conference on Industrial and Information Systems*, Kandy, Sri Lanka, August 16-19, 2011, pp. 477-482.

[10] D. Scaramuzza and F. Fraundorfer, "Visual odometry [Tutorial]. Part I: The first 30 years and fundamentals," *IEEE Robotics Automation Magazine*, vol. 18, no. 4, pp. 80–92, 2011.

[11] F. Fraundorfer and D. Scaramuzza, "Visual odometry. Part II: Matching, robustness, optimization, and applications," *IEEE Robotics Automation Magazine*, vol. 19, no. 2, pp. 78–90, 2012.

[12] S. Lowry, N. Sünderhauf, P. Newman, J. Leonard, D. Cox, P. Corke, and M. Milford, "Visual place recognition: A survey," *IEEE Transactions on Robotics*, vol. 32, no. 1, pp. 1–19, 2016.

[13] C. Engels, H. Stewénius, and D. Nistér, "Bundle adjustment rules," *Photogrammetric computer vision*, vol. 2, pp. 124-131, 2006.

[14] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle adjustment—a modern synthesis," *International workshop*

on vision algorithms, pp. 298-372, 1999.

[15] F. Dellaert, “Factor graphs and GTSAM: A hands-on introduction,” Georgia Institute of Technology.

[16] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g2o: A general framework for graph optimization," *IEEE International Conference on Robotics and Automation*, pp. 3607-3613, 2011.

[17] S. Agarwal and K. Mierle, “Google Ceres Solver,” Available: http://ceres-solver.org, [Accessed: May 1, 2017].

[18] M. Kaess, A. Ranganathan, and F. Dellaert, “iSAM: Incremental smooth- ing and mapping,” *IEEE Transactions on Robotics*, vol. 24, no. 6, pp. 1365–1378, 2008.

[19] U. Frese, “Interview: Is SLAM Solved?,” *KI-Künstliche Intelligenz*, vol. 24, no. 3, pp. 255-257, 2010.

[20] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, L. Reid, and J. J. Leonard, “Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age,” *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309-1332, 2016.

[21] T. Deyle, “Why indoor robots for commercial spaces are the next big thing in robotics,” IEEE Spectrum, Mar. 1, 2017. Available: http://spectrum.ieee.org/automaton/robotics/robotics-ardware/indoor-robots-for-commercial-aces?utm_source=feedburner&utm_medium=

feed&utm_campaignFeed%3A+IeeeSpectrumRoboticsChannel+%28IEEE+Spectrum%3A+Robotics+2%29 [Accessed: Mar. 28, 2017].

[22] A. Geiger, P. Lenz, and R. Urtasun. “Are we ready for autonomous driving? The KITTI vision benchmark suite,” in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3355-3361, 2012.

[23] A. Bonarini, W. Burgard, G. Fontana, M. Matteucci, D. G. Sorrenti, and J. D. Tardos, “RAWSEEDS: Robotics Advancement through Web-publishing of Sensorial and Elaborated Extensive Data Sets,” in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006.

[24] E. Rosten, and T. Drummond, “Machine Learning for High-Speed Corner Detection," in *Proc. of European Conference on Computer Vision,* pp. 430-443, 2006.

[25] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, “ORB-SLAM: A Versatile and Accurate Monocular SLAM System,” *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147-1163, 2015.

[26] D. Gálvez-López, and J. D. Tardos, “Bags of Binary Words for Fast Place Recognition in Image Sequences,” *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188-1197, 2012.

[27] J. Engel, T. Schops, and D. Cremers, “LSD-SLAM: Large-scale direct monocular SLAM,” in *Proc. of European Conference on Computer Vision*, pp. 834-849, 2015.

[28] A. Glover, W. Maddern, M. Warren, S. Reid, M. Milford, and G. Wyeth, “OpenFABMAP: An open source toolbox for appearance-based loop closure detection,” in *Proc. of IEEE International Conference on Robotics and Automation*, pp. 4730-4735, 2012.

[29] M. Cummins and P. Newman. “FAB-MAP: Probabilistic localization and mapping in the space of appearance.” *International Journal of Robotics Research*. Vol. 27, no. 6 pp. 647-665, 2008.

[30] OpenCV library, Available: http://opencv.org/ [Accessed: May 1, 2017]

[31] M. Calonder, V. Lepetit, C. Strecha, P. Fua, “BRIEF: Binary Robust Independent Elementary Features,” In *Proc. of European Conference on Computer Vision*, pp. 778-792, 2010.

[32] H. Bay, T. Tuytelaars, and L. V. Gool, “SURF: Speeded Up Robust Features,” in *Proc. of European Conference on Computer Vision*, pp. 404-417, 2006.

[33] N. Sünderhauf, and P. Protzel, “BRIEF-Gist - closing the loop by simple means,” in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1234-1241, 2011.

[34] H. Lim, J. Lim, and H. J. Kim, “Real-time 6-DOF monocular visual SLAM in a large-scale environment,” in *Proc. of IEEE International Conference on Robotics and Automation*, pp. 1532-1539, 2014.

[35] G. Zhang, J. H. Lee, J. Lim, and I. H. Suh, “Building a 3-D Line-Based Map Using Stereo SLAM,” *IEEE Transactions on Robotics*,

vol. 31, no. 6, pp. 1364-1377, 2015.

[36] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. “DTAM: Dense tracking and mapping in real-time.” in *Proc. of IEEE International Conference on Computer Vision*, pp. 2320-2327, 2011.

[37] A. Concha and J. Civera. “DPPTAM: Dense piecewise planar tracking and mapping from a monocular sequence,” in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5686-5693, 2015.

[38] Nexell NXP4330Q, Available: http://www.nexell.co.kr/kor/pro/pro03.html [Accessed: May 1, 2017]

[39] A. Davison, “Real time simultaneous localisation and mapping with a single camera,” in *Proc. of IEEE International Conference on Computer Vision*, vol. 2, pp. 1403–1410, 2003.

[40] J. Civera, A. J. Davison, and J. M. M. Montiel, “Inverse depth parametrization for monocular SLAM,” IEEE Transactions on Robotics, vol. 24, no. 5, pp. 932–945, 2008.

[41] H. Strasdat, J. M. M. Montiel, and A. J. Davison, “Visual SLAM: Why filter?” *Image and Vision Computing*, vol. 30, no. 2, pp. 65–77, 2012.

[42] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, “FastSLAM: A factored solution to the simultaneous localization and mapping problem,” *in Proc. of AAAI National Conference on Artificial Intelligence*, pp. 593-598, 2002.

[43] E. Eade, and T. Drummond, “Scalable monocular SLAM,” in *Proc. of Computer Vision and Pattern Recognition*, pp. 469-476, 2006.

[44] J. Kwon, and K. Lee, “Monocular slam with locally planar landmarks via geometric Rao-blackwellized particle filtering on lie groups,” in *Proc. of Computer Vision and Pattern Recognition*, pp. 1522-1529, 2010.

[45] S. Hwang and J. Song, “Monocular vision-based SLAM in indoor environment using corner, lamp, and door features from upward-looking camera,” *IEEE Transactions on Industrial Electronics*, vol 58, no. 10, pp. 4804-4812, 2011.

[46] L. Clemente, A. Davison, I. Reid, J. Neira, and J. Tardós, “Mapping Large Loops with a Single Hand-Held Camera,” *Robotics: Science and Systems*, vol. 2, 2007.

[47] G. Klein, and D. Murray, “Parallel tracking and mapping for small AR workspaces,” in *Proc. of the IEEE and ACM International Symposium on Mixed and Augmented Reality*, pp. 225-234, 2007.

[48] G. Klein, Georg and D. Murray, “Improving the agility of keyframe-based SLAM,” in *Proc. of European Conference on Computer Vision,* pp. 802-815, 2008.

[49] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd, “Real time localization and 3D reconstruction,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 1, pp. 363–370, 2006.

[50] C. Forster, M. Pizzoli, and D. Scaramuzza, “SVO: Fast semi-direct

monocular visual odometry," in *Proc. of IEEE International Conference on Robotics and Automation*, pp. 15-22, 2014.

[51] E. Eade, F. Philip and M. Munich, "Monocular graph SLAM with complexity reduction," in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3017-3024, 2010.

[52] H. Strasdat, J. Montiel, and Andrew J. Davison, "Scale drift-aware large scale monocular SLAM," *Robotics: Science and Systems,* 2010.

[53] H. Strasdat, A. Davison, J. Montiel, and K. Konolige, "Double Window Optimisation for Constant Time Visual SLAM," in *Proc. of the IEEE International Conference on Computer Vision*, pp. 2352-2359, 2011.

[54] K. Konolige and M. Agrawal, "FrameSLAM: From bundle adjustment to real-time visual mapping," *IEEE Transactions on Robotics*, vol. 24 no. 5, pp. 1066-1077, 2008.

[55] P. Smith, I. D. Reid, and A. J. Davison. "Real-time Monocular SLAM with Straight Lines." in *Proc. of British Machine Vision Conference*, pp. 17-26, 2006.

[56] T. Lemaire, and S. Lacroix, "Monocular-vision based SLAM using Line Segments," in *Proc. IEEE International Conference on Robotics and Automation*, pp. 2791-2796, 2007.

[57] E. Perdices, L. M. López, and J. M. Canas, "LineSLAM: Visual Real Time Localization Using Lines and UKF," in *Proc. of ROBOT2013: First Iberian Robotics Conference*, pp. 663-678, 2014.

[58] J. Sol` a, T. Vidal-Calleja, and M. Devy, “Undelayed initialization of line segments in monocular SLAM,” in *Proc. IEEE International Conference on Intelligent Robots and Systems*, pp. 1553-1558, 2009.

[59] W. Y. Jeong, and K. M. Lee, “Visual SLAM with Line and Corner Features,” in *Proc. IEEE International Conference on Intelligent Robots and Systems*, pp. 2570-2575, 2006.

[60] S. S. Lee, and S. H. Lee, “Embedded visual SLAM: Applications for low-cost consumer robots,” *IEEE Robotics & Automation Magazine*, vol. 20, no.4, pp. 83-95, 2013.

[61] H. D. Choi, D. Y. Kim, J. P. Hwang, C. W. Park, and E. T. Kim, “Efficient Simultaneous Localization and Mapping Based on Ceiling-View: Ceiling Boundary Feature Map Approach,” *Advanced Robotics*, vol. 26, no. 5-6, pp. 653-671, 2012.

[62] H. D. Choi, R. S. Kim, and E. T. Kim, "An Efficient Ceiling-view SLAM Using Relational Constraints Between Landmarks," *International Journal of Advanced Robotic System*, vol. 11, no. 4, 2014.

[63] V. Huttunen and P. Robert, “A monocular camera gyroscope,” *Gyroscopy and Navigation*, vol. 3, no. 2, pp. 124-131. 2012.

[64] W. Elloumi, S. Treuillet, and R. Leconge, “Real-Time Estimation of Camera Orientation by Tracking Orthogonal Vanishing Points in Videos,” in *Proc. of International Conference on Computer Vision and Theory Application*, pp. 215-222, 2013.

[65] Y. H. Lee, C. Nam, K. Y. Lee, Y. S. Li, S. Y. Yeon, and N. L. Doh, "VPass: Algorithmic compass using vanishing points in indoor environments," in *Proc. IEEE International Conference on Intelligent Robots and Systems*, pp. 936-941, 2009.

[66] G. Zhang, D. H. Kang and I. H. Suh, "Loop closure through vanishing points in a line-based monocular SLAM," in *Proc. IEEE International Conference on Robotics and Automation*, pp. 4565-4570, 2012.

[67] F. Camposeco and M. Pollefeys, "Using vanishing points to improve visual-inertial odometry," in *Proc. IEEE International Conference on Robotics and Automation*, pp. 5219-5225, 2015.

[68] H. Zhou, D. Zou, L. Pei, R. Ying, P. Liu, and W. Yu, "StructSLAM: Visual SLAM with building structure lines," *IEEE Transactions on Vehicle Technology*, vol. 64, no. 4, pp. 1364–1375, 2015.

[69] O.D. Faugeras. *Three-Dimensional Computer Vision*. MIT Press. 1993.

[70] J.L. Mundy and A. Zisserman. (Eds). *Geometric Invariants in Computer Vision*. MIT Press. 1992.

[71] J. M. Coughlan and A. L. Yuille, "Manhattan world: Compass direction from a single image by bayesian inference," *in Proc. of the IEEE International Conference on Computer Vision*, pp. 941-947, 1999.

[72] G. Schindler and F. Dellaert, "Atlanta world: An expectation max-

imization framework for simultaneous low-level edge grouping and camera calibration in complex man-made environments," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 203-209, 2004.

[73] V. G. R. Grompone, J. Jakubowicz, J. M. Morel, and G. Randall, "LSD: A fast line segment detector with a false detection control," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 4, pp. 722-732, 2010.

[74] L. Zhang, H. Lu, X. Hu, and R. Koch, "Vanishing Point Estimation and Line Classification in a Manhattan World with a Unifying Camera Model," *International Journal of Computer Vision*, vol. 117, no. 2, pp. 111-130, 2015.

[75] R. Hartley, A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge university press (second edition).

[76] J. H. Lee, G. Zhang, J. W. Lim, and I. H. Suh, "Place recognition using straight lines for vision-based SLAM," *in Proc. of IEEE International Conference on Robotics and Automation*, pp. 3799-3806, 2013.

[77] S. M. Lowry, G. F. Wyeth and M. J. Milford, "Towards training-free appearance-based localization: probabilistic models for whole-image descriptors," *in Proc. of IEEE International Conference on Robotics and Automation*, pp. 711-717, 2014.

[78] Y. Liu, and H. Zhang, "Visual loop closure detection with a compact

image descriptor," *in Proc. of IEEE International Conference on Intelligent Robots and Systems*, pp. 1051-1056, 2012.

[79] H. Choset and P. Philippe, "Coverage path planning: The boustrophedon cellular decomposition." *Field and Service Robotics*, 203-209, 1998.

[80] T. Nicosevici and R. Garcia, "Automatic visual bag-of-words for online robot navigation and mapping," *IEEE Transactions on Robotics*, Vol. 28, No. 4, pp. 886-898, 2012.

[81] A. Oliva and A. Torralba, "Building the gist of a scene: The role of global image features in recognition," *Progress in brain research*, vol. 155, pp. 23-36, 2006.

[82] M. J. Milford and G. F. Wyeth, "SeqSLAM: Visual route-based navigation for sunny summer days and stormy winter nights," in *Proc. of the IEEE International Conference on Robotics and Automation*, pp. 1643-1649, 2012.

초록

본 논문은 전방향 단안카메라를 이용한 동시적 위치인식 및 지도작성 방법을 제안한다. 제안하는 방법은 저가의 임베디드 시스템 기반의 실내 서비스로봇에 적용하기 위해 개발되었다. 제안하는 방법은 동시적 위치인식 및 지도작성을 위하여 로봇에 부착된 저가의 전방향 단안카메라를 주 센서로 사용하고, 로봇 휠 인코더와 자이로스코프를 보조 센서로 사용한다. 제안하는 방법은 특징이 적은 영역, 움직이는 사람이나 변화하는 환경을 포함하는 다양한 어려운 실내환경에서 강인하게 동작할 수 있다. 위치인식을 위한 랜드마크로는 영상에서 추출한 소실점과 선분을 사용한다. 제안하는 소실점 기반의 로봇 각도추정방법은 동시적 위치인식 및 지도작성 방법의 비선형 추정문제를 단순화시켜준다. 소실점 기반 로봇 각도추정을 이용하여 로봇의 각도추정으로부터 로봇의 이동 변화량 추정 및 랜드마크 위치추정 문제를 분리할 경우, 로봇 이동 변화량 추정과 랜드마크 추정을 단순한 선형 모델로 표현할 수 있다. 이 모델을 이용하여 제안하는 local map 보정방식은 카메라 자세와 랜드마크의 위치를 효과적으로 보정한다. 제안하는 local map 보정방식은 standard bundle adjustment 방식에 비해 정확도가 더 높으며 연산속도가 3.6배 빠르다. 장시간

주행에서의 정확도를 높이기 위해, 로봇이 이전에 방문하였던 장소에 재방문 할 경우, 확률기반의 루프 검출과 로봇 위치보정을 수행한다. 성능평가를 위해 다양한 실내 환경에서 실험을 수행하였다. 실험환경은 가정, Vicon 모션캡쳐시스템이 구비된 공간을 포함한다. 위 환경은 특징이 적은 영역, 움직이는 사람이나 변화하는 환경을 포함하여 동시적 위치인식 및 지도작성을 수행하기 어려운 환경이다. RAWSEEDS 벤치마크 데이터세트를 이용하여서도 제안하는 방법의 성능평가를 수행하였다. 실시간 성능평가에서는 제안하는 방법을 저가의 임베디드 시스템에 구현하고 자율주행 시스템과 연동하여 실험을 수행하였으며, 제안하는 방식이 저가의 실내 서비스로봇에 적용가능함을 확인하였다.

주요어 : 동시적 위치추정 및 지도작성, 단안카메라, 소실점, 선분, 실내서비스로봇, 임베디드 시스템.

학번 : 2011-20914