Advances in Robotics & Automation
ISSN: 2168-9695

Shuai Yang et al., Adv Robot Autom 2015, S2:1
DOI: 10.4172/2168-9695.1000S2-008

Research Article Open Access

Place Recognition using Multiple Feature Types

Shuai Yang*, Wei Mou and Han Wang

School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798, Singapore

***Corresponding author:** Shuai Yang, School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798, Singapore, Tel: +656790 6699; E-mail: sayang@ntu.edu.sg

Received date: Nov 13, 2015; **Accepted date:** Dec 11, 2015; **Published date:** Dec 21, 2015

Abstract

Place recognition has been intensively studied in the context of robot vision. BoW-based approach gains its popularity for its efficiency and robustness using features extracted from images. Many features have been examined in the past for place recognition purpose. However, there is no such feature that can outperform others in all environments. Each feature has its own advantage, thus, they should be carefully chosen depending on the context and environments. In this paper, we propose a modified vocabulary tree with the ability of merging multiple kinds of features such that it allows users to customize different combination of features for better place recognition performance. The system is tested in real-time on real-world datasets and the experiments demonstrates the advantage of our system compared to existing approaches.

Keywords: Place recognition; Modified vocabulary tree; Multiple feature types

Introduction

Localization is a key issue for the autonomous navigation of robots. The location and orientation of a robot can be estimated incrementally from robot motion between current and previous states. Recent years, navigation based on visual sensors has drawn great attention due to its low cost and possibility to provide a full 6-DOF motion estimation. However, with this kind approach, the small errors of motion estimation will accumulate over time and result in drift on localization results. SLAM is the process of estimating robot pose and building map at the same time which can correct accumulated drift. When a place has been revisited, the accumulated error can be determined and corrected. Hence, the ability of recognizing a pre-visited place or the ability of place recognition is critical for robot localization as it can be used to resolve data association issues in SLAM.

Vision-based place recognition problems can be considered as that, given a query image captured at a particular place, it is desired to return images that depict the same place from the database. Efficient place recognition methods often build on the bag-of-words representation developed for object and image retrieval. Each image is represented by a vector of visual words. The visual words are created by extracting descriptors of local features from images in the database and clustering them into a vocabulary. When querying an image, the visual words are extracted from the query image using the pre-built vocabulary and the images are returned from database according to the similarity of visual words.

Most current visual place recognition approaches use point features, such as scale-invariant feature transform (SIFT) or oriented FAST and rotated BRIEF, to represent a scene. One of the most well-known approach is Fast Appearance-Based Mapping (FAB-MAP) [1-3] which can perform very large trajectory estimation based on bag of visual words model using SIFT features. Zamir et al. [4] proposed a method, which utilizes Google Street View images as the reference dataset. They extract local features (e.g., SIFT) from the query image and retrieve a number of nearest neighbors for each query feature from the reference dataset, then a feature pruning method which incorporates geo-spatial information is em-ployed to discover incorrectly matched features. Valgren et al. [5] evaluated outdoor appearance-based topological localization for a mobile robot over seasons using SIFT and SURF features. Badino et al. [6] proposed a system, dubbed topometric localization method: at the first step, images are recorded and image features (standard SURF) describing each location of the mapping route are extracted from the images and stored as a map. During localization stage, current image features are matched with the stored dataset and position estimates are smoothed by fusing velocity information.

Besides point features, there are other popular types of features such as line features that are used in different applications [7,8]. Compared to point features, line features carries more structural information since each of them is spanned over a 2D space instead of a single point. Moreover, they are more robust to environmental changes such as illumination, viewing direction, or occlusion. The authors proposed a place recognition approach using line features and demonstrated better performance than using point features in well-structured environments [9].

In an indoor environment, many visual navigation approach based on ceiling vision [10,11] has been proposed in the past. Ceiling vision has an advantage of no or little obstruction, as well as only involves rotation and affine transformation without scale changes compared to the frontal view. In our previous work [12], we use lines extracted from ceiling images to calculate the robot motion and use the ceiling as an absolute reference to determine the robot's global orientation. Although our motion estimation approach demonstrates robustness on disturbance, the place recognition using ceiling vision can be even more challenging due to lack of point features and similar line patterns of ceiling images captured at different places. As can be seen in Figure 1, lines are the dominant patterns in both images. Since they are similar in both images, it is difficult to distinguish both places using only line features. On the other hand, point features are too few (about

20 FAST features in each image) to generate a meaningful set of visual words to describe an image properly, however, they can provide extra distinctiveness to find correct matchings in a database. A good combination of both kind of features should provide a better place recognition result.

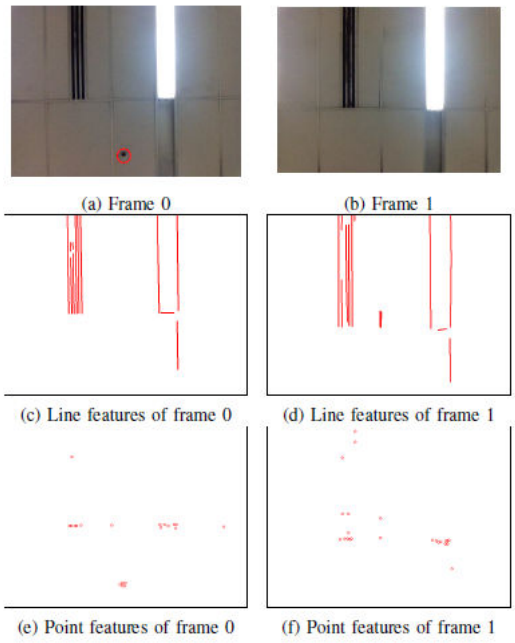


Figure 1: Point and line features extracted from ceiling images. Frame 0 and frame 1 looks very similar from line features, but these two frames were taken at different locations (note the fire alarm in frame 0). In order to distinguish these frames, we combine line and point features together by using a modified vocabulary tree.

In this paper, we propose a bag-of-word place recognition method that uses both line and point features. Firstly, point and line features are detected and their descriptors are computed. For ground robots using ceiling vision in a typical indoor environment, the scale of different images remain constant but not the orientation of features. Hence a feature descriptor that is invariant to the rotation is desired. In order to maximize the efficiency of the system, a specially designed point feature descriptor is used. As for line features, we use the Line Band Descriptor (LBD) [13] to represent lines detected in images, because it is computationally more efficient and has better performance for most image transformations. An approach that can describe an image with a good combination of the afore-mentioned point and line features as well as other possible feature types is desired. To represent an image with multiple feature types, a modified vocabulary tree that can create visual words with a good combination of different feature types is proposed. Instead of building separate vocabulary trees for each feature type, we use all features together to create a single vocabulary tree. Although this method is proposed for ceiling vision based place recognition, it can be used in different environments.

The rest of this paper is structured as follows: In Section 4, we describe the point and line features that we used. After that the modified vocabulary tree that can combine multiple types of features for place recognition purpose is introduced in Section 5. Experiments are conducted in Section 6 to evaluate the proposed approach. Conclusions are drawn in Section 7.

Feature Extraction

Point extraction and description

A large number of point feature detection approaches have existed in the literature. SIFT [14] and SURF [15] is very popular choices for visual feature detectors for their robustness against lighting and scale changes. However, without the help of modern GPU, it is time consuming to detect SIFT or SURF features, which does not meet our purpose for real-time performance. Hence, we use FAST [16] feature detector with non-maximal suppression. In order to increase the processing speed and reduce the number of features that are very close to each other, the images are down-sampled before applying feature detection and transform these feature locations back to full-size images when performing descriptor calculation.

Feature descriptors are extracted to find feature similarity across frames. Because we intend to speed up place recognition process, scale invariant feature descriptors such as SIFT or SURF are computational expensive and in our ceiling vision application scale invariance is not a necessary requirement of the system. To fulfill the real-time requirement, a customized and fast feature descriptor is desired.

For each frame, we first apply two 5 × 5 Sobel filters (Figure 2) across the down-sampled image along horizontal and vertical directions. For each point in the image, Gx and Gy are the resulting Sobel responses along horizontal and vertical directions. The feature direction can be obtained as atan2 (Gy, Gx) and the gradient magnitude can be represented as $G = \sqrt{G_x^2 + G_y^2}$. The descriptor is designed as concatenation of gradient magnitudes around detected features.

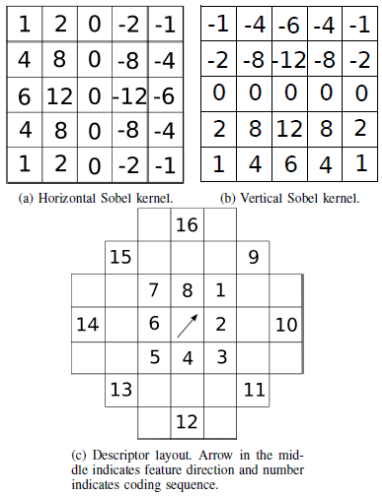
Its layout consists of two circles with 16 coding positions as shown in Figure 2. The descriptor is encoded sequentially from inner circle to outer circle and both follow a clockwise direction. The feature direction is quantized into 8 discrete directions which determine the starting coding position. The value for each dimension of the descriptor is scaled into [0, 255] so that each descriptor can be represented by 16 bytes. By this simple design, the descriptor can be extracted efficiently and invariant to rotation and illumination changes. The descriptor similarity is defined as sum of absolute difference (SAD). In order to optimize the implementation, we use the Streaming SIMD Extensions (SSE) instructions which can perform 16-byte operations very efficiently. Hence, the 16-byte SAD can be performed within one call of SSE instruction.

Line extraction and description

In order to extract line segments from input images, we use a method presented in [13]. The extraction starts with a scale-space pyramid consisting of N octave images which are generated by down-sampling the original image with a set of scale factors and Gaussian blurring. Then, from each layer in the pyramid, EDLine [17] algorithm is applied, which produce a set of lines in the scale space. Lastly, all the detected lines are re-organized by finding corresponding in the scale space. After the extraction of all the line segments, a binary descriptor representing each line is generated using LBD. Given a line segment, a local rectangular region around the line is chosen as the line support region (LSR). The LSR is divided into a set of bands {B1, B2,..., Bm},

whose length equals the one of line. The line direction dL together with the orthogonal direction d forms a local 2D reference frame, whose origin is located at the middle of the line. The gradient of each pixel in the LSR is projected to the local frame. Later on, a global Gaussian function fg is applied to all rows in the LSR, while a local Gaussian function fl is applied to all rows in each band and its nearest two neighbor bands. Using the local and global Gaussian function as well as the gradient of each pixel in the LSR, a so called band description matrix (BDM) is constructed. Each band Bj in LSR has an associated band descriptor (BD), which can be obtained using the mean and standard deviation of the BDM. Once each band has been assigned its BD, the Line Band Descriptor LBD is simply generated by concatenating them:

$$LBD = (BD_1^T, BD_2^T, ..., BD_m^T) \quad (1)$$



(a) Horizontal Sobel kernel. (b) Vertical Sobel kernel.

(c) Descriptor layout. Arrow in the middle indicates feature direction and number indicates coding sequence.

Figure 2: Sobel kernel and feature descriptor layout.

Modified Vocabulary Tree

Conventional vocabulary tree creation

The vocabulary tree based method is an widely used one for place representation [18,19]. It is built by hierarchical k-means clustering, and this makes vocabulary tree approach a practical unsupervised learning method.

In conventional vocabulary tree approach, in order to build the tree structure, a large amount of features are extracted from some training images first. Then, an initial k-means process is run on the training descriptors, obtaining k cluster centers. These clusters form the first level of nodes in the vocabulary tree. The same process is then applied to the subsequent levels (up to L levels). Finally, a tree with a maximum of kL leaves (visual words) is constructed. Given the constructed vocabulary tree, a BoW vector can be created for each image. These BoW vectors consists the database for query purpose.

Although this method has been demonstrated to be scalable and efficient [18,19] it is not designed for applications with multiple feature types. Different feature types may have different dimensions, e.g. SIFT descriptors have 128 dimensions while SURF descriptors have 64 dimensions, which makes it impossible to compare SIFT and SURF descriptor vectors with different dimensions. It is also problematic for different feature types with the same dimension. For example, a 32-dimension ORB descriptor vector could be very close to a 32-dimension LBD descriptor vector in Euclidean space. Thus, they will be clustered into the same visual word which is not acceptable since they convey completely different visual meanings. Moreover, when creating a vocabulary tree, we want to make the vocabulary tree to be distinctive (requiring small quantization cells and a deep vocabulary tree) as well as repeatable (requiring large quantization cells). Apparently, it is not appropriate to build a vocabulary tree with the same branch factor or depth for all kinds of feature types, because, given an image, the feature number of each feature type is different from each other.

It could be argued that multiple separate vocabulary trees can be built for different feature types to sovle the above issues. In that case, not only several vocabulary trees but also multiple databases are required for different feature types. Moreover, when comparing the query image and an image in database, multiple scores are returned for different feature types. Features have different performance in different environment, for example, line features have better performance in environment like corridor but not in rural area full of trees or grass. Hence, these scores should not be treated equally in different environments. However, it is unclear on how to find a general weighting strategy to combine these scores to perform well in different environments.

Modified vocabulary tree creation

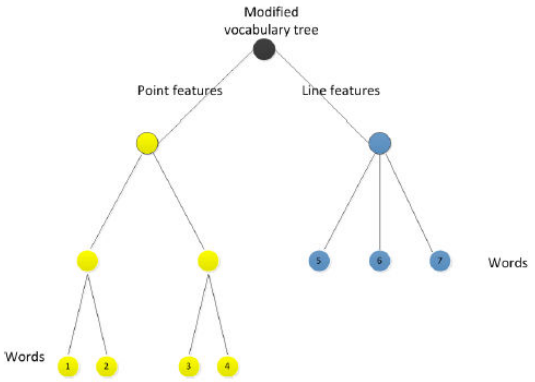


Figure 3: Example of our modified vocabulary tree with a combination of point and line features. Instead of running k-means clustering on the first level, a feature type quantization is performed first. In this example, there are 3 levels and 2 branches on each level for point features, while 2 layers with a branch factor 3 for line features.

Based on the above observations, we propose a modified vocabulary tree as depicted in Figure 3. As can be seen, a feature type quantization step is performed on the training data, defining n two (Figure 3) feature types. The training descriptors are then partitioned into n groups, where each group consists of the descriptor vectors that belong to the same feature type. The hierarchical k-means clustering process is then applied to each feature type, constructing the final tree structure.

One could argue that there may be more than one trees of the heuristic mixing of features, hence the modified vocabulary tree cannot be constructed. Nister et al. [19] have done plenty of experiments comparing different parameter settings (branch and layer factors) and suggested appropriate ones. So even though there are more than one trees of the heuristic mixing of features, the one with the most appropriate setting is preferable. As long as the conventional trees trained with separate feature type can be constructed, the modified vocabulary tree which combines multiple feature types can be construed. Thus it has all the advantages that the conventional trees have. In order to distinguish different feature types in the second layer of vocabulary tree, one extra byte is appended to the end of each feature descriptor to represent feature type in our implementation. And please note that this tree structure can allows us assign different branch and layer factors for different feature types.

To query an image, an image database need to be pre-built. When adding an image It to the database, multiple types of features are extracted from the image and each descriptor vector of any feature type traverses through the vocabulary tree until it reaches any leaf node i. It is inserted to an image list stored in ith node so that when querying an image in the database, the comparison is only needed to be performed with those images that have common words. To accomplish this efficient comparison, the term frequency-inverted document frequency (TF-IDF) scheme is used. We define query qi and database vectors di as:

$$q_1 = n_i w_i$$

$$d_i = m_i w_i$$

where ni is the term frequency of word i in the query image while mi is the term frequency of word i in the database image which are defined as:

$$n_i = \frac{Q_i}{Q} \quad (4) \qquad m_i = \frac{D_i}{D} \quad (5)$$

where Qi is the number of words i in query image. Q is the number of total words in query image. Di is the number of word i in database image. D is the number of total words in database image. The reason why we choose to use the number of total words instead of using the number of words that belong to one feature type in query or database images, is to reach fairness between feature types with different feature numbers. Using TF-IDF scheme, wi is defined as:

$$w_i = ln\frac{N}{N_i} \quad (6)$$

where N is the number of images in the database, and Ni is the number of images in the database with word i. By using this weighting scheme, the frequently occurred words are penalized, while the words that are more distinctive are awarded.

As a result, images are described by bag-of-words vectors. The normalized L1 distance dist (q,d) between two such vectors q and d is defined as:

$$dist(q,d) = \left| \frac{q}{|q|} - \frac{d}{|d|} \right| \quad (7)$$

Hence, for a query image, the better matches in database images receive lower dist (q,d).

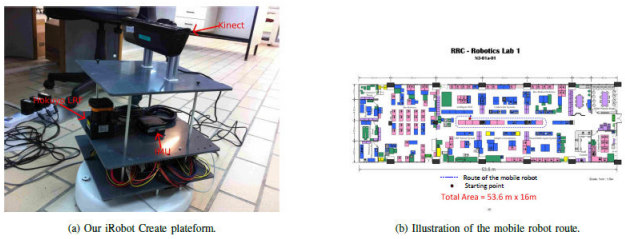
Experiments

In this section, several experiments are conducted and the results are shown to demonstrate the performance of our proposed approach. In the first experiment, we use a Kincect camera pointing up to the ceiling to grab image sequences. Comparisons are made between the performance of our modified tree trained with a combination of LBD and the self-designed point descriptor with other trees trained using LBD and the self-designed point descriptor separately. The proposed method is also tested under varied illumination conditions using ceiling images. Afterwards, the challenging KITTI dataset is used in our second experiment to show the effectiveness of the proposed approach in much more complex environment. All the experiments are conducted in real-time using a Intel (R) i7-4710MQ processor.

Experiments on real robot with ceiling camera

The ceiling images used in this experiment were captured using a Kinect mounted on a iRobot-Create robot (Figure 4a) pointing up to ceiling. The robot was manually navigated by a wireless game pad and the testing data were collected by driving the robot in the Robotics Research Center of Nanyang Technological University (Figure 4b).

Furthermore, the platform is equipped with a high precision IMU and Hokuyo laser range finder which are used to provide the position ground truth of the robot.



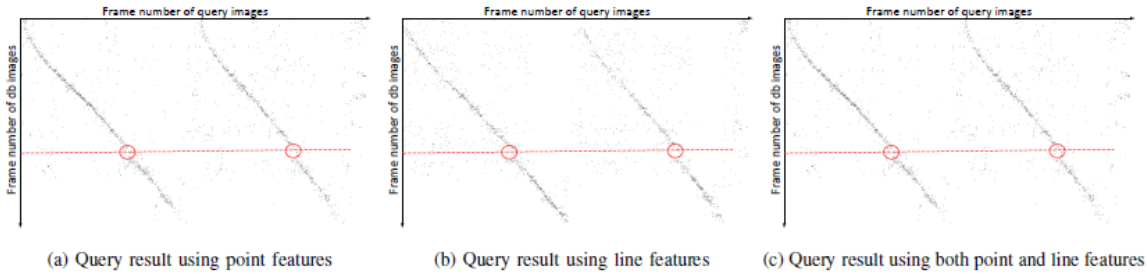
(a) Our iRobot Create plateform. (b) Illustration of the mobile robot route.

Figure 4: Our mobile robot and its trajectory. The Kinect camera is mounted on the iRobot Create platform and pointed up to the ceiling. The robot traversed the same route three times at different time of a day. Using these three image sequences, we compare the retrieval performance of the proposed modified vocabulary tree trained with both point and line features with conventional vocabulary trees trained with point or line feature alone.

In order to show the effectiveness of the modified vocabulary tree, comparisons between conventional vocabulary trees and the proposed are needed. We first controlled the robot moving around the research center and recording a ceiling video. Our self-designed point descriptor and LBD line descriptor are used to describe point and line features. Around 2 million point descriptors and 1 million line features extracted from this video are used to train the conventional and modified vocabulary trees. Following the analysis by Nister [19], a setting with k = 8, L = 5 is used for the line vocabulary tree, obtaining a 20200 visual words, while for the point vocabulary, k = 7, L = 5 is used, resulting in 8987 visual words. And the same setting is applied to our modified vocabulary tree.

After creating the above vocabulary trees, we first navigated the robot traveling the same trajectory three times but at different time of a day. The blue dashed line in Figure 4b is the route used in this experiment. The first loop was collected in the morning when the

lights were still on, while the second loop was collected at noon when the lights were off, and the third loop was collected at night when the lights were on again. So the scenes of the three loops were under illumination changes. We choose the images (1104 frames) in the first loop as database while images in the second (1112 frames) and third (1091 frames) loop are used as query. Thus, images in the second and third loop are queried sequentially. Point and line features are extracted from ceiling images as described in Section 4.



(a) Query result using point features (b) Query result using line features (c) Query result using both point and line features

Figure 5: Query results using separate vocabulary trees and our modified vocabulary tree that combines point and line features. Two axes represent the frame numbers of the database and query images. The dots represent the top five results returned from database.

These three vocabulary trees are tested using the same ceiling image sequence. In order to show the place recognition performance, we plot the top five query results as shown in Figure 5. The horizontal and vertical axes represent the frame numbers of the database and query images, respectively. The dots represent the top five results returned from database. Since in query image sequence, the robot traversed the same trajectory as in database images twice, the distribution of plotted points should follow two diagonal lines. It can be seen, the result using the proposed approach outperforms the ones using single feature type because more dots are distributed on the two diagonal lines. There are some obvious query failures as marked by red circles which occur at the same position of the database images. The reason for that is, there was slight trajectory difference between query and database images when robot made a turn. The detailed comparison results of the three vocabulary trees are shown in Table 1. From the table it can be seen that the results from both query sequences using the modified vocabulary tree have higher success retrieval rate even under different illumination conditions. Some examples of successful retrieval under illumination changes using the proposed method are shown in Figure 6.

Experiments on KITTI dataset

In order to quantitatively evaluate the proposed method and demonstrate its ability in much more complex environment, we use the challenging KITTI dataset [20] which contains both urban and sub-urban environment. Also, the position ground-truth of each captured frame in KITTI dataset is available. We use image sequences with number 00, 02, 05 and 06, in which loop closures occurred most frequently, so that we can use images in the first pass as database images and images in the second pass as query images. The modified vocabulary tree combining point and line features is created in a similar way as discussed above. One difference is that instead of using our self-designed descriptor, we use ORB [21] as point descriptor to gain scale invariance ability. Besides that, we use a setting of k = 10, L = 6 to create the proposed vocabulary tree, which leads to 739725 point visual words and 349613 line visual words.

The database and query image numbers for each sequence are shown in Table 2. We compare our method with the popular FAB-MAP method. The precision-recall curves for the sequences 00, 02, 05 and 06 are shown in Figure 7.

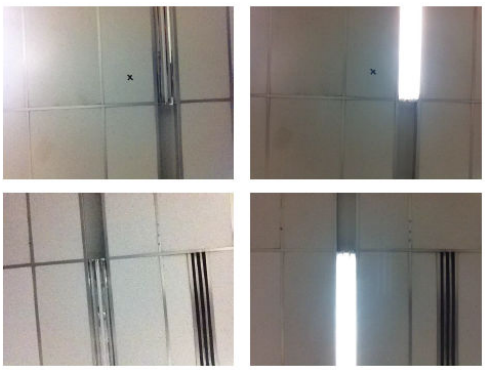


Figure 6: Examples of successful retrieval under illumination changes using the proposed method. Left images are queried scenes in the second loop and right images are the returned scenes from database with highest scores.

Here, precision is the proportion of returned frame pairs that were correct, and recall is the proportion of total correct frame pairs that were returned. It can be seen that our approach shows significant improvement on both the precision and recall rates. In Figure 8, we show examples of correct loop detections in the four KITTI sequences, with point and line corresponding features. Note that the feature matching example image is taken in a well-structured environment where has enough saliency. But pure point or line feature matching results still have a certain percentage of mis-matches.

Since image position ground-truth is available, given a query image, we measure the Euclidean position distance between the query image and its best matching in the database to evaluate the accuracy of the proposed system. The results are shown in Table 3. Different distance threshold levels are defined. One query is considered as correct if the measured distance is smaller than or equal to the distance threshold.

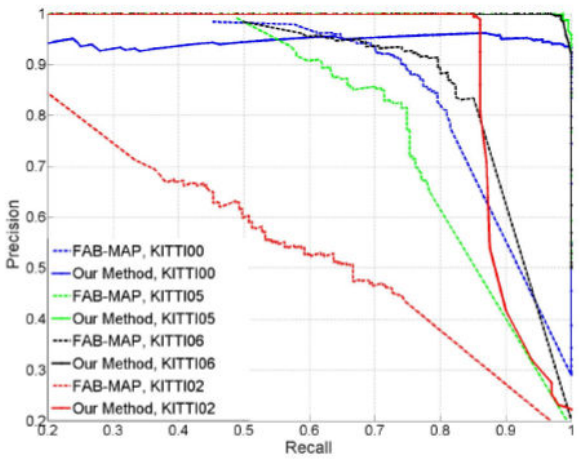


Figure 7: Precision-recall curves in four KITTI dataset sequences without geometrical or temporally consistent check, showing the performance improvement compared to FAB-MAP.

	Tree trained with point features	Tree trained with line features	Modified tree
	Q: Second loop (noon, lights off)		
DB frame numbers (morning, lights on)	1104	1104	1104
success return	763 / 1112	713 / 1112	872 / 1112
success rate	68.61%	64.12%	78.42%
	Q: Third loop (night, lights on)		
DB frame numbers (morning, lights on)	1104	1104	1104
success return	741 / 1112	701 / 1112	858 / 1112
success rate	66.64%	63.04%	77.16%

Table 1: Experimental evaluation of the trees, showing the retrieval performance improvement compared to conventional vocabulary trees. The success return is the number of success retrievals and success rate is the successful retrievals in percentages.

The smallest distance threshold is set to one meter, because the average distance between two consecutive frames is about one meter in KITTI dataset. By setting the distance threshold to one meter, a query result is considered as correct only if it is taken at the same position as query image. Most of the query results are correct even with this smallest distance threshold except KITTI02. The reason for that is, in this sequence, the car was driving through an area filled with trees and grass which is less distinctive than other sequences.

Sequence No.	Total No. of Frames	Database images	Query images
00	4541	0 - 1570	3400 - 3900
02	4661	0 - 4000	4400 - 4600
05	2761	0 - 1200	1300 - 1600
06	1101	0 - 800	830 - 1090

Table 2: Frame numbers of database and query images in the four KITTI sequences.

Dist. Threshold	KITTI00		KITTI02		KITTI05		KITTI06	
	true	false	true	false	true	false	true	false
≤1 meter	363	48	29	142	173	58	206	45
≤5 meter	411	0	169	2	226	5	244	7
≤10 meter	411	0	171	0	229	2	250	1
≤15 meter	411	0	171	0	231	0	251	0
Avg. Error (meter)	0.66		1.58		0.96		0.84	

Table 3: Quantitative evaluation of query performances. Dist. Threshold: distance threshold within which the query results are considered as correct. Avg. Error: average Euclidean distance between the positions of query and result images.

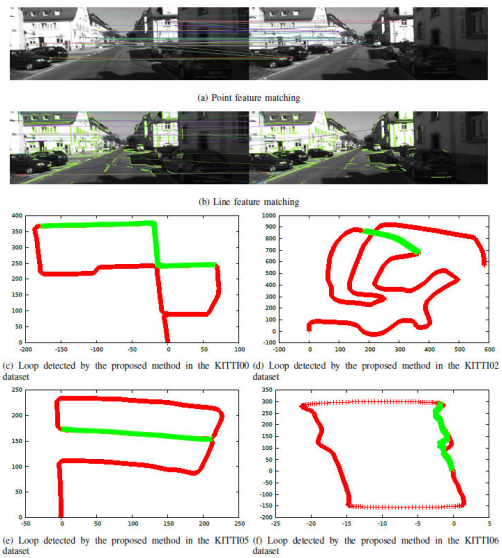


Figure 8: Point and line feature matching examples and loops detected by the proposed method in the KITTI dataset.

Conclusion

In this paper, a modified vocabulary tree that can combine multiple feature types for place recognition is proposed. The ability of combining different feature types makes it possible for users to customize feature combination for various environments. An efficient self-designed point descriptor and LBD line descriptor is combined for loop closure detection on ceiling images. The comparison results show that the proposed modified vocabulary tree that combines both point and line features outperforms the conventional vocabulary tree using each feature alone. To evaluate the performance of the proposed approach in outdoor environment, we test our system on the KITTI dataset. The ORB and LBD descriptors are combined to perform the test. The results demonstrate the effectiveness of the proposed approach. Although only the combination of point and line features is discussed in this paper, the proposed approach can be used on other types of features such as shape descriptors [22].

For all the experiments carried out in this paper, no geometrical or temporal consistent checking is applied to improve the performance. In the future, we plan to use these verification approaches and extend this work to a visual SLAM system.

References

1. Cummins M, Newman P (2008) FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance. The International Journal of Robotics Research 27: 647-665.
2. Paul R, Newman P (2010) Fab-map 3d: Topological mapping with spatial and visual appearance. Robotics and Automation IEEE International Conference pp: 2649-2656.
3. Cummins M, Newman P (2009) Highly scalable appearance-only SLAM-FAB-MAP 2.0. Proceedings of Robotics: Science and Systems, USA.
4. Zamir AR, Shah M (2010) Accurate image localization based on google maps street view. Proceedings of the European Conference on Computer Vision (ECCV).
5. Valgren C, Lilienthal AJ (2010) Sift, SURF and seasons: Appearance-based long-term localization in outdoor environments. Robotics and Autonomous Systems 58: 149-156.
6. Badino H, Huber DF, Kanade T (2012) Real-time topometric localization. IEEE International Conference on Robotics and Automation, USA.
7. Hirose K, Saito H (2012) Fast line description for line-based slam. Proceedings of the British Machine Vision Conference pp: 83.1-83.11.
8. Chandraker M, Lim J, Kriegman D (2009) Moving in stereo: Efficient structure and motion using lines. Computer Vision, pp: 1741-1748.
9. Lee JH, Lee S, Zhang G, Lim J, Chung WK, et al. (2014) Outdoor place recognition in urban environments using straight lines. Robotics and Automation, pp: 5550-5557.
10. Jeong W, Lee KM (2005) Cv-slam: A new ceiling vision-based slam technique. Intelligent Robots and Systems IEEE/RSJ International Conference pp: 3195-3200.
11. Lee HS, Lee KM (2009) Multi-robot slam using ceiling vision. Intelligent Robots and Systems IEEE/RSJ International Conference pp: 912-917.
12. Wang H, Mou W, Suratno H, Seet G, Li M (2012) Visual odometry using rgb-d camera on ceiling vision. Robotics and Biomimetics IEEE International Conference pp: 710-714.
13. Zhang L, Koch R (2013) An efficient and robust line segment matching approach based on (LBD) descriptor and pairwise geometric consistency. Journal of Visual Communication and Image Representation 24: 794-805.
14. Lowe DJ (2004) Distinctive image features from scale-invariant keypoints. Int J Comput Vision 60: 91-110.
15. Bay H, Ess A, Tuytelaars T, Van L (2008) Speeded-up robust features (surf). Comput Vis Image Underst 110: 346-359.
16. Rosten E, Drummond T (2006) Machine learning for high-speed corner detection. Proceedings of the 9th European Conference on Computer Vision pp: 430-443.
17. von GR, Jakubowicz J, Morel JM, Randall G (2010) Lsd: A fast line segment detector with a false detection control. Pattern Analysis and Machine Intelligence IEEE Transactions 32: 722-732.
18. Galvez-Lopez D, Zardos JD (2012) Bags of binary words for fast place recognition in image sequences. IEEE Transactions on Robotics 28: 1188-1197.
19. Nister D, Stewenius H (2006) Scalable recognition with a vocabulary tree. Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, USA.
20. Geiger A, Lenz P, Urtasun R (2012) Are we ready for autonomous driving? the kitti vision benchmark suite. Conference on Computer Vision and Pattern Recognition (CVPR).
21. Rublee E, Rabaud V, Konolige K, Bradski G (2011) Orb: An efficient alternative to sift or surf. Computer Vision (ICCV) IEEE International Conference pp: 2564-2571.
22. Amanatiadis A, Kaburlasos V, Gasteratos A, Papadakis S (2011) Evaluation of shape descriptors for shape-based image retrieval. Image Processing IET 5: 493-499.