2013 IEEE International Conference on Robotics and Automation (ICRA)
Karlsruhe, Germany, May 6-10, 2013

Place Recognition using Straight Lines for Vision-based SLAM

Jin Han Lee, Guoxuan Zhang, Jongwoo Lim, and Il Hong Suh

Abstract— **Most visual simultaneous localization and mapping systems use point features as their landmarks and adopt point-based feature descriptors to recognize them. Compared to point landmarks, however, lines have strength in conveying the structural information of the environment. Despite the benefit, they have not been widely used because lines are more difficult in detecting, tracking, and recognizing, and this delayed the use of lines as landmarks. In this paper, we propose a place recognition algorithm using straight line features, which enables reliable loop closure detections in large complex environments under significant illumination changes. A vocabulary tree trained with mean standard-deviation line descriptor is used in finding the candidate matches between keyframes, and a Bayesian filtering framework enables reliable keyframe matching for large-scale loop closures. The proposed algorithm is compared with state-of-the-art point-based methods using scale-invariant feature transform or speeded up robust features. The experimental results show that the proposed method outperforms the others in challenging indoor environments.**

I. Introduction

The objective of simultaneous localization and mapping (SLAM) research is to enable robot systems to map their surrounding environment and localize themselves in the mapped area. One of the main difficulties in visual SLAM methods is the localization of a robot that returns to a previously visited area. This problem is typically solved using loop closure detection, because it can localize the robot by connecting its current position with the map.

We consider two types of loop closure detection. The first type uses geometric information gathered from the SLAM process, and the second uses appearance information gathered from input scenes. Geometry-based approaches often exhibit scalability problems because, during pose estimation, small errors accumulate and eventually become too large to be corrected via geometric reasoning. Therefore, many recent visual SLAM approaches utilize appearance-based loop closure detection techniques [1]–[4].

Currently, most visual SLAM systems use feature points as landmarks. Moreover, they normally perform loop closure detection using feature descriptors, such as the scale-invariant feature transform (SIFT) [5] or speeded-up robust features (SURF) [6] around the points. However, there has been other attempts to use lines as SLAM features [7]–[10], because lines can effectively convey structural information with fewer number of them, as a line spans over a one-dimensional space, rather than a single point in a space (see Fig.1). Despite the benefit of lines, however, they have not been widely adopted because tracking lines are harder than tracking feature points and the loop closure detection using lines is difficult due to the lack of reliable feature descriptors.

Fig. 1. A source image, and two figures containing 189 line features and 1043 SURF features, respectively, extracted from the source image. In this type of structured environment, straight lines are preferred over point features, because they represent structural information more effectively.

To the best of our knowledge, due to the limitations there has been no visual SLAM system with loop closure detection using only line features. Therefore, in this paper, we propose to use lines as main landmarks of a visual SLAM system. We utilize the vocabulary tree presented by Nister et al. [11] that we train for line-to-line matching. Then, for loop closure detection, we apply a Bayesian filtering framework similar to the one presented by Angeli et al. [3], which we modify to consider appearance similarities between scenes in state transitions. We show that a vocabulary tree built with line descriptors works better than the trees built with state-of-the-art point descriptors in structured indoor environments, and the potential of using line descriptors in practical vision-based SLAM applications. As a line descriptor, we adopt the mean standard-deviation line descriptor (MSLD) proposed by Wang et al. [12]. Because MSLD represents a line with statistics from its neighboring region, it is invariant to the locations of its endpoints. However, it is not invariant to scale changes.

The main contributions of this paper are as follows:

- An implementation of a vocabulary tree built with MSLD, and experimental comparisons of this tree with others built with SURF and SIFT.

Jin Han Lee and Guoxuan Zhang are with the Department of Electronics and Computer Engineering, Hanyang University, Seoul, Korea. {jhlee, imzgx}@incorl.hanyang.ac.kr.

Jongwoo Lim and Il Hong Suh are with the Division of Computer Science and Engineering, College of Engineering, Hanyang University, Seoul, Korea. {jlim, ihsuh}@hanyang.ac.kr. All correspondences should be addressed to Il Hong Suh.

- A real-time implementation and experimental validation of a loop closure detection algorithm that uses only line features

The remainder of this paper is organized as follows. Section II describes algorithms for keyframe matching using a vocabulary tree. Section III presents a Bayesian filtering algorithm used for incremental database construction. In Section IV, we compare the vocabulary tree trained with line descriptors with other trees trained with SIFT and SURF. Further, in this section, we present results obtained from experiments conducted in two structured indoor environments containing several low-textured regions. This paper concludes in Section V.

II. Quantized Scene Representation

A. Line Extraction and Description

In order to extract line segments from input images, we adopt a method presented by Bay et al. [13]. Canny edges are detected first, then straight lines are fit to the edges. Only lines longer than 30 pixels are accepted, and line descriptors for the line segments are generated using MSLD. The MSLD first identifies the perpendicular direction $\mathbf{d}_\perp$ (*i.e.,* the average gradient direction) and parallel direction $\mathbf{d}_\parallel$ (*i.e.*, turned anticlockwise from $\mathbf{d}_\perp$) of each line. Let l be the length of the line. Then, for l pixels on the line, it sets c subregions (each with a size of $r \times r$) along to $\mathbf{d}_\perp$ in a non-overlapping manner. Thus, it results in a total of $c \times l$ subregions overlapped along the $\mathbf{d}_\parallel$. In this work, we adopt the settings used in [12], *i.e.,* $c = 9$ and $r = 5$. In each subregion, accumulating distributed gradients along the direction $\mathbf{d}_\perp$, $\mathbf{d}_\parallel$ and their opposite directions results 4-dimensional descriptor vector for the subregion. Then, by calculating the mean and standard deviation of the vectors along the direction $\mathbf{d}_\parallel$, MSLD can represent different-length lines with uniform $(4 + 4) \times 9 = 72$ dimensional vectors. This allows using the MSLD directly with the vocabulary tree. Additional procedures can be applied to increase the robustness of the descriptors (please refer to [12] for more details).

B. Vocabulary Tree

The vocabulary tree approach [11] is one of the most popular algorithms used for scene recognition. This approach uses the bag-of-words methods for visual features. It hierarchically divides the feature space by clustering the feature descriptors in the training set. Then, using the cluster centers as nodes, it constructs a tree. The nodes of the tree represent different classes of descriptors. Hence, the nodes represent a quantized representation of the space of descriptors. Thus, a set of lines detected from a scene can be represented by a set of nodes. One of the advantages of this method is that it enables on-the-fly database insertion and querying when it is used with an inverted file mechanism.

In order to build a vocabulary tree for MSLD features, we extracted four million MSLD descriptors as the training set from four documentary videos about historical buildings in Europe. Then, with the training set we performed hierarchical k-means clustering with branching factor $k = 40$, and number of levels $l = 3$, resulting a tree with 65641 nodes. In our implementation, however, we used only 64000 leaf nodes following the analysis by the authors of [11]. In Section IV-A, we experimentally evaluate the retrieval performance of the tree.

When an image is registered in the database, each descriptor vector of a line in the input image traverses through the tree until it reaches a leaf node. The ID of the image is then listed in the node. Similarly, when querying a scene, also every descriptor vector in the query image traverses through the tree, then the IDs in the leaf node represent potential candidate matches and receive votes. In this voting scheme, we use the normalized difference with term frequency-inverted document frequency (TF-IDF) weighting [14] in the L_1-norm [11]. Because the scores are a measure of distance, the best hypothesis receives a score of zero.

However, because we use these scores to compute likelihoods, we need scores with higher values for close matches in the database to queries. Therefore, we slightly modified the original scoring scheme. We define query $\mathbf{q}$ and database $\mathbf{d}$ vectors as:

$$q_k = n_k w_k, \tag{1}$$

$$d_k = m_k w_k, \tag{2}$$

where,

$$n_k = \frac{\text{number of word } k}{\text{number of total words in query scene}} \tag{3}$$

is the term frequency of word k in the query image and,

$$m_k = \frac{\text{number of word } k}{\text{number of total words in the database scene}} \tag{4}$$

is the term frequency of word k in a database image. Moreover, w_k is the inverted-document frequency, expressed as:

$$w_k = \ln \frac{N}{N_k}, \tag{5}$$

where N is the total number of database images and N_k is the number of database images containing the word k. Then, if the word k is observed in the query scene, the score assigned to database image i is:

$$s_i = - \sum_{k | q_k \neq 0, d_k \neq 0} \big(|q_k - d_k| - q_k - d_k \big). \tag{6}$$

Note that we need to normalize vectors $\mathbf{q}, \mathbf{d}$ beforehand. In this scoring scheme, the words with high "term frequency" (*i.e.,* they frequently appear in the image) receive higher scores. Meanwhile, words with high "inverted-document frequency" (*i.e.,* they also frequently appear in other images) are penalized. Finally, images in a database that match well to a query receive higher scores.

III. Bayesian Filtering Framework

We use a Bayesian filtering framework to increase the tolerance of our method to noisy responses generated from the raw matching scores. In addition, the Bayesian filtering framework enables the system to insert input scenes into the database in an incremental fashion. Hence, if a scene is very similar to a registered scene, it is not added to the database. Otherwise, the scene is considered as from a new part of the environment, it is added to the database. Our proposed algorithm is similar to the one presented in [3]. However, in order to adopt and improve the method for our objective the following ideas are adopted.

- Instead of using bags-of-words with SIFT and color histogram, we utilize vocabulary tree with line descriptors for scene representation and scoring in likelihood calculation.
- In state transition modeling, we consider appearance similarity for faster probability diffusion between scene recognition hypotheses in terms of both the distance in time (*i.e.,* the difference of sequential orders between two hypotheses) and the distance in the space of appearance (*i.e.,* the score of appearance similarity between two hypotheses).

A. Discrete Bayesian Filtering

Under the complete state assumption, or the Markov assumption, the discrete Bayesian filter in our work is derived as follows. Given measurements up to the current time t, $z_{0:t}$, the full posterior probability of state x_t is expressed as:

$$p\big(x_t|z_{0:t}\big) = \eta p\big(z_t|x_t\big)p\big(x_t|z_{0:t-1}\big). \tag{7}$$

By expanding $p(x_t|z_{0:t-1})$ using the *Theorem of total probability*, we get

$$p\big(x_t|z_{0:t}\big) = \eta p\big(z_t|x_t\big)\sum_i p\big(x_t|x_{t-1}=i\big)p\big(x_{t-1}=i|z_{0:t-1}\big), \tag{8}$$

where $p\big(x_{t-1}|z_{0:t-1}\big)$ is the prior (*i.e.,* the posterior probability at the previous step), $p\big(z_t|x_t\big)$ is the likelihood of the current hypothesis, and $p\big(x_t|x_{t-1}\big)$ is the state transition probability. Because the state x_t is a random variable representing the scene recognition hypotheses and the measurement z is a quantized representation of the scene, $z_{0:t-1}$ denotes the database scenes in the tree and the z_t denotes the queried input scene quantized by the tree structure. Practically, the recent n scenes are excluded from the hypothesis set $\mathbf{x}$, because we are interested in finding long-ranged scene matches, *i.e.,* $i \in \big\{-1, 0, ..., t-n\big\}$.

According to [3], the state $x_t = -1$ is used to indicate that at step t the input scene has not been recognized. For the hypothesis, we construct a virtual scene consisting of the m most frequent words in the tree. Specifically, we construct the tree by assigning to the corresponding leaf nodes an ID of -1. Thus, if the input scene has not been previously observed, we assume that it will be more similar to the virtual scene. In our implementation we set the m equal to the current average number of features per image frame and $n = 40$. This n should be adjusted according to the velocity and frame rate of the camera.

B. State Transitions between Hypotheses over Time

We use a Gaussian function with a standard deviation σ_{trans} to assign higher probability transitions from one hypothesis to another that has a closer distance in time. In typical filtering algorithms, the probability of each hypothesis is only diffused to other hypotheses that are close to it. This increases the tolerance against noise, thus preventing transient errors and enabling smooth state transitions over time. However, in Fig.4, at the point marked with "x", although the two scenes appear similar (*i.e.,* they are closely located in the space of appearance), they cannot diffuse their probabilities to each other because they are not closely located in time. To address this situation, we model the distance in the space of appearance between two hypotheses, and consider it when computing state transitions. Therefore, in the probability evolution, we consider both the temporal and spatial relations. The state transition model is defined as follows:

- $p\big(x_t = -1|x_{t-1} = -1\big) = \alpha$ represents the probability that at current step t no scene has been recognized if there was no recognized scene at previous step $t-1$. In this work, we set the $\alpha = 0.9$.
- $p\big(x_t = i|x_{t-1} = -1\big) = \frac{1-\alpha}{t-n+1}$, where $i \in \big\{0, ..., t-n\big\}$, represents the probability that at current step t a scene has been recognized with hypothesis i when there was no recognized scene at previous step $t-1$. In this case, the probabilities are uniformly distributed over $t-n+1$ hypotheses.
- $p\big(x_t = -1|x_{t-1} = i\big) = 1-\alpha$, where $i \in \big\{0, ..., t-n\big\}$, represents the probability that there is no recognized scene at current step t is low if there was a recognized scene at previous step $t-1$.
- $p\big(x_t = i|x_{t-1} = j\big) = max\{0.01, G\big(|i-j|\big)\} \times sim(i,j)$, where $i, j \in \big\{0, ..., t-n\big\}$, $G\big(|i-j|\big)$ is a value on the distance $|i-j|$ of the zero-mean normal distribution with standard deviation σ_{trans} (we empirically set $\sigma_{trans} = 2$, and it should be adjusted according to the velocity and frame rate of the camera), and $sim(i,j)$ is a measurement of appearance similarity between two hypotheses i and j that use only the term frequency of each word. Therefore,

$$sim(i,j) = -\sum_{k|h_{i,k}\neq 0, h_{j,k}\neq 0}\big(|h_{i,k}-h_{j,k}|-h_{i,k}-h_{j,k}\big), \tag{9}$$

where,

$$h_{l,k} = \frac{\text{number of word } k \text{ in hypothesis } l}{\text{number of total words in hypothesis } l} \tag{10}$$

is the term frequency of word k in hypothesis l.

Here, the computation of the appearance similarity is different from the scoring computations performed for database images (presented in Section II-B), because the former is done between database images and the latter is performed

between a query image and the database images. Note that, under the same condition in the last case, the probabilities of all hypotheses have to sum up to α. We only compute $sim(i,j)$ whenever a new scene is inserted into the database, involving the new scene and each of $t-n+1$ database scenes.

C. Likelihood Calculation

We calculate likelihoods using the method presented in [3] with a modification. The original method selects hypotheses with scores higher than $\mu+\sigma$, where μ and σ are the mean and standard deviation of the scores, respectively. In this work, we use a more strict condition to select hypotheses. Specifically, we select hypotheses with scores higher than $\mu+2\sigma$ in order to give dominant probabilities to the selected hypotheses. Therefore, the likelihood for hypothesis i with score s_i is defined as:

$$L(x_t = i|z_t) = \begin{cases} \frac{s_i - 2\sigma}{\mu} & \text{if } s_i > \mu + 2\sigma \\ 1 & \text{otherwise} \end{cases} . \quad (11)$$

We use this likelihood function in place of the conditional probability $p(z_t|x_t)$ defined in Equation (8). After the measurement update with likelihoods, the full posteriors have to be normalized.

D. Selecting a Hypothesis

When the camera re-enters a previously visited area, several hypotheses in consecutive orders will receive relatively high probabilities. Therefore, it is reasonable to search for a subset of consecutive hypotheses whose sum of probabilities is over a threshold. If we find a subset satisfying this condition, then we select the hypothesis in the subset with the maximum probability as a recognized scene. Here, we empirically set the size of the subset to 10, and the threshold to 0.6.

IV. Experimental Results

In this section, we present the results obtained from experiments conducted to evaluate the performance of our proposed method. The image sequences used for the experiments were gathered using a Point Grey Bumblebee stereo rig camera (BB2-08S2C-38) and a Canon DSLR camera (EOS-600D with a bundle lens, EF-S 18-55mm f/3.5-5.6 IS II). All optical distortions in the images were removed before conducting the experiments. In our first experiment, we compare the performance of a tree trained using MSLD with other trees trained with SIFT and SURF. Afterwards, we conduct three experiments using the Bayesian filtering framework. In these experiments, if the number of features contained in an image was less than half of the average number of features per image, then it was discarded. We applied this rule because, in the TF-IDF scheme, if an image contains a small number of features, then words in the image receive overly high weights. All three experiments were performed in real-time using an Intel i7-2600K processor. The trajectories presented in this section are all estimates, because information on the actual position and orientation of the camera was not available during sequence acquisitions. A

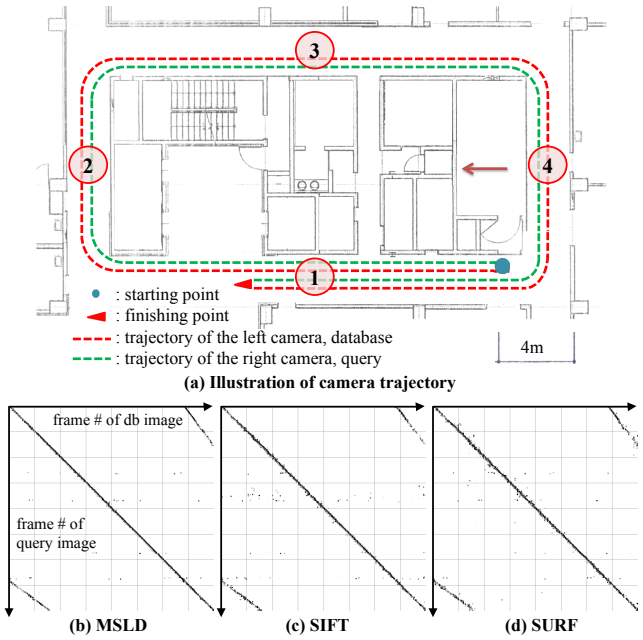


Fig. 2. The figure (a) shows the trajectories followed to acquire the image sequence used for the evaluation. Note that there was an overlap in the last part of the trajectory. The bottom plots show the results. Using the image sequence, we evaluated retrieval performances of vocabulary trees trained with (b) MSLD, (c) SURF, and (d) SIFT. The horizontal and vertical axes represent the frame numbers of database and query, respectively. The dots in the plots represent the top five scores. The performances measured were 98.97% for MSLD, 96.49% for SIFT, and 95.05% for SURF. The small lines symmetrically located in the upper right and lower left parts in each figure correspond to the overlapped area. The numbers in the circles indicate the circled areas and numbered image pairs presented in Fig. 3.

video of the results is available at http://youtu.be/dgbmTI9S-Ko?hd=1.

A. Evaluation of the Built Vocabulary Tree

A key part of our work is the vocabulary tree trained using line descriptors. Therefore, we evaluate its retrieval performance using image-pairs gathered by a robot-equipped stereo rig. We assign left images as database images and right images as query images, assuming that there is an exact correspondence (*i.e.,* ground truth) between left and right images. The image-pair sequence used for testing was captured as the camera traversed a loop in 3rd floor of the IT/BT building at Hanyang University in Seoul, Korea (at the end of the loop, the camera traveled approximately 12 additional meters although the loop had already been completed). The environment contained structured features, as well as a small number of distinctive features. After constructing the database using 485 left images, we sequentially queried every right images. In Fig. 2-(b), we plot the results obtained. The horizontal and vertical axes represent the frame numbers of the database and query images, respectively. The top five results of the query are represented by points darkened according to their scores. In this plot, we observe a diagonal, linear distribution of points. The shorter lines that are symmetrically located on either side of the plot are points in the database corresponding to the revisited area in

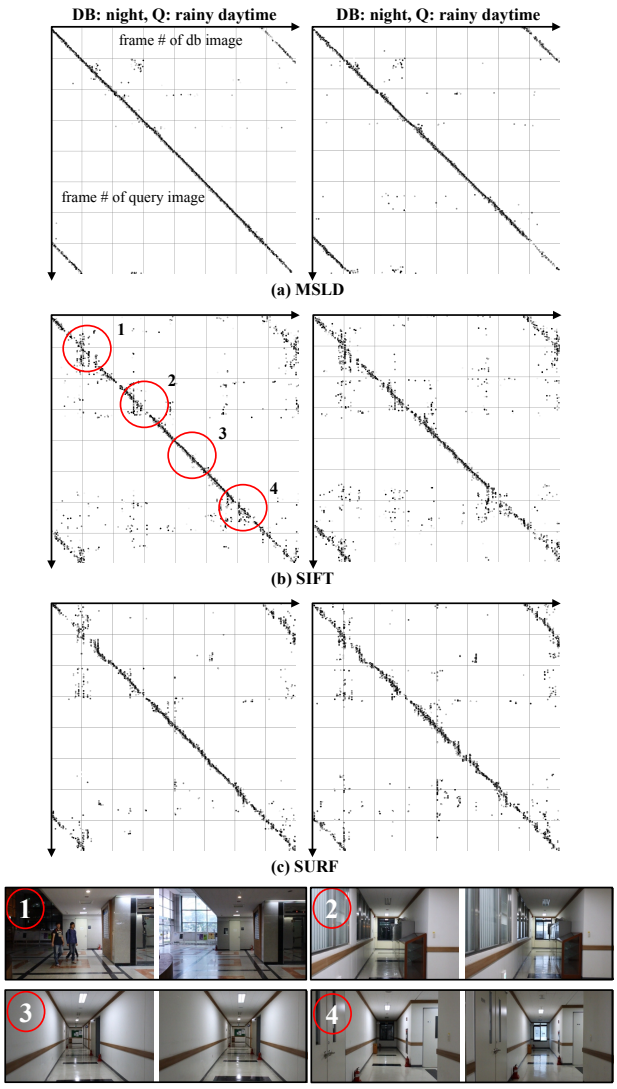


Fig. 3. The plots show results obtained under illumination changes using the vocabulary tree trained with (a) MSLD, (b) SIFT, and (c) SURF. The numbered image pairs in (d) are scenes obtained from the corresponding locations on the trajectory marked with red circles in Fig. 2. In pairs of (d), scenes from nighttime are on the left and scenes from daytime under rainy condition are on the right.

constructing the database. A query is considered successful if at least one of the top five results returned was not farther than two frames from the ground truth. The measured successful retrieval rate was 98.97% (*i.e.,* $480/485$).

For comparison purposes, we conducted the same experiments using the SURF and SIFT point features. The only difference in these experiments is the vocabulary trees used. We built another two vocabulary trees with SURF and SIFT descriptors, respectively, with the same settings (*i.e.,* $k = 40, l = 3$, hierarchical k-means clustering). The trees were also trained with 4 million descriptors extracted from the same videos used to obtain the training dataset for the MSLD descriptors. We used standard implementations for SIFT and SURF (*i.e.,* from Willow Garage's OpenCV library

TABLE I

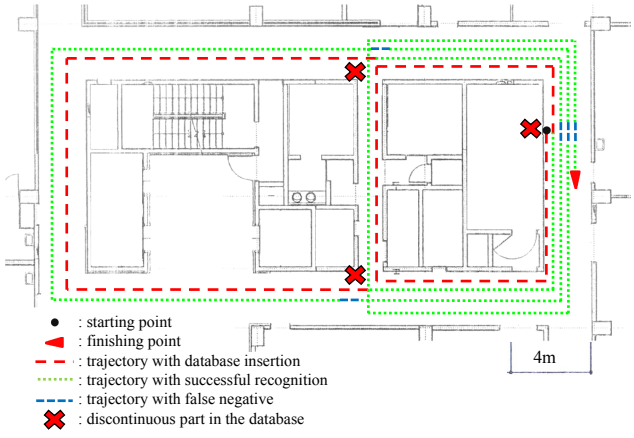
EXPERIMENTAL EVALUATIONS OF THE TREES

	MSLD	SIFT	SURF
hessian	.	100	100
line length	30	.	.
stereo			
# database	485	485	485
# success return	480	468	461
% success	98.97 %	96.49 %	95.05 %
# feature	91,448	237,558	620,651
illumination change 1 (DB: night, Q: rainy daytime)			
# success return	409 / 453	304 / 453	288 / 453
% success	90.29 %	67.11 %	63.58 %
# feature	93,313	179,683	617,884
illumination change 2 (DB: night, Q: sunny daytime)			
# success return	433 / 507	308 / 507	337 / 507
% success	85.40 %	60.75 %	66.47 %
# feature	98,994	192,401	653,584

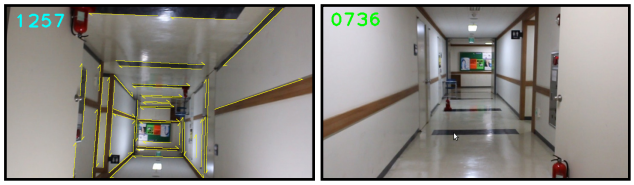
From top to bottom, the rows indicate **hessian**: hessian threshold for SIFT and SURF, **line length**: acceptance length threshold in line extraction, **# database**: number of inserted database images, **# success return**: number of successful retrievals, % **success:** successful retrievals in %, # **feature**: number of features processed in total.

available at [15]). In Figs. 2-(c) and (d), we present the results of the evaluations. We observe that, in these cases, the distribution of points along the diagonal is spread more than in the case of the results obtained by the tree trained with MSLDs. By applying the same rule, the successful retrieval rates measured using these two trees were 96.49% for SIFT (468/485) and 95.05% for SURF (461/485), respectively.

Fig. 3 shows other evaluation results obtained for the vocabulary trees under illumination changes. In these experiments, we used three image sequences acquired in the same environment but under different illumination conditions, using the Canon camera equipped on a robot (in these sequence acquisitions, we tried to maintain a constant velocity for the robot, and follow the trajectory traversed with the stereo rig in the previous evaluation). We constructed the database using the image sequence captured at night. The other two image sequences were captured during daytime under sunny and rainy conditions, and were used as queries. The plots in Fig. 3 show the results obtained from the vocabulary trees trained with (a) MSLD, (b) SIFT, and (c) SURF. The results depicted in the left column were obtained using the database scenes acquired during nighttime and query scenes acquired during daytime under rainy conditions (thus, they had a relatively small difference in illumination). The results depicted in the right column were obtained using query scenes acquired during daytime under sunny conditions (thus, they had a stronger difference in illumination). Because it was also difficult to obtain the actual trajectories of the robot, we approximated the retrieval performance by first adjusting the scale of the vertical axis to the scale of the horizontal



(a) Illustration of camera trajectory



(b) An example of successful recognition

Fig. 4. (a) Trajectory followed by the camera on the 3rd floor of the IT/BT building at Hanyang University. The three discontinuous parts in the database sequence are marked with "x". The system successfully recognized all scenes, except those from sections with false negatives. (b) An image pair showing an example of a successful recognition of a place where the query image is rotated approximately 180 degrees along the optical axis.

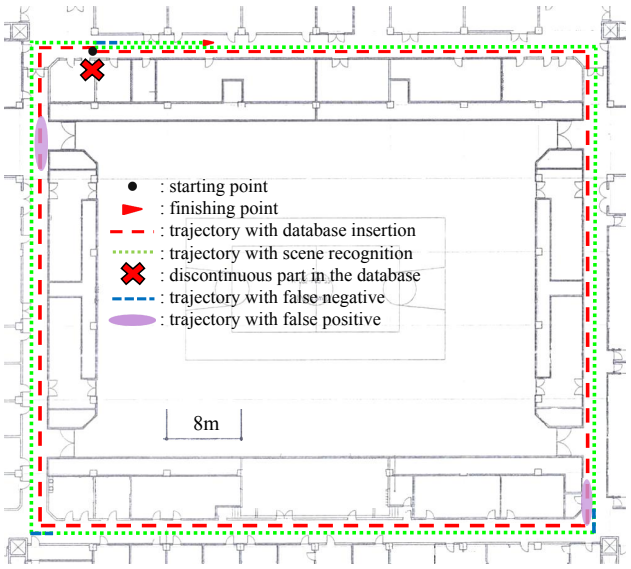


Fig. 5. Trajectory followed by the camera on the 3rd floor of the Olympic Gymnasium at Hanyang University. During the first exploration of the loop, 30 frames of false positives occurred in the trajectory (marked with purple ellipses). After the camera completed the loop, which was approximately 230 meters long, the system continuously recognized scenes (except for only 6 frames of false negatives) until it reached the discontinuous part in the database. Examples of successful recognition and false positives in this experiment are shown in Fig. 6.

axis, then applying the same rule in the previous evaluations. From the plots, we observe that the results obtained from the vocabulary tree built with MSLDs deviate less from the diagonal line than the other trees. In locations 1, 2, and 4, marked with red circles in the Fig. 3-(b), the distributions of the points generated by the trees trained with SIFT and SURF have more spread than those obtained by the tree trained with MSLD. This result is obtained because the scenes at the corresponding areas were affected by natural light.

The approximated retrieval performances for rainy and sunny illumination conditions were 90.29% and 85.40%, respectively. As shown in Fig. 3-(d), the tested environment was very structured (thus advantageous for straight lines) and had numerous low-textured regions (thus disadvantageous for keypoint-based features). Therefore, it is more reasonable to attribute these results to the experimented environment and not the performances of SIFT or SURF. In Table I, we present the results of the evaluations.

B. Indoor Experiment with a Complex Loop Configuration

The image sequence used in this experiment was captured while the hand-held Canon camera repeatedly looped the same environment of the previous evaluations. In Fig. 4-(a), we plot the camera trajectory used in this experiment. The red dashed line represents the trajectory used for database insertion, the green dotted line represents a trajectory with scene recognition, and the blue dashed line represents a trajectory with false negatives. From this figure, we observe that after the camera completed the small loop, the system began recognizing scenes, because they were acquired from areas that had been previously visited. When the camera left the area, the system restarted database insertion and continued until the camera faced another previously visited area (close to the top "x" marked in the floor plan). From this point on, the system recognized scenes without generating false positives. Only 26 frames of false negatives were generated near the points marked with "x", which are used to indicate the discontinuous parts in the database. These discontinuities occur due to our database insertion strategy, *i.e.,* our system inserts scenes into the database as a function of time. Therefore, some parts such as those points indicated with "x" may have a broken junction. Consequently, this prevents probability diffusion between two scenes even though they are obtained from close locations. As mentioned in Section III-B, we attempted to overcome this problem by considering a distance metric that accounts for both space and time. Hence, in this experiment, the system generated only 26 frames of false negatives and recognized 1145 scenes successfully. In addition, to evaluate the modified state transition model, we conducted an another experiment with replacing $p(x_t = i|x_{t-1} = j) = G(|i-j|) \times sim(i,j)$ to $p(x_t = i|x_{t-1} = j) = G(|i-j|)$. The system generated 38 false negatives and recognized 1118 scenes. We observe that the system generates less false negatives and starts recognizing scenes earlier with our modified transition model. The results are shown in columns of indoor 1-(a) and (b) in Table II.

(a) Examples of successful recognitions

(b) Examples of false positives

Fig. 6. Examples of successful recognition and false positives in the experiments in Section IV-C. In each image pair, left image is a queried scene and right image is the recognized scene from the database.

C. Indoor Experiment while Traversing a 230-Meter-Long Loop Twice

In this experiment, we used an image sequence captured with the camera equipped on a robot, while moving along a corridor of the Olympic Gymnasium at Hanyang University. The robot traversed the 230-meter-long loop twice. In Fig. 5, we plot the trajectory using the same representation used in Fig. 4. Fig. 6-(a) shows examples of successful recognition where the scenes contained regions that were occluded by people. In this experiment, however, the system generated 30 frames of false positives in two areas. Fig. 6-(b) shows examples of the false positives in each area. In the pairs of images, we can see strong structural similarities between the scenes. However, when we adjusted the threshold of the line length in extraction from 30 to 50, we avoided line extractions from repeated patterns on the floors in the environment and consequently no false positives occurred. Instead, greater number of false negatives occurred, because a smaller number of features were available to distinguish between scenes.

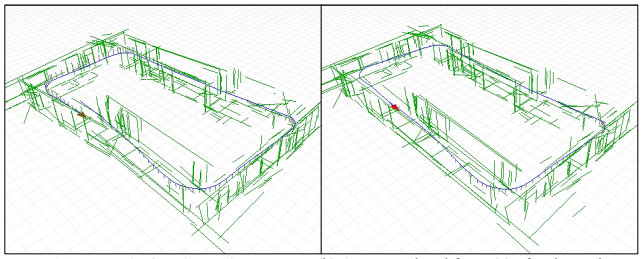
Fig. 7. Examples showing successful scene recognition under illumination changes. Left images are queried scenes and right images are the returned recognized scenes.

D. Indoor Experiment under Illumination Changes

In Section IV-A, we tested the retrieval performance of the vocabulary tree under different illumination conditions. From the results obtained, we saw that the tree trained with MSLDs exhibited the least noisy response compared to the other trees. Therefore, we tested our proposed method using the two image sequences that were also used for the evaluation in Section IV-A. First, the system constructed the scene database using the image sequence acquired during nighttime. Then, we used as queries the other image sequence, which were gathered during daytime under rainy conditions. Although the sequences were captured in an indoor environment, there were three areas strongly affected by the sun-light coming through the windows, as shown in Fig. 3-(d). Among the 447 query scenes, 422 scenes were successfully recognized, and no false positives occurred. Fig. 7 shows examples of scene recognition results obtained from this experiment.

E. Performance and Integration into a SLAM System

Table. II shows the parameter settings used in each experiment and the performances obtained. The average times consumed by the system for each frame was approximately 50 ms. Most of the time was consumed to extract lines and generate descriptors. We believe that the time required to



(a) A map before loop closure (b) A map updated from (a) after loop closure

Fig. 8. Loop closure result using the proposed method in a SLAM system

TABLE II

PARAMETER SETTINGS AND PERFORMANCES

	indoor 1		indoor 2		illum change
	(a)	(b)			
resolution	720×405		640×480		720×405
line length	50		30	50	30
n	40				
σ_{trans}	2				
α	0.9				
Threshold	0.6				
subset size	10				
# database	602	629	895	914	418
# recog	1145	1118	988	912	422 / 447
# falsepos	0		30	0	0
# falseneg	26	38	6	19	25
# total	1747		1883	1826	865
Avgtime [ms]	49.45	42.93	69.56	56.87	47.33
avgtime line [ms]	34.11	33.89	45.89	33.58	38.82
# msld	93,238		175,464	93,258	93,027

Columns indoor 1-(a), (b) indicate the settings and performances with/without our modified state transition model, respectively. The rows indicate **resolution**: image resolution, **line length**: acceptance length threshold in line extraction, n: number of excluded recent scenes, σ_{trans}: standard deviation in state transition modeling, α: parameter of no scene recognition probability at current step t if there was no scene recognition at previous step t-1, in the state transition model, **threshold**: threshold in searching a subset of hypotheses, **subset size**: size of subset in searching, # **database**: number of database images inserted, # **recog**: number of recognized images, # **falsepos**: number of false positives, # **falseneg**: number of false negatives, # **total**: number of images used in total, **avgtime**: average processing time per input image, **avgtime line**: average processing time for line extraction and description per input image, # **msld**: number of MSLD descriptors extracted in total.

perform the extraction and generation of descriptors for lines will be reduced if we utilize graphics processing units.

In addition, we integrated our proposed method into a SLAM system that we are currently working on. In Fig.8, we present a result obtained from this integrated system. When this integrated system processes SLAM, in every keyframe insertion it attempts to detect loop closures using only our proposed method, and it results effective loop closures.

V. CONCLUSION

In this work, we proposed a place recognition algorithm using straight-line features. We adopted a vocabulary tree to effectively represent scenes using mean standard-deviation line descriptors (MSLD). Following this, we evaluated the retrieval performance of the tree by comparing with other trees trained with the point feature descriptors of speeded-up robust features (SURF) and scale-invariant feature transforms (SIFT). In addition, we adopted a Bayesian filtering framework to reliably detect loop closures using raw scores returned from the vocabulary tree. Moreover, the Bayesian filtering framework enabled the system to construct the database in an incremental fashion. Our proposed algorithm successfully recognizes scenes in a complex loop configuration, a large-scale loop, and under illumination changes. We have also presented a result obtained from integration of the proposed method into a SLAM system. Because the evaluations and experiments reported in this work are limited to indoor environments, we plan to extend our method to outdoor environments.

ACKNOWLEDGMENT

This research was supported by the Global Frontier R&D Program on "Human-centered Interaction for Coexistence" funded by the National Research Foundation of Korea grant funded by the Korean Government (MEST) (NRF-M1AXA003- 2011-0028353). This work was also supported by the Industrial Strategic Technology Development Program (10044009) funded by the Ministry of Knowledge Economy (MKE, Korea).

REFERENCES

[1] K. Konolige, J. Bowman, J. D. Chen, P. Mihelich, M. Calonder, V. Lepetit, and P. Fua, "View-based Maps," *The International Journal of Robotics Research (IJRR)*, vol.29, no.8, pp.941-957, July 2010.

[2] M. Cummins and P. Newman, "FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance," *The International Journal of Robotics Research (IJRR)*, vol.27, no.6, pp.647-665, June 2008.

[3] A. Angeli, D. Filliat, S. Doncieux, and J. -A. Meyer, "Fast and Incremental Method for Loop-Closure Detection using Bags of Visual Words," *IEEE Transactions on Robotics (TRo)*, vol.24, no.5, pp.1027-1037, Oct. 2008.

[4] D. Filliat, "A Visual Bag of Words Method for Interactive Qualitative Localization and Mapping," *in Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, pp.3921-3926, April 2007.

[5] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision (IJCV)*, vol.60, no.2, pp.91-110, Nov. 2004.

[6] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "SURF: Speeded-Up Robust Features," *Computer Vision - ECCV 2006*, vol.3951, pp.404-417, Jan. 2006.

[7] R. I. Hartley, "A Linear Method for Reconstruction from Lines and Points," *in Proc. of the Fifth International Conference on Computer Vision (ICCV)*, pp.882-887, June 1995.

[8] G. Klein, D. Murray, "Improving the Agility of Keyframe-based SLAM," *in Proc. of the tenth European Conference on Computer Vision: Part II (ECCV)*", pp.802-815, Jan. 2008.

[9] M. Chandraker, J. Lim, and D. Kriegman, "Moving in Stereo: Efficient Structure and Motion using Lines," *in Proc. of the IEEE 12th International Conference on Computer Vision (ICCV)*, pp.1741-1748, Sept.29 2009-Oct. 2 2009.

[10] G. Zhang and I. H. Suh, "A Vertical and Floor Line-based Monocular SLAM System for Corridor Environments," *International Journal of Control, Automation and Systems (IJCAS)*, vol.10, no., pp.547-557, June 2012.

[11] D. Nister and H. Stewenius, "Scalable Recognition with a Vocabulary Tree," *in Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, vol.2, no., pp.2161-2168, June 2006.

[12] Z. Wang, F. Wu, and Z. Hu, "MSLD: A Robust Descriptor for Line Matching," *Pattern Recognition*, vol.42, no.5, pp.941-953, May 2009.

[13] H. Bay, V. Ferraris, and L. Van Gool, "Wide-Baseline Stereo Matching with Line Segments," *in Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol.1, no., pp.329-336, June 2005.

[14] J. Sivic and A. Zisserman, "Video Google: a Text Retrieval Approach to Object Matching in Videos," *in Proc. of the IEEE 9th International Conference on Computer Vision (ICCV)*, vol.2, no., pp.1470-1477, Oct. 2003.

[15] http://opencv.willowgarage.com/wiki/