The 2010 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 18-22, 2010, Taipei, Taiwan

SoF-SLAM: Segments-on-Floor-based Monocular SLAM

Guoxuan Zhang and Il Hong Suh, *Senior Member, IEEE*

***Abstract*— In this paper, we propose a novel monocular SLAM method in corridor environment which employs Segments-on-Floor (SoF) as feature data. Given that the height of the camera and the angle between the camera and the floor are known, an image of the SoF can be efficiently distinguished from the other space-lines by a simple data-association method, deriving the line correspondence from a simplified homography matrix of two sequentially gathered images. Furthermore, use of SoF simplifies the analysis of the geometrical property of the camera projection matrix. Therefore, we can reconstruct SoF by using a one-step inverse projection. Once SoF is calculated from visual data processing, they are then used in a normal SLAM process as feature data. We employ a simple particle filter in our corridor SLAM. Experimental results show that it is sufficient for mapping a moderately sized building environment.**

I. Introduction

Vision-based Simultaneous Localization And Mapping (vSLAM) methods have advantages over range sensor-based SLAM methods, in that, unlike the restricted functionality of proximity detection provided by the range sensor, the vision sensor provides richer information for robots to perceive and move and manipulate its surrounding environment. Thanks to the development of 3D computer vision technologies in the past decade, extracting distance information from multiple viewing angles has become a well utilized technique in the field of robot navigation.

The stereo camera is a well known device for capturing information in the three-dimensional world. A stereo camera is a device that can reconstruct a 3D image given two separate viewpoints of the same scene. In contrast, a monocular camera can only reconstruct 3D image by using images gathered sequentially at different time instances. If the camera becomes frozen in the same position and the environment is composed of only still objects it becomes impossible to create 3D images using a monocular camera without any knowledge of the external world. However, if the camera is kept in motion, then it is possible to reconstruct a 3D scene from two images by analyzing the disparity of the same features presented at different images. In this sense, a mobile robot provides a natural platform to carry a monocular camera for reconstructing a 3D environment.

This work was supported by the Technology Innovation Program of MKE/KEIT [2008-F-038-01, Development of Context Adaptive Cognition Technology]

G. Zhang is with the Division of Computer Science and Engineering, College of Engineering, Hanyang University, Korea. (E-mail: imzgx@incorl.hanyang.ac.kr).

I. H. Suh is with the Division of Computer Science and Engineering, College of Engineering, Hanyang University, Korea. (E-mail: ihsuh@hanyang.ac.kr. All correspondence should be addressed to I. H. Suh.)

Fig. 1. Examples of indoor environment that include rich sources of lines. These are formed by floor tiles, furniture, interior moldings, doors, boundaries between walls and floor or ceilings.

On the other hand, the most commonly used visual feature in vSLAM is the point-based feature. The point-based visual feature can be easily identified, and many 3D computer vision algorithms are more suite for this type of feature. However, when a map is composed of clouds of many 3D points, the map has never provided any physical meaning of these 3D points, since a point has zero dimensions in a mathematical sense. A common solution to this problem is to render a surface into densely clustered points in such a way that the cluster of points become easier to understand by the human eye.

As a line spans the one-dimensional space, and, therefore, it is possible to describe the environment only by means of lines. It is worth noting that the geometrical property of one-dimensionality of the lines also has the advantage over zero dimensionality of points, in the sense that lines can provide stable and long distance guidance for robot navigation. As Fig. 1 shows, there are a lot of linear components in structured indoor environments for a robot to perform vSLAM.

In many previous studies related to line-based vSLAM, the line was commonly treated as a general 3D-space object, and consequently, there was large uncertainty when a line was initialized. In one of the early works [1]–[4], the 3D line was represented as a line lies on a plane, and the plane was determined by two rays back-projected from two end points of the image segment [1]. In [2], the line was encoded in several small edge landmarks, called ‘*edgelets*’, which could also be used to represent a curved line. In [3] and [4], the 3D line was represented using Plücker coordinates. The former represented the hypothetical line as a geometric series, and the latter adopted the Inverse Depth Parameterization (IDP) method [5]. What these pioneering works had in common was that all approaches were verified only in small-sized real experimental environments.

The most challenging problem in monocular SLAM is that the robot has to move forward when visual data is gathered from a forward-facing camera. Unlike many line-based vSLAM studies where the moving trajectory is perpendicular to the optical axis of the camera, we mount our camera in the same direction as that of the robot movement.

The motivation of our work is to fully exploit the constraints of the Segments-on-Floor (SoF) as feature data. If we restrict ourselves to treating only SoF, the challenging problem of data-association becomes easier to control, and it also has some benefit on the process of reconstructing the segment map from sequential images. Since our approach exclusively uses SoF as sensory input in SLAM processing, we name it SoF-SLAM. For the purpose of future extension, neither Manhattan world assumption [6] nor vanishing component-related computation were used in our work and, all of the two-view geometries are dealt with in a general computational framework.

This paper is structured as follows. Section II presents the visual data processing for SoF, which provides the foundational mathematics for our research. In Section III, we discuss the development of vSLAM strategy using SoF as feature data. Experiments using an actual robot platform and conclusions are presented in Sections IV and V, respectively.

II. Visual Data Processing

A. Basic Geometry

A *line* is an infinitely-extending one-dimensional figure that has no curvature; a *segment* is a part of a line bounded by two distinct end points. In this paper, we use the terms line and segment indiscriminately whenever the meaning is clear from its context.

In the image plane, two points $\mathbf{x}_1 = (x_1, y_1, 1)^T$ and $\mathbf{x}_2 = (x_2, y_2, 1)^T$ can determine a line $\mathbf{l}$ given by $\mathbf{l} = \mathbf{x}_1 \times \mathbf{x}_2$. Globally we represent a line by two end points in 2-vector space; however, locally, while comparing the similarity or computing the sensor model, we represent a line by the polar coordinate form of $\mathbf{l} = (\rho, \theta)^T$, where ρ is the length of the incidence line from the origin of the reference coordinate frame to line $\mathbf{l}$, and θ is the angle between the line $\mathbf{l}$ and the horizontal axis.

In this paper, the endpoints representation of the line uses the world coordinate frame, and the polar-coordinate representation uses the camera coordinate frame as their respective reference coordinate frames. The reason for this classification is that the polar coordinate is more efficient for the purpose of comparison between similar sensor data, but if the robot is far from the origin of the world coordinate frame, then the accumulated error is prone to be magnified if the robot still maintains the polar coordinate based on the world origin. Therefore, the robot has to maintain the endpoints representation in the world coordinate frame, but whenever it needs to perform a similarity comparison, the segments should be temporarily transformed into polar coordinates based on the camera coordinate frame.

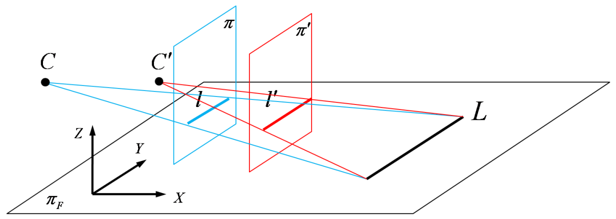


Fig. 2. Line correspondence of SoF in consecutive images, where space segment L on floor is projected to image plane $\boldsymbol{\pi}$ and $\boldsymbol{\pi}'$ as $\mathbf{l}$ and $\mathbf{l}'$, respectively. There exists a homography for $\mathbf{l}$ and $\mathbf{l}'$ related to L.

A homography matrix, $\mathbf{H}$, is a mapping that transforms points from one plane to another while maintaining collinearity. We are concerned with two cases of homography mapping: first, for a fixed camera center there exists a homography between coplanar space points and the corresponding points at the image plane. We will use this property in Section II.C. Second, for moving camera centers there exists a homography between two image planes provided the two images are related by the same coplanar space points. We will use this property in Section II.B. For two planes $\boldsymbol{\pi}$ and $\boldsymbol{\pi}'$, if $\mathbf{x}, \mathbf{l} \in \boldsymbol{\pi}$ and $\mathbf{x}', \mathbf{l}' \in \boldsymbol{\pi}'$, then under the point transformation $\mathbf{x}' = \mathbf{H}\mathbf{x}$, the line transforms as $\mathbf{l}' = \mathbf{H}^{-T}\mathbf{l}$.

When the camera calibration matrix, $\mathbf{K}$, is given from an explicit calibration process, and the coordinates of the camera center $\tilde{\mathbf{C}}$ and the rotation matrix of the camera $\mathbf{R}$ are provided by a SLAM process, then we can project a 3D point $\mathbf{X} = (X, Y, Z, 1)^T$ to an image point $\mathbf{x} = (x, y, 1)^T$ through the camera projection matrix $\mathbf{P}$:

$$\mathbf{x} = \mathbf{P}\mathbf{X} \tag{1}$$

The 3×4 homogeneous camera projection matrix $\mathbf{P}$ can be further represented as follows:

$$\mathbf{P} = \mathbf{K}\mathbf{R}[\mathbf{I} \mid -\tilde{\mathbf{C}}] \tag{2}$$

Here $\mathbf{I}$ is the 3×3 identity matrix. We adopted the world coordinate frame as Fig. 2 shows, where the floor is defined by the X-Y plane, and the position of the robot always has a fixed value of 0 along the Z-*Axis*.

B. Line Correspondence

Unlike point-based vSLAM approaches in which each individual feature data is encoded in a unique descriptor, in line-based vSLAM, feature data is only encoded in a geometric object. No explicit descriptor can be directly used to compare similarity, but the geometric properties must be extracted every time for comparison. After a robot moves from one position to another, the robot must then be able to decide which segment in the second image corresponds to the segment in the first image. This is the form of the data-association problem expressed in line-based vSLAM approaches.

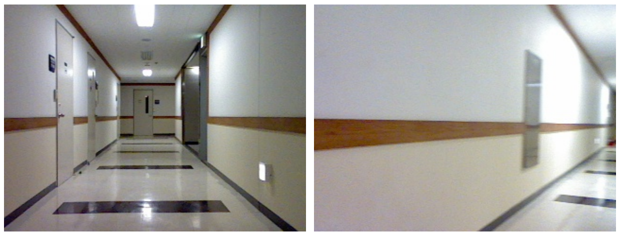

(a) Light reflection (b) Robot rotation

Fig. 3. The length and the angle of segments suffer from severe changes in the real environment.

In Fig. 2, $\boldsymbol{\pi}_F$ denotes the floor plane, $\boldsymbol{\pi}$ and $\boldsymbol{\pi}'$ denote consecutive image planes made by the robot motion, and $\mathbf{C}$ and $\mathbf{C}'$ denote corresponding camera centers. The spatial SoF $\mathbf{L}$ is projected to $\boldsymbol{\pi}$ and $\boldsymbol{\pi}'$ as $\mathbf{l}$ and $\mathbf{l}'$, respectively. Since the plane made by $\mathbf{L}$ and $\mathbf{C}$ is $\mathbf{P}^T\mathbf{l}$, and the plane made by $\mathbf{L}$ and $\mathbf{C}'$ is $\mathbf{P}'^T\mathbf{l}'$, we can, therefore parameterize $\mathbf{L}$ by a 2×4 matrix as follows:

$$\mathbf{L} = \begin{bmatrix} \mathbf{l}^T\mathbf{P} \\ \mathbf{l}'^T\mathbf{P}' \end{bmatrix} \tag{3}$$

where $\mathbf{P}$ and $\mathbf{P}'$ are projection matrices corresponding to $\mathbf{C}$ and $\mathbf{C}'$, respectively. Unfortunately, before we can apply equation (3) to calculate the parameters of line $\mathbf{L}$, we need to decide the correspondence between $\mathbf{l}$ and $\mathbf{l}'$. In other words, lines $\mathbf{l}$ and $\mathbf{l}'$ should be data-associated. It is impossible to determine the line correspondence from the relative position or length of segments by a simple comparison, because these measurements suffer from severe lighting conditions, as Fig. 3 (a) illustrates, and due to the motion of the robot, especially when the robot is performing rotation as shown in Fig. 3 (b). Accordingly the relative transition and rotation of the robot have to be considered in the data-association process.

Returning to Fig. 2, let us suppose that the position of the robot corresponding to image planes $\boldsymbol{\pi}$ and $\boldsymbol{\pi}'$ are $\mathbf{X}$ and $\mathbf{X}'$, and the relative transition and rotation of $\mathbf{X}'$ against $\mathbf{X}$ are represented by $\bar{\mathbf{t}}$ and $\bar{\mathbf{R}}$, respectively. Since all SoF exist on the plane $\boldsymbol{\pi}_F$, so there exists a homography $\mathbf{H}$ between $\boldsymbol{\pi}$ and $\boldsymbol{\pi}'$; and the lines, $\mathbf{l}$ and $\mathbf{l}'$, projected by the same spatial line $\mathbf{L}$ must be associated with $\mathbf{H}$. If we can compute this $\mathbf{H}$, then the line correspondence problem can be solved.

We follow the method described in [7], and set $\mathbf{C}$ as the origin of the world coordinate frame. The projection matrices become:

$$\mathbf{P} = \mathbf{K}[\mathbf{I} \mid \mathbf{0}] \tag{4}$$

$$\mathbf{P}' = \mathbf{K}'[\bar{\mathbf{R}} \mid \bar{\mathbf{t}}] \tag{5}$$

Here $\mathbf{K}$ and $\mathbf{K}'$ are the camera calibration matrices of $\mathbf{C}$ and $\mathbf{C}'$, respectively. The floor $\boldsymbol{\pi}_F$ has coordinates $\boldsymbol{\pi}_F = (\mathbf{n}^T, h)^T$, where $\mathbf{n}$ is the normal of $\boldsymbol{\pi}_F$, and therefore $\mathbf{n} = (0,0,1)^T$, and h is the height of the camera. All points $\mathbf{X}_F = (\tilde{\mathbf{X}}_F^T, 1)^T$ on $\boldsymbol{\pi}_F$ satisfy the condition as follows:

$$\mathbf{n}^T\tilde{\mathbf{X}}_F + h = 0 \tag{6}$$

When neither $\mathbf{K}$ nor $\mathbf{K}'$ are taken into account, the homography for the cameras $\mathbf{P} = [\mathbf{I} \mid \mathbf{0}]$, $\mathbf{P}' = [\bar{\mathbf{R}} \mid \bar{\mathbf{t}}]$ is

$$\mathbf{H} = \bar{\mathbf{R}} - \bar{\mathbf{t}}\mathbf{n}^T/h \tag{7}$$

After applying the transformations $\mathbf{K}$ and $\mathbf{K}'$ to the images, we finally obtain the cameras $\mathbf{P} = \mathbf{K}[\mathbf{I} \mid \mathbf{0}]$ and $\mathbf{P}' = \mathbf{K}'[\bar{\mathbf{R}} \mid \bar{\mathbf{t}}]$, and the resulting homography is

$$\mathbf{H} = \mathbf{K}'(\bar{\mathbf{R}} - [\ \mathbf{0}\ \mathbf{0}\ \bar{\mathbf{t}}\ ]/h)\mathbf{K}^{-1} \tag{8}$$

where $\mathbf{0}$ is the 3×1 zero-vector. Since the internal parameters of the camera have not changed within our vSLAM process, accordingly $\mathbf{K} = \mathbf{K}'$ for all instances. Equation (8) means that for any spatial line $\mathbf{L}$ on the floor, we can find the data-associated line $\mathbf{l}$ and $\mathbf{l}'$ using $\mathbf{H}$.

For most robots, the values of $\bar{\mathbf{R}}$ and $\bar{\mathbf{t}}$ are usually provided by the odometer reading, and the error is rather small between two consecutive data readings. Given these two values, in Fig. 2 the homography $\mathbf{H}$ between two robot image planes $\boldsymbol{\pi}$ and $\boldsymbol{\pi}'$ can be calculated from equation (8). Since lines $\mathbf{l}$ and $\mathbf{l}'$ are imaged by the same spatial line $\mathbf{L}$, theoretically $\mathbf{l}' = \mathbf{H}^{-T}\mathbf{l}$. However, due to uncontrollable odometer and imaging errors, this is hardly the case. To find the corresponding line $\mathbf{l}'$ at $\boldsymbol{\pi}'$ relating to $\mathbf{l}$ at $\boldsymbol{\pi}$, we transform the line $\mathbf{l}$ to get a newly mapped line $\mathbf{l}'' = \mathbf{H}^{-T}\mathbf{l}$, and check if they are imaged from the same SoF by using the next criterion:

$$\varepsilon = \alpha(\rho' - \rho'') + \beta(\theta' - \theta'') < \Theta \tag{9}$$

Here $\mathbf{l}' = (\rho', \theta')$, $\mathbf{l}'' = (\rho'', \theta'')$; α, β and Θ are predefined parameters for checking the similarity between two lines, and ε is the measured error between them. If equation (9) is satisfied, and the line $\mathbf{l}'$ and $\mathbf{l}''$ overlap at $\boldsymbol{\pi}'$, then we treat $\mathbf{l}$ and $\mathbf{l}'$ as if they are data-associated. We say that two segments overlap in the same image, when the first segment is projected onto the second segment they share the same part in the direction of the second segment.

C. Simplified Projection Matrix

As was noted in equation (3), as soon as the line correspondence of $\mathbf{l}$ and $\mathbf{l}'$ can be identified between two image planes $\boldsymbol{\pi}$ and $\boldsymbol{\pi}'$, then the spatial line $\mathbf{L}$ on the floor can be uniquely determined. This calculation not only needs to consider two projection matrices, $\mathbf{P}$ and $\mathbf{P}'$, but also the resulting space line $\mathbf{L}$, expressed in a 2×4 matrix, further manipulation, therefore becomes a more complicated process. In the remainder of this section we will introduce a new homography constructed in the context of a 1-view 2-planar geometric relationship.

In Fig. 4, a spatial point, $\mathbf{X} = (X, Y, Z, 1)^T$, is projected to an image point $\mathbf{x} = (x, y, 1)^T$ on the plane $\boldsymbol{\pi}$ by a projection matrix $\mathbf{P}$ which was related to the camera center $\mathbf{C}$, as evidence in equation (1). We replace $\mathbf{P}$ with the elements-enumerated form, $[\mathbf{p}_1\ \mathbf{p}_2\ \mathbf{p}_3\ \mathbf{p}_4]$, where $\mathbf{p}_i$ denotes

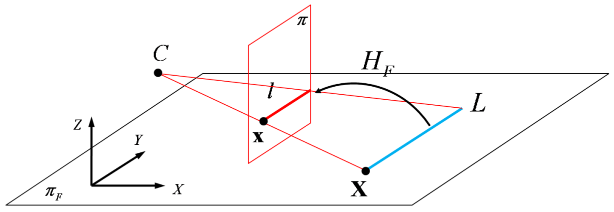


Fig. 4. If the internal and external camera parameters are given, then the inverse projection from image to floor can be simplified as per a simple homography.

the i-th column vector of $\mathbf{P}$, and we repeat equation (1) in detail:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{p}_1 & \mathbf{p}_2 & \mathbf{p}_3 & \mathbf{p}_4 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \qquad (10)$$

As we stated before, for the spatial point $\mathbf{X}$ on floor $\boldsymbol{\pi}_F$ it has the zero value on the Z-*Axis*, that is $\mathbf{X} = (X, Y, 0, 1)^T$, and consequently equation (10) becomes:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{p}_1 & \mathbf{p}_2 & \mathbf{p}_4 \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \qquad (11)$$

We represent the 3×3 matrix $[\mathbf{p}_1\ \mathbf{p}_2\ \mathbf{p}_4]$ as $\mathbf{H}_F$. Since $\mathbf{H}_F$ transforms any points on the floor $\boldsymbol{\pi}_F$ to the image plane $\boldsymbol{\pi}$, therefore, $\mathbf{H}_F$ is a homography and is also an invertible matrix. After the projection matrix is reduced to a homography matrix $\mathbf{H}_F$, we can project the image point $\mathbf{x}$ to a space point $\tilde{\mathbf{X}} = (X, Y, 1)^T$ inversely:

$$\tilde{\mathbf{X}} = \mathbf{H}_F^{-1}\mathbf{x} \qquad (12)$$

Furthermore, for any data-associated line $\mathbf{l}'$, we can reconstruct the corresponding spatial line $\tilde{\mathbf{L}}$ using the equation:

$$\tilde{\mathbf{L}} = \mathbf{H}_F^T\mathbf{l}' \qquad (13)$$

The whole process of visual data processing is shown in Fig. 5. After this step, the segments on floor $\tilde{\mathbf{L}}$ can be distinguished from all the other confusing segments from various spatial structures, and be ready for use in the map building process. In next section, we use the final segment $\tilde{\mathbf{L}}$ as sensor data to feed into the SLAM process.

Algorithm: Extracting SoF

1) Move from position $\mathbf{X}$ to position $\mathbf{X}'$.
2) Calculate $\bar{\mathbf{t}}$ and $\bar{\mathbf{R}}$ for consecutive robot positions $\mathbf{X}$ and $\mathbf{X}'$ from the odometer readings.
3) Extract segments $\mathbf{l}$ and $\mathbf{l}'$ for two sequentially gathered images $\mathcal{I}$ and $\mathcal{I}'$, corresponding to $\mathbf{X}$ and $\mathbf{X}'$.
4) Transform each segment $\mathbf{l}$ in $\mathcal{I}$ to $\mathbf{l}''$ in $\mathcal{I}'$ using the homography matrix of equation (8).
5) Check the similarity between $\mathbf{l}''$ and $\mathbf{l}'$ using equation (9), and choose the most similar $\tilde{\mathbf{l}}$ among $\mathbf{l}'$ as the corresponding line to $\mathbf{l}$.
6) Project $\tilde{\mathbf{l}}$ inversely to segment $\tilde{\mathbf{L}}$ on floor using equation (13).
7) Return $\tilde{\mathbf{L}}$ and go to Step 1).

Fig. 5. The algorithm for extracting SoF

III. vSLAM using SoF

A. SLAM Algorithm

The SLAM algorithms can be divided into two large families. The first group uses Kalman filters, and the extended Kalman filter (EKF-SLAM) is representative of this family. The second family of SLAM algorithms is based on particle filters, and representative of this family is the FastSLAM - a mix of Rao-Blackwellized particle filter and Kalman filtering [8].

Among the particle filter-based SLAM algorithms, DP-SLAM is one of the best known approaches which uses a particle filter to maintain a joint probability distribution over maps and robot positions [9]. DP-SLAM relies on laser sensor data and aims to achieve truly simultaneous localization and mapping without landmarks. DP-SLAM is known for its accuracy and that, in most cases, no special loop closing techniques are required for building a complete map. When performing the SLAM process it makes only a single pass over the sensor data.

Inspired by the work of DP-SLAM, we devised our SLAM algorithm based on a simple particle filter, but with visual data as sensory input to the SLAM process. For simplicity we used only a single map to track the observed segment data. This is a common trick used by the earlier researchers in an attempt to avoid the complicated work of map copying. In this method, a single map is maintained and used for localization for all of the particles. At each iteration a single most likely particle is chosen under competing strategies, and all other particles are ignored during the mapping stage. In traditional methods the map was normally updated only once based upon this greedy choice of robot position. We slightly revised the map update process, as our approach is able to update all related map data at each initialization step by using observed feature data. This was done mainly on behalf of the parameterized line information. At the next SLAM computational cycle, the chosen particle is resampled based on movements made by the robot, and the number of produced particles are the same as the number of particles applied in the previous step.

We adopt the strategy of using only a single map in our SLAM algorithm for another reason. As a novel approach of vSLAM, we intended to evaluate the mapping performance of using SoF as sensory input data on the SLAM process, but not to evaluate the whole performance of vSLAM. We gave more weight to the evaluation of the ability to perform vSLAM merely relying on using SoF as feature data.

In our particle filter, both of the motion model and the sensor model took the form of a Gaussian distribution. In the motion model, the mean is determined by the odometer reading; in the sensor model, the mean is determined by the values of ρ and θ based on the camera coordinate frame, which was temporarily calculated from two endpoints in the world coordinate frame.

Algorithm: SoF-SLAM

1) Move from position $\mathbf{X}$ to position $\mathbf{X}'$.
2) Calculate $\bar{\mathbf{t}}$ and $\bar{\mathbf{R}}$ for consecutive robot positions $\mathbf{X}$ and $\mathbf{X}'$ from odometer readings.
3) Sample $\mathbf{X}'$ as $\mathcal{N}$ particles using $\bar{\mathbf{t}}$, $\bar{\mathbf{R}}$ and the motion model.
4) For each hypothetical robot position, apply the algorithm in Fig. 5 to derive the image of SoF.
5) For $\mathcal{N}$ particles, compare newly observed segments with existing map data from sensor model, and choose a single particle.
6) For that particle, integrate newly observed segments into existing map data.
7) Substitute the position maintained by that particle to position $\mathbf{X}$ and return to Step 1).

Fig. 6. The algorithm for SoF-SLAM

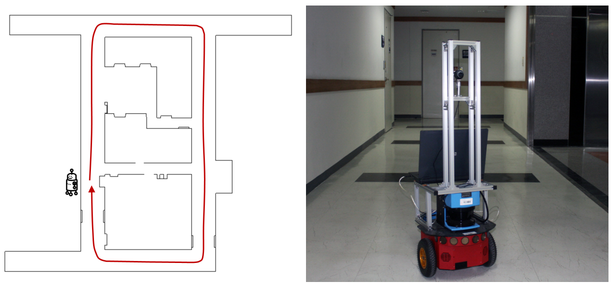


Fig. 7. The experiment was conducted in corridor environment and used a real robot platform.

B. Line Initialization

As the result of SoF-SLAM, the constructed map is composed of several SoFs, and each segment is represented by two endpoints. Therefore, in our case, the map building means that every segments produced by the SoF extraction algorithm in Fig. 5, is processed in one of two ways: one way is for it to be initialized as new feature data and newly added into the map; the other is that the observed segment is recognized as existing map data and it is used in the process of updating the corresponding segment in the previously constructed map.

It is clear from the algorithm in Fig. 5, that because we used two consecutively acquired images for extracting SoF, our algorithm should be classified as delayed initialization. We adopted an extending-only policy for line updating, which means that the newly extracted segments are used in line updating only when it lengthens the existing SoF. We followed a conservative strategy in our SoF-SLAM because when the angle and length of the SoF are extracted from the first image, they are more likely to be changed in the next observation. Therefore, the lazy methodology is more suitable for the SoF-SLAM.

Updating the existing segments map data $\mathbf{L} = (\rho, \theta)$ based on newly observed segment $\mathbf{L}' = (\rho', \theta')$ is as follows:

$$\begin{aligned} \rho &\leftarrow \rho\gamma + \rho'(1-\gamma) \\ \theta &\leftarrow \theta\gamma + \theta'(1-\gamma) \end{aligned} \tag{14}$$

where $\gamma = \frac{\delta}{1+\delta}$, and δ represents the time of updating from the first initialization of $\mathbf{L}$ to just the previous step. Equation (14) also coincides with our conservative vSLAM strategy, because segments are prone to be unstable at the early initialization stage, but after more and more observed segments are integrated in the same segments' map data, it becomes stable as the weighting parameter γ increases. Unlike the updating of ρ and θ in which process we applied a weighting parameter γ for the stability of the map data, the length of segments are updated as per an extending-only policy, regardless of δ.

As we have repeatedly stressed that our vSLAM method is a conservative one, this idea was presented in two more strategies. We restricted our updating range $\mathcal{R}$ for the observed segments. This means that after a segment is extracted from the algorithm in Fig. 5 it was then checked for $\mathcal{R}$, and the parts of the segment that were further than $\mathcal{R}$ were not used in the line extension. We also restricted the updating length $\mathcal{L}$ for the existing segments' map data. This implies that after the extension step, each segment's map data was checked for $\mathcal{L}$, and if it was longer than $\mathcal{L}$, then the corresponding segment was temporarily switched off for several steps for continuous updating. The latter parameter of $\mathcal{L}$ has more importance than $\mathcal{R}$, since if the segment is not correctly initialized for some reason, then the error caused by this segment can propagate for a long period, and may severely distort the final map. From our experimental studies, we chose $\mathcal{R} = 5m$ and $\mathcal{L} = 2\mathcal{R}$.

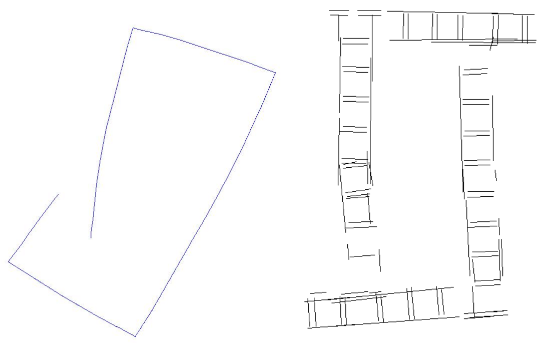
The complete SoF-SLAM algorithm is presented in Fig. 6. This algorithm calls the algorithm of Fig. 5 as a subroutine in step 4 to extract SoF and uses that as the sensory data in the SLAM process.

IV. Experimental Results

A. Experiment

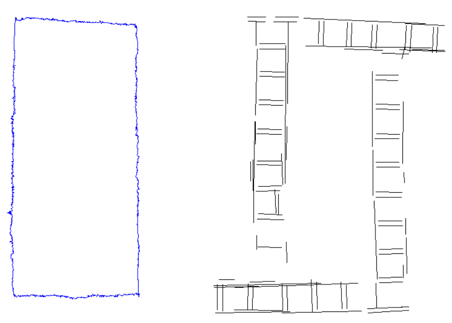
We tested our approach through actual robot experiments in corridor environment, as shown in Fig. 7 (a). The floor is flat and decorated with rectangular black blocks, each block has width $0.45m$ and they are spaced about $2.3m$ apart. A Pioneer 3-DX was used as the real robot platform in the experiment, with one Logitech QuickCam E3500 Webcam mounted on top of the robot as Fig. 7 (b) shows. The camera was placed at a height of 100cm from the floor and faced forward. Images were collected at a resolution of 320×240 pixels from the camera at a frame rate of 10 fps. Before the experiment the Webcam was calibrated with a common checkboard method, and the extracted intrinsic parameters were used in visual data processing. All image segments were extracted using Hough transform at the step 3 of the algorithm of extracting SoF.

As shown in Fig. 7 (a), the robot started from the lower part of the left corridor, and traveled a loop through the inside of the 6th floor of IT Building at Hanyang University. The whole rectangular path is in a dimension of about $11.5m \times 24m$, and the robot was manually driven during the experiment.



(a) Odometer data (b) Initial map

Fig. 8. Comparing (a) and (b), the figure of the initially constructed map is far more accurate than the odometer readings.



(a) Loop closing (b) Final map

Fig. 9. After the loop closing, the trajectory of the robot and the final map are nearly the same as that of Fig 7. (a).

B. Results

The odometer readings were collected during our experiment are plotted in Fig. 8 (a) for reference. The robot was calibrated before the experiment, and accordingly it showed a straight trajectory when it was traveling forward; however, it showed a strong tendency of turning right when changing direction. As a result, the whole path shows a severe trend of spinning inward, and forms a twisted rectangle.

Figure 9 (b) shows the initial resulting map before loop closing. Unlike the figure with the odometer readings, in the resulting SoF-map, the tendency of turning right was not there. The degree of matching between segments of the start and end position is also noticeable. Except for slight mismatch in moving direction, the distance matching is almost exact even after traveling a distance of about $70m$. These accuracies were achieved by two complementary elements: for the segments perpendicular to the robot's direction of motion, these provided the longitudinal guidance for robot motion, which forced the robot to adjust its moving distance during SLAM. The segments parallel to the moving direction of the robot provided lateral guidance for the motion, and caused the robot not to deviate from the correct path.

There were some duplicate lines worth mention. These lines were formed by the top and bottom of baseboards at the foot of the side walls. Since the top of baseboards is near floor plane, they were also extracted as SoF by algorithm of Fig. 6. In most cases we merged these doubled lines into one line, but some were missed when the distance between these lines crossed over a predefined threshold.

We adopted a simple loop closing strategy, after the mismatch error was detected at the end of forward loop, the error was propagated through backward loop, forced it distributed uniformly on the whole path. In the finally constructed map, there were 99 segments formed in the SoF-map. Since each segment only needs a pair of end points which were represented in the form of an $\langle x, y \rangle$ coordinate set, the whole map size is very compact. For the purpose of comparison, the traveling path and the final adjusted SoF-map are shown in Fig. 9 (a) and (b), respectively. The demonstration video clip is available from [10].

V. Conclusions and Future Work

This paper proposed a novel monocular vSLAM method which used SoF as feature data in the SLAM algorithm. The main contribution of our work is to fully exploit the constraint of the SoF. This choice made the extraction of SoF easier, and also simplified the line initialization of the map data from extracted image segments. The experimental results demonstrated the robustness, accuracy and feasibility of our proposed SoF-SLAM in the corridor environment.

We believe that the SoF-SLAM provides a good starting point to building a more precise and linearly presentable indoor map. Our future work is to incorporate all extractable segment components to build a complete 3D indoor map by using only a monocular camera.

References

[1] P. Smith, I. Reid, and A. Davison, "Real-time monocular SLAM with straight lines," *in British Machine Vision Conference*, vol. 1, pp. 17-26, 2006.

[2] E. Eade and T. Drummond, "Edge landmarks in monocular SLAM," *in British Machine Vision Conference*, Edimburgh, Scotland, September 2006.

[3] T. Lemaire and S. Lacroix, "Monocular-vision based SLAM using line segments," in Proc. of IEEE International Conference on Robotics and Automation, Rome, Italy, pp. 2791-2796, 2007.

[4] J. Solà and T. Vidal-Calleja and M. Devy, "Undelayed initialization of line segments in monocular SLAM," *in Proc. of The IEEE/RSJ International Conference on Intelligent Robots and Systems*, October 11-15, St. Louis, USA, 2009.

[5] J. Civera and A. Davison and J. Montiel, "Inverse depth parametrization for monocular SLAM," *IEEE Transactions on Robotics*, vol. 24, no. 5, 2008.

[6] G. Schindler and F. Dellaert, "Atlanta world: An expectation maximization framework for simultaneous low-level edge grouping and camera calibration in complex man-made environments," in *CVPR*, 2004.

[7] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, 2nd Edition, pp. 326-327, 2004.

[8] A. Doucet and N. de Freitas and K. Murphy and S. Russell, "Rao-blackwellised particle filtering for dynamic bayesian networks," in *Uncertainty in Artificial Intelligence (UAI)*, 2000.

[9] A. Eliazar and R. Parr, "DP-SLAM: Fast, Robust Simultaneous Localization and Mapping Without Predetermined Landmarks," *in Proc. of The International Joint Conference on Artificial Intelligence*, 2003.

[10] http://incorl.hanyang.ac.kr/xe/data/sofslam.avi