Visual Simultaneous Localization and Mapping with Deep Neural Network Based Loop Detection for Augmented Reality

Yang Li, Chao Ping Chen*, Yuan Liu*, Jie Chen*, Changzhao Zhu*, and Ziqi Peng***

*Smart Display Lab, Department of Electronic Engineering, Shanghai Jiao Tong University, Shanghai, China
**Kura AR, San Francisco, California, United States
Email: ccp@sjtu.edu.cn

Abstract

We present a visual simultaneous localization and mapping, in which a deep neural network is adopted for the loop detection. Its working principles, including the tracking, local mapping, loop detection, and global optimization, are set forth in detail. Its overall performance regarding the loop detection and trajectory estimation is investigated.

Keywords

simultaneous localization and mapping; deep neural network; loop detection; augmented reality.

1. Introduction

Over the past two decades, simultaneous localization and mapping (SLAM) has been gaining ground in the applications such as augmented reality [1], autonomous driving *etc*. With SLAM, not only the trajectory of the moving object can be estimated, but also the surrounding three-dimensional scene can be reconstructed in real time. To date, a variety of SLAMs using various sensors—such as lasers, inertial measurement units, cameras *etc*.—have been proposed. Visual SLAM [2], among others, refers to a type of SLAM, which merely relies on the cameras. Further, they can be divided into two subcategories, *i.e.* the feature-based and direct. For example, MonoSLAM [3], PTAM [4] and ORB-SLAM [5] are the feature-based SLAMs. On the other hand, LSD-SLAM [6] and DTAM [7] are known as the direct SLAMs. In this paper, a feature-based visual SLAM is introduced, in which both the points and lines are extracted as features to enhance the accuracy and robustness. Plus, a deep neural network (DNN) based loop detection is adopted to recognize the previously visited scenes.

2. Proposed Architecture

Architecture: Fig. 1 outlines the architecture of the proposed SLAM, which consists of four main threads [8]. Thread 1. Tracking: Through this thread, the extraction and tracking of points and lines are completed; the camera's localization is obtained and the local map is tracked; and whether the current frame is a keyframe is determined. This thread leverages the optimization to minimize the re-projection error. Only the camera's pose in the local map will be treated as the state variables. Thread 2. Local Mapping: The local maps and keyframes are managed and optimized, including the keyframes inserting and culling, and features creating and culling. The 3D spatial features, *i.e.* points and lines, are assigned as optimization variables. Thread 3. Loop Detection: The loops are detected by the DNN, and the accumulated error through the loop fusion is consequently corrected. Thread 4. Global Optimization: This thread reduces the accumulated errors so as to correct the camera trajectory.

Computing Process: Fig. 2 shows the computing process of proposed SLAM, which involves two parallel threads executed for points and lines, respectively [9]. After the points and lines are extracted and described, feature matching is carried out, followed by the camera pose evaluation and scene reconstruction.

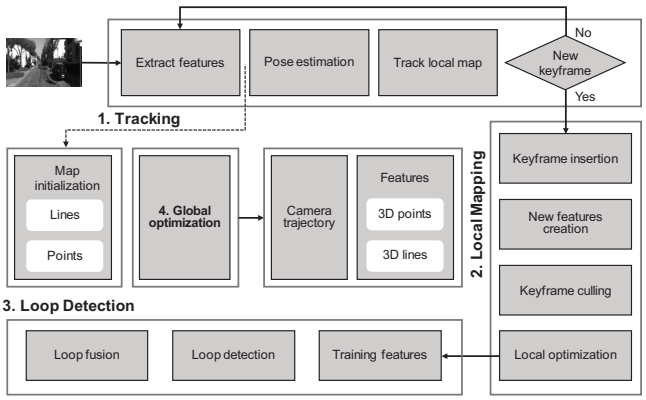


Figure 1. Architecture of the proposed SLAM.

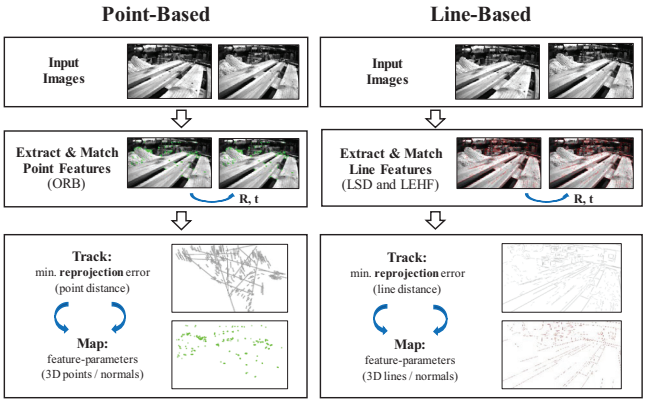


Figure 2. Computing process of the point-based and line-based threads.

3. Loop Dectection

Training: Our DNN for loop detection, which is basically a stacked auto encoder, is built on TensorFlow, an open-source platform for machine learning. The training dataset comes from the TUM RGB-D benchmark, which contains both the RGB-D images and their ground truth trajectories. Each sequence of the images is randomly sampled, and 30% of them are allocated for training, while 20% for testing the ability of the network to represent other images. In order to obtain accurate and robust features of the trained network, eight hyper-parameters related to the deep neural network are assigned in the process of training visual features, which are itemized in Table 1. The auto encoder comprises a total of three hidden layers, and the units in each layer is 200, 50 and 50, respectively. Firstly, the influence of some hyper-parameters on the

performance of the neural network is evaluated. The corruption level, which corresponds to the de-noising, is added to the weight matrix of the first hidden layer. Since the noises always exist in the images, adding the corruption is indispensable for increasing the robustness and preventing the network from being over-fitted. In the absence of corruption, namely the corruption level is 0, in the weight matrix from the hidden layer, the meaningful information in the image will be clouded by the noise, as shown in Fig. 3(a). When the corruption is present, as shown in Fig. 3(b), Fig. 3(c) and Fig. 3(d), sparse edge features of the images can be observed. Out of different corruption levels, the optimal corruption level is obtained at 12%.

Table 1. Hyper-parameters defined for feature training.

Parameter	Value	Description
n_{hidden}	[200,50,50]	units in each layer
η	0.1	learning rate
γ	0.002	penalty factor of continuity
λ	0.005	penalty factor of sparsity
l_c	0.12	corruption level
s_h	0.05	sparsity threshold
n_{batch}	10	batch size of SGD
n_{epoch}	20	times of iteration

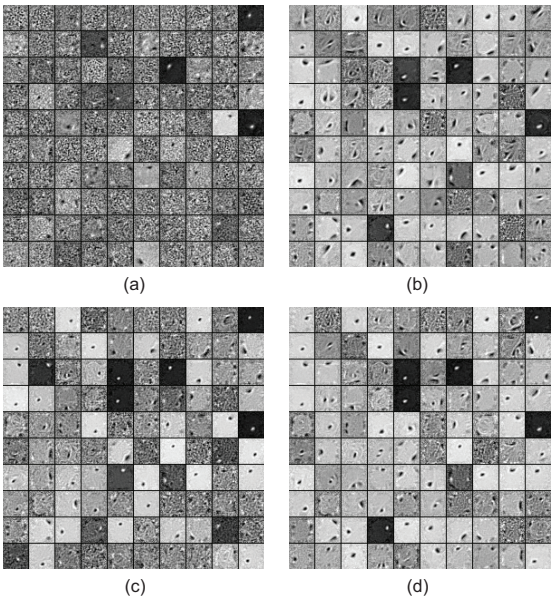


Figure 3. Visualization of the structures trained (10 × 10 units in the result are selected) from the hidden layer at four different corruption levels: (a) 0, (b) 5%, (c) 12% and (d) 30%.

Sparsity: Fig. 4 shows the average response of the images in the hidden units. The impact of adding the corresponding penalties on the solution is judged from the perspective of sparsity. With a sparse penalty in the loss function, most of the useful information in the trained results is preserved, and the redundant features in the network, which are irrelevant to the current training task, are discarded. Adding the sparsity penalty can prevent the network from being over-fitted, and eliminate the redundant features in the network. In addition, it can reduce the computational complexity in the training of the network parameters, and make the network more interpretable.

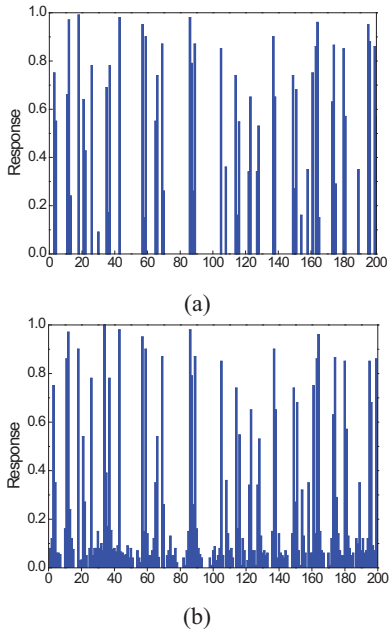


Figure 4. Average response of the hidden units in the images: (a) with the sparsity penalty, and (b) without the sparsity.

Similarity: The difference matrix, computed by the first hidden layer response, is used for detecting the loops [8]. A value that is lower than the threshold indicates that a loop candidate is detected. The similarity matrix, on the other hand, is responsible for fusing the loops between these candidate frames and current frames. Fig. 5 shows the comparison results of the feature graph and their difference in two scenes. In the same scene, the differences of the feature representation are very small, while in the different scenes, the feature differences are relatively large.

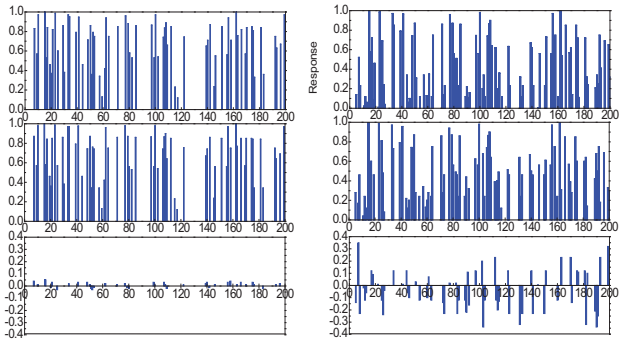


Figure 5. Comparison of the feature graphs in two scenes. (a) In the same scenes, the differences of the feature representation are small. (b) In the different scenes, the feature differences of the feature representation are large.

Precision-Recall Curve: To evaluate the performance of trained features for loop detection, the precision-recall curve is calculated. The precision-recall curves of two competing methods on TUM RGB-D benchmark are shown in Fig. 6. It can be seen that the performance of the deep neural network is better than that of the bag-of-words (BoW). In practical scenes, once a loop is detected, its neighboring frames will also be detected with the current frame, therefore a recall rate approaching 50% is sufficient. When the recall rate hits 50%, the accuracy of the DNN-based loop detection is about 10% higher than that of the traditional BoW.

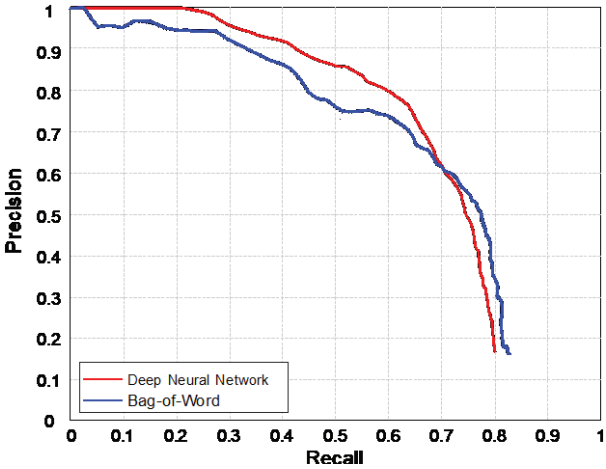


Figure 6. Precision-recall curves of the deep neural network and bag-of-word.

Trajectory Estimation: The sequence fr2/pioneer_slam in TUM RGB-D benchmark is selected to evaluate the performance of the loop detection. The estimated trajectory and the ground truth are compared in Fig. 7. It can be seen that the proposed method accurately detects the loops of the scene, and the estimation trajectory agrees well with the ground truth.

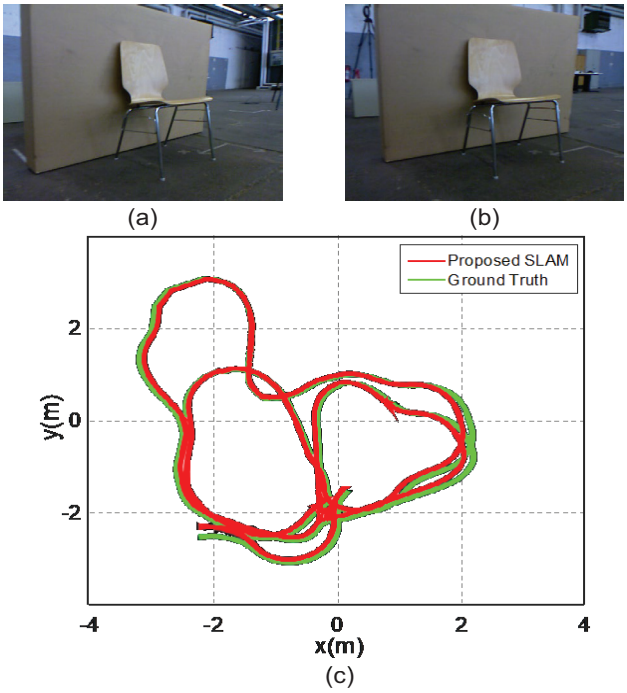


Figure 7. Estimated trajectories with loops. Image (a) and image (b) are captured in the same location at different time. (c) the estimated trajectory versus the ground truth.

4. Localization

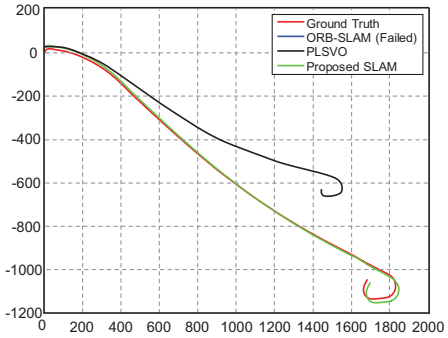
Settings: Localization is evaluated in terms of the accuracy and robustness of trajectory estimation. Based on the indoor datasets known as TUM RGB-D benchmark, the proposed SLAM is compared with other visual SLAMs, including PTAM, LSD-SLAM and ORB-SLAM, to evaluate their localization accuracies. The image sequences, which are not suitable for monocular visual SLAM in TUM RGB-D benchmark, are removed, and consequently only 10 relative sequences are selected. The absolute median RMS error serves as the metric for comparison, and it can be computed through a python script provided by the benchmark. Before calculating the error, the similarity transformation is applied to all the trajectories to align the data. For the sake of reducing the amount of calculation, only the keyframes extracted from the image sequence are calculated.

TUM RGB-D Test: Table 2 summarizes the results of the absolute median RMS errors of the camera trajectories executed for over five times in each sequence. In all these 10 indoor scenes, the proposed SLAM exhibits a high accuracy in the trajectory of camera compared with others. In particular, in the floor and office scenes containing abundant lines, the localization accuracy of the proposed method is much higher than the point-based ORB-SLAMs. By contrast, PTAM was unreliable in several scenes since it fails in 6 out of 10 sequences. In addition, LSD, which is a direct-based method, also loses its trajectory in 2 scenes, and its localization accuracy is much lower than that of our SLAM.

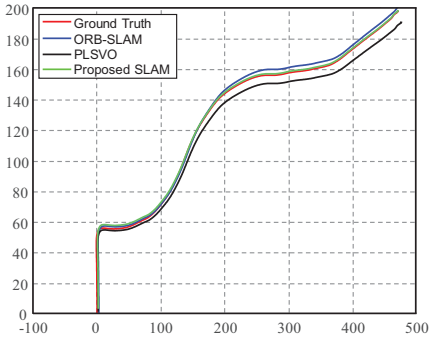
Table 2. Absolute median RMS errors (cm) of camera trajectories on TUM RGB-D benchmark.

Sequence	ORB-SLAM	PTAM	LSD-SLAM	Proposed SLAM
fr1_xyz	1.41	1.23	9.21	1.34
fr1_desk	1.69	x	10.65	1.66
fr1_floor	5.64	x	38.07	3.43
fr2_xyz	0.64	0.49	2.15	0.56
fr2_desk_person	0.63	x	31.73	6.34
fr2_360_kidnap	4.99	2.63	x	3.78
fr3_office	4.35	x	38.53	2.13
fr3_sit_xyz	0.80	0.95	7.84	0.66
fr3_walk_xyz	1.64	x	12.44	1.78
fr3_walk_halfsph	1.92	x	x	1.70

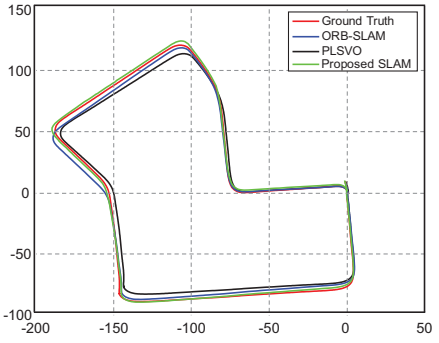
KITTI Test: As for the outdoor scenarios, KITTI, an outdoor dataset with the ground truth, is used for the experiments. It contains 22 sequences collected from outdoor autonomous driving scenes. In order to highlight the performance of the SLAM after adding the lines into the features, four representative sequences: 01, 03, 07 and 09 with sufficient lines are picked as test sequences. The trajectories obtained by the proposed SLAM, ORB-SLAM and points and lines based stereo visual odometry (PLSVO) are compared with the ground truth, as shown in Fig. 8. The trajectories of our SLAM overlap better with the ground truth than the ORB-SLAM and PLSVO. Plus, the proposed SLAM exhibits good robustness of loop detection at the same time. As can be seen from Fig. 8(a), our SLAM works well in Sequence-01 (highway), while the ORB-SLAM fails. This is because the number of points appearing on the highway scene are insufficient. On the contrary, in our SLAM being added with the lines, the number of features available for matching and tracking increase substantially. Especially in the linear running stage, the estimated trajectory of our SLAM agrees well with the ground truth, since there is an adequate amount of lines. Besides, PLSVO, which adopts the violent matching between the extracted features in the visual odometry, greatly suffers from the accumulated errors due to the absence of loop detection. Though able to work in Sequence-01, the trajectories of PLSVO deviate far away from the ground truth as the distance increases.



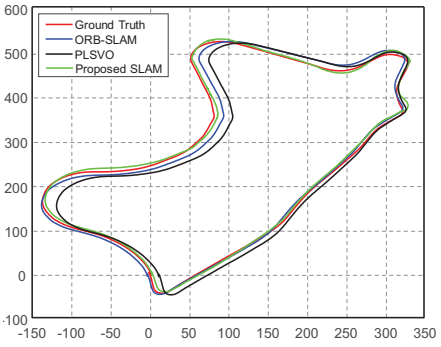
(a)



(b)



(c)



(d)

Figure 8. Estimated trajectories of the proposed SLAM, PLSVO, ORB-SLAM, and ground truth from the four sequences from KITTI: (a) sequence-01, (b) sequence-03, (c) sequence-07, (b) sequence-09.

Localization Accuracy & Robustness: Table 3 lists the results of the absolute median RMS errors of keyframe trajectories in each sequence, from which it can be seen that the proposed SLAM outrivals the point-based ORB-SLAM and direct-based PLSVO with better localization accuracy and robustness.

Table 3. Absolute median RMS errors (unit: meter) of keyframe trajectories on KITTI dataset.

Sequence	ORB-SLAM	PLSVO	Proposed SLAM
01	N/A	314.07	19.48
03	2.63	23.41	2.39
07	3.45	12.81	3.93
09	7.72	25.96	5.27

5. Conclusions

A visual SLAM, which features both points and lines and employs DNN for the loop detection, has been demonstrated. Its overall performance regarding the trajectory estimation and loop detection has been investigated. Our experimental results indicate that the proposed SLAM, compared to its counterparts, shows better accuracy and robustness during the trajectory estimation. As for the loop detection, DNN turns out to be superior to the traditional BoW, as it could decrease the accumulated errors of the estimated trajectories and reconstructed scenes.

6. Acknowledgments

National Natural Science Foundation of China (61831015); Science and Technology Commission of Shanghai Municipality (19ZR1427200); Shanghai Rockers Inc. (15H100000157).

7. References

[1] J. Chen, L. Mi, C. P. Chen, H. Liu, J. Jiang, and W. Zhang, "Design of foveated contact lens display for augmented reality," *Opt. Express* **27**, 38204 (2019).

[2] B. Yu, Y. Li, C. P. Chen, N. Maitlo, J. Chen, W. Zhang, and L. Mi, "Semantic simultaneous localization and mapping for augmented reality," in *SID Display Week*, pp. 391–394, Los Angeles, US (2018).

[3] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: real-time single camera SLAM," *IEEE Trans. Pattern Anal. Mach. Intell.* **29**, 1052 (2007).

[4] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *Proc. 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, Nara, Japan (2007).

[5] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM: a versatile and accurate monocular SLAM system," *IEEE Trans. Robot.* **31**, 1147 (2015).

[6] J. Engel, T. Schops, and D. Cremers, "LSD-SLAM: large-scale direct monocular SLAM," in *Proc. 13th European Conference on Computer Vision*, Zurich, Switzerland (2014).

[7] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, "DTAM: dense tracking and mapping in real-time," in *Proc. IEEE International Conference on Computer Vision*, Barcelona, Spain (2011).

[8] Y. Li, C. P. Chen, N. Maitlo, L. Mi, W. Zhang, and J. Chen, "Deep neural network-based loop detection for visual simultaneous localization and mapping featuring both points and lines," *Wiley Advanced Intelligent Systems* **1**, 1900107 (2019).

[9] Y. Li, B. Yu, C. P. Chen, N. Maitlo, W. Zhang, and L. Mi, "Monocular SLAM using probabilistic combination of point and line features," in *OSA 3D Image Acquisition and Display: Technology, Perception and Applications*, p. 3W2G.7, Orlando, US (2018).