2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
September 24–28, 2017, Vancouver, BC, Canada

Deep Learning for 2D Scan Matching and Loop Closure

Jiaxin Li[1*], Huangying Zhan[2*], Ben M. Chen[3], Ian Reid[2], Gim Hee Lee[4]

***Abstract*— Although 2D LiDAR based Simultaneous Localization and Mapping (SLAM) is a relatively mature topic nowadays, the loop closure problem remains challenging due to the lack of distinctive features in 2D LiDAR range scans. Existing research can be roughly divided into correlation based approaches e.g. scan-to-submap matching and feature based methods e.g. bag-of-words (BoW). In this paper, we solve loop closure detection and relative pose transformation using 2D LiDAR within an end-to-end Deep Learning framework. The algorithm is verified with simulation data and on an Unmanned Aerial Vehicle (UAV) flying in indoor environment. The loop detection ConvNet alone achieves an accuracy of 98.2% in loop closure detection. With a verification step using the scan matching ConvNet, the false positive rate drops to around 0.001%. The proposed approach processes 6000 pairs of raw LiDAR scans per second on a Nvidia GTX1080 GPU.**

I. Introduction

The ability to detect and complete loop closure correctly is critical to many autonomous robotic applications. Firstly, loop closure is a major component in the back-end graph optimization in the front-end / back-end framework of SLAM [1]. It helps to eliminate odometry errors that are accumulated in the front-end for long term operations. Secondly, localization in a prior map becomes possible with loop closure. A direct extension of such ability is to allow recovery from the kidnapped robot problem [2], which significantly improves the robustness of the system. Thirdly, cooperative mapping with multiple robots requires loop closure algorithms, so that the submaps from various robots can be merged into a global consistent map.

这里说 loop
closure的作用

In the community of computer vision, loop closure has experienced intensive research in the past few decades. Mature algorithms such as DBoW2, FAB-MAP are widely used in state-of-the-art visual SLAM systems including ORB-SLAM and LSD-SLAM etc. On the contrary, little attention has been given to loop closure with 2D LiDAR sensor. Most popular solutions such as FAST-SLAM [3], Hector SLAM [4] are LiDAR-based odometry without loop closure. Only very recently, LiDAR based loop closure is implemented together with odometry in the Google Cartographer [5].

分别说了
1. 基于visual
的loop closure
DBoW2
FAB-MAP

2. 基于LIDAR的
loop closure

In comparison to images, 2D scans encode much less information. The lack of rich intensity gradients, which are most commonly used in image-based feature extraction, makes it extremely difficult to extract distinctive features for loop closure detection. Furthermore, some structured features e.g. corners have weak variations in the range measurements. Consequently, a naive migration of feature detectors and descriptors from computer vision is impractical [6].

Existing works on detecting loop closures mainly focus on designing proper feature extraction, followed by Nearest Neighbor (NN) search, BoW retrieval or classifier etc. Inspired by the recent astonishing success of deep learning algorithms in extracting and classifying features, we approach the 2D LiDAR loop closure problem as a classification problem, to answer whether a pair of laser scans are captured in nearby location. It is shown that our formulation leads to extremely high accuracy. Moreover, by making use of state-of-the-art deep learning framework and the computational power of contemporary GPUs, loop closure can be performed in a short time with exhaustive matching, which was believed to be impractical previously [7].

这里说loop
closure的
就检索方法主
要有
1. NN search
比如说计算特
征描述子的最
近距离

2. BoW
retrival

3. classfier
这种方法少见

The main contribution of this paper is to introduce a solution for scan matching and loop closure detection using 2D LiDAR scans within a single deep learning framework as shown in Figure 1. The design, training and analysis of our proposed deep network are discussed in Section III. In Section IV, the proposed algorithm is validated by both simulation and real world applications to provide reliable detection of loop closure at an extremely high speed. In addition, the visualization and analysis of the trained network give a novel insight into the type of features that are critical to scan matching and loop closure detection. Finally, the conclusion is elaborated in Section V.

II. Related Work

Early attempts utilized Monte Carlo localization to solve 2D LiDAR based loop closure, but the idea of searching in the pose space is not scalable. Subsequently, Stachniss et al. [8] improved standard Rao-Blackwellized algorithm by associating both occupancy grid map and topological map to each particle. In [9], Neira et al. achieved linear time relocation by applying geometric constraints on features in a stochastic map.

1) Feature Based Approach: A majority of research focuses on feature based approaches, i.e. detect and describe features from either single scan or submap, and employ various data association or classification algorithms to detect loop closure. In [10], Bosse and Zlot described submaps with orientation histogram, projection histogram, and a novel entropy sequence of projection histogram. Also, they proposed an exhaustive approach to deal with unstructured environments which common correlation methods can not handle. In 2009, Bosse and Zlot [11] further extended their work by

[*] The first two authors contributed equally. The source code of this work is released at `http://uav.ece.nus.edu.sg/li2017deep.html`

[1] Jiaxin Li is with the Graduate School for Integrative Science & Engineering, National University of Singapore (NUS). `jli@u.nus.edu`

[2] Huangying Zhan and Ian Reid are with The University of Adelaide and The Australian Centre of Excellence in Robotic Vision.

[3] Ben M. Chen is with the Department of Electrical and Computer Engineering, NUS.

[4] Gim Hee Lee is with the Department of Computer Science, NUS.

978-1-5386-2682-5/17/$31.00 ©2017 IEEE

proposing a methodology to design keypoint locations and descriptors that models the region around a keypoint.

In 2010, Diego et al. [6] proposed the Fast Laser Interest Point Transform (FLIRT) algorithm that uses curvature based detector and beta grid descriptor. The former defines an integral operator to map the scan curve into multi-scale parameterization. The latter encodes occupancy probabilities and the variance in a polar histogram. This work is further improved by Diego et al. in 2013 [12], where the Geometry FLIRT Phrases is introduced as an efficient retrieval approach. The extracted FLIRT features from 10000 training scans are clustered into a few hundred words, i.e. a vocabulary. By using a modified Inverse Document Frequency (IDF) technique–i.e. scan ID numbers are stored against the words–place recognition can be greatly accelerated. Also, Diego et al. proposed a geometry verification step to compare any two scans by checking the clockwise ordering of their FLIRT features. This idea of utilizing geometrical information is extended by Himstedt et al. in 2014 [7], where the relative orientation of the landmarks and the distance of co-occurring landmarks are encoded into the original FLIRT descriptor.

2) Optimization Based Approach: In 2015, Olson [13] adopted the multi-resolution image pyramid method to achieve robust scan-to-scan matching. In addition, a heap structure is utilized to realize "one-to-many" and "many-to-many" search, which makes it possible to perform loop closure in large scale environments. A year later, Hess et al. published the Google Cartographer LiDAR SLAM algorithm [5], which implements odometry together with a real-time loop closure algorithm. The key idea is building multiple submaps, and aligning new scans to nearby submaps to generate constrains on a graph. The real-time alignment is achieved by a branch-and-bound approach.

3) Learning with LiDAR: The benefits of deep learning have not been fully felt in the field of geometric matching of 2D LiDAR data. Other machine learning techniques were employed for loop-closure problem. In 2009, Granstrom et al. [14] used an AdaBoost classifier and designed 20 hand-crafted features e.g. average range etc. to serve as the input to the classifier. The detection rate is 85% and is likely to be higher if deep learning algorithms were employed.

Nicolai et al. [15] utilized ConvNets on processing 3D LiDAR scans for odometry, but the results are disappointing compared with existing scan matching techniques. The closest work from ours is from Pfeiffer et al. [16], who trained a target-oriented navigation model to give steering commands. However, their work focused on deep learning with motion planning, which is different from ours.

III. Design of the Deep Network

This section describes our end-to-end deep network for the scan matching and loop closure detection problems. Our network shown in Figure 1 first learns the features needed for both scan matching and loop closure detection in a common stack of convolutional layers. Next, these features are passed on to two separate fully connected layers for scan matching and loop closure detection respectively.

A. Fully Connected Layers

One of the fully connected layer stacks is designed to do regression and compute the relative transformation between two given scans. The other fully connected layer stacks is designed to do classification and determine the presence of a loop closure opportunity.

1) Scan Matching: Given any pair of LiDAR scans s_i, the objective is to find the relative pose transformation $T = [\Delta x, \Delta y, \Delta\theta]^T$ between the two scans. $\Delta x, \Delta y$ represent translation, and $\Delta\theta$ is the rotation angle. Let us denote g(.) as the unknown function that maps s_i to T:

$$T = g(s_i). \tag{1}$$

The function $g(.)$ is learned during the training phase. The cost to be optimized is defined as the Euclidean loss $\|T_{label} - T\|_2$, where T_{label} represents the label, i.e. ground truth.

2) Loop Closure Detection: Scan-to-scan loop closure detection checks whether a place is revisited given a pair of scans s_i. This can be formulated as a classification problem, where class 0 represents no loop closure and class 1 represents loop closure detected. We design and train a ConvNet as a scoring function $f(s_i) \in \mathbb{R}^2$ to assign scores to the two classes with a pair of scans as input. The cost to be minimized is the cross-entropy loss,

$$L_i = -\log(p) = -\log(\frac{e^{f_{y_i}}}{\sum_j e^{f_j}}) \tag{2}$$

where $y_i \in \{0, 1\}$ is the ground truth label, and f_j represents the j-th element in the vector f. Here $p = \frac{e^{f_{y_i}}}{\sum_j e^{f_j}}$ is the Softmax function. For instance, for a pair of scans s_i that belongs to the same place, $p = \frac{e^{f_1}}{e^{f_0}+e^{f_1}}$. If the scoring function $f = [f_0, f_1]^T$ gives $f_1 \gg f_0$, the cross-entropy loss L_i will be near zero. On contrary, the loss L_i will be large if $f_1 \ll f_0$.

In a probabilistic view, the Softmax function p can be interpreted as a normalized probability of assigning the correct label y_i given the input s_i and the network f. Minimizing the cross-entropy loss L_i equals to minimizing the negative log likelihood of the correct class, i.e. performing Maximum Likelihood Estimation (MLE).

B. Convolutional Neural Network

Both $g(.)$ and $f(.)$ are complex functions that take scans as input and generate either pose transformation or loop closure classification. We use ConvNets to learn the features needed for these functions because they are proven to be capable of modeling highly non-linear problems. Furthermore, the choice of ConvNet is inspired by previous efforts on SLAM and loop closure detection. Most early algorithms like FastSLAM relies on features / landmarks to perform localization, while recent loop closure solutions with LiDAR or camera highly depend on detecting and matching features. It is well known that convolutional layers in ConvNet can be interpreted as transforming inputs into feature representations, which are used for regression or classification later in the fully connected layers. A consistent result for the deep

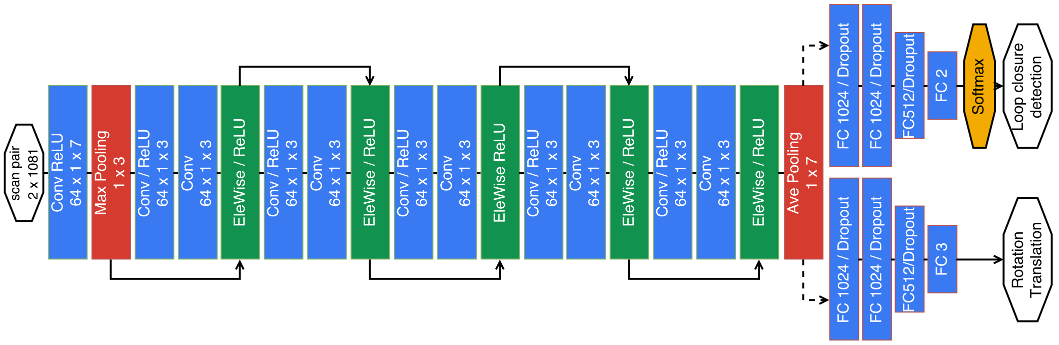


Fig. 1. Our deep network for scan matching and loop closure detection. There is a common stack of convolutional layers with two fully connected layers for each task.

learning community is that the learned features outperform hand-crafted ones in most situations.

Based on the assumption that the features needed for scan matching are similar to features needed for loop closure detection, we design a single stack of convolutional layers to learn the features needed for $g(.)$ and $f(.)$ as shown in Figure 1. This assumption is verified during the training phase stated in Section III-C. The design of the convolutional layers follows the ResNet framework [17]. Similar to results in the computer vision community, we found that the residual structure has superior performance in both scan matching and loop closure detection compared with traditional networks.

C. Training the Network

1) Training the Scan Matching: The input is a pair of laser scans i.e. a 2×1081 matrix for the scan matching network. The output is a vector $T = [\Delta x, \Delta y, \Delta\theta]^T$. We set up a UAV simulation platform to get millions of samples for training. The simulator is built with Robot Operation System (ROS) and Gazebo, where the UAV is mounted with a Hokuyo UTM-30LX range finder with a maximum range of $30m$ and field-of-view of 270°. Multiple indoor environments are constructed for data acquisition. The maximum linear velocity of the simulated drone is $0.8m/s$, and the maximum yaw velocity is $0.5rad/s$. The ground truth positions $[x_j, y_j, \theta_j]^T$ and the range measurements are recorded at the speed of 10Hz. A training sample can be acquired according to Equation (3), where c is an integer to control the interval between two range measurements.

$$\begin{aligned} \Delta x &= (x_{j+c} - x_j)\text{cos}\theta_j + (y_{j+c} - y_j)\text{sin}\theta_j \\ \Delta y &= (y_{j+c} - y_j)\text{cos}\theta_j - (x_{j+c} - x_j)\text{sin}\theta_j \\ \Delta\theta &= \theta_{j+c} - \theta_j \end{aligned} \tag{3}$$

For example, the actual frequency of generating $[\Delta x, \Delta y, \Delta\theta]^T$ is 2Hz when $c = 5$. Here we choose $c = \{5, 10, 20\}$, i.e. the training samples are a combination of 2Hz, 1Hz and 0.5Hz data. Considering the maximum velocity of the simulated UAV, Δx and Δy is $1.6m$ at most while $\Delta\theta$ is $1rad$ at maximum in the training dataset. In addition, samples of $c = \{-5, -10, -20\}$ are added so that both forward and backward motions can be captured. A training dataset consisting of around 3 millions samples is created for the scan matching ConvNet.

2) From Scan Matching to Loop Detection: The dataset generation for loop closure detection is similar to that for scan matching. Samples of $c = \{\pm5, \pm10, \pm20, \pm50, \pm500, \pm2500\}$ are built. The labels are set to 1, i.e. loop detected, if the pose transformation satisfies $\Delta x < 1.6, \Delta y < 1.6, \Delta\theta < 1$. On the other hand, samples with large transformations are set to label 0.

Loop closure detection shares the same convolutional layers with the scan matching ConvNet. By only back propagating the fully connected layers, we obtained a test accuracy of 98.2%. Actually we attempted to train the loop detection ConvNet from scratch without the pre-trained convolutional layers, but this resulted in a slightly lower testing accuracy of 95%. This phenomenon proves our hypothesis that the features extracted by convolutional layers of scan matching ConvNet can be directly applied to detecting loop closures.

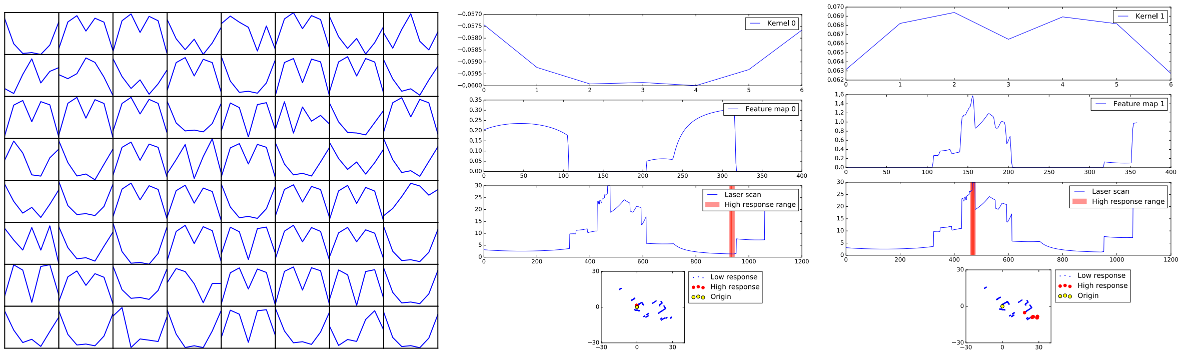
D. Visualization and Analysis

In order to understand the semantic meaning of learned features, the learned convolutional kernels, convolved output (feature maps) and the neuron responses are visualized.

1) Visualization: There are two methods adopted in this work for the visualization. The first method focus on kernels and feature maps. After visualizing kernels learned in all convolutional layers, we found that the kernels usually follow a few patterns in each layer except those learned to be approaching zeros. For example in Figure 2(a), there are typically two patterns in the first convolutional layer - a concave and convex kernel.

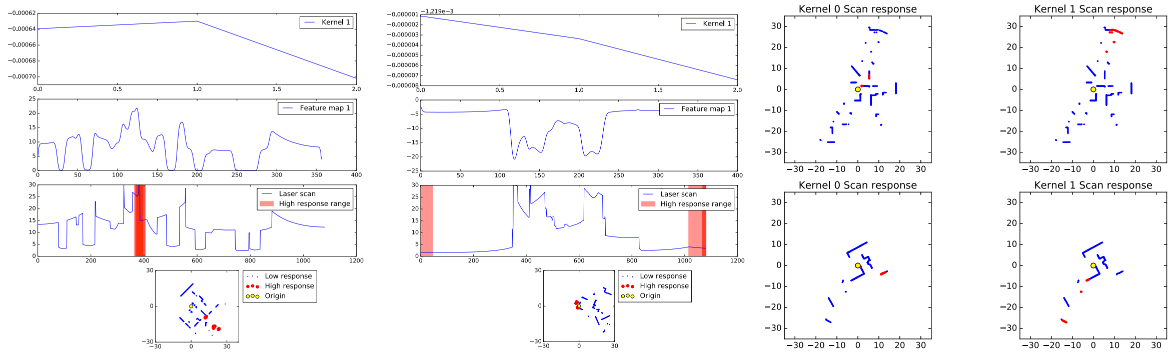
The second visualization method concentrates on the receptive field with high neuron response, i.e., we try to find the range measurements that can produce high values after convolution with the kernels. In Figure 2(c), the 1081 range measurements generate a feature map of length 360 after convolving with a 1×7 kernel (stride equal to 3). Each value in the feature map corresponds to a receptive field in the input scan, i.e., 7 range measurements. The high response receptive fields are visualized in Figure 2 and 3.

2) Analysis: From the visualization results shown in Figure 2(b) and Figure 2(c), it is found that the kernels in the first convolutional layer have an effect similar to distance filters. Kernel 0 responds to the small range measurements strongly while kernel 1 responds to the large measurements.



(a) (b) (c)

Fig. 2. (a) 64 Kernels learned in the first convolutional layer. (b) The first row is the visualization of Kernel 0; The second row plots the feature map of Kernel 0. In the original input scan shown in the third row, highlighted areas are the receptive fields corresponding to top-5 feature map response. The intensity of highlighted area is positively correlated to the strength of response. The fourth row draws the input scan in 2D Cartesian space where the high response measurements are highlighted in red. (c) Similar visualization result based on Kernel 1.



(a) (b) (c)

Fig. 3. (a) Kernel 1 in 7th convolutional layer and corresponding visualization. (b) Kernel 1 in 11th convolutional layer and corresponding results. (c) High response examples from the deepest residual layer.

The 2D Cartesian space visualization proves the hypothesis. In Figure 2(b), the receptive fields with high response are the points that are close to the origin, while in Figure 2(c), the distant points produce high response.

Further investigations are carried out on deeper layers. Some visualization results from various convolutional layers, including kernels and feature maps, are plotted in Figure 3(a) and 3(b). Also, some high response receptive fields from the deepest convolutional layer are shown in 2D Cartesian coordinate, as in Figure 3(c). Unfortunately, physical meanings of the kernels in deeper layers are not as obvious as that in CONV1. Some distant lines or close lines, and points at the edge of lines produce high response. And surprisingly, there are few corner points that generate high response in the ConvNet. Although it is not clear why these regions are emphasized by ConvNets, the visualization results are inspiring. It may be a good choice to pay more attention to edge points, distant lines and close lines when dealing with geometrical perception of 2D laser scans.

E. Geometrical Validation

After training, the loop closure detection ConvNet achieves a testing accuracy of 98.2% that outperforms existing algorithms. However, the 1.8% error may still lead to false alarm in large scale applications. Therefore a validation scheme is proposed to reduce the false positive rate to almost zero. In the case that the loop closure detection ConvNet reports a positive prediction, the same pair of input scans (s_0, s_1) are put into the scan matching ConvNet to get a pose transformation $T = [\Delta x, \Delta y, \Delta\theta]^T$. We align s_1 with s_0 using the predicted transformation T in the Cartesian space:

$$s'_{1i} = \begin{bmatrix} \cos\Delta\theta & -\sin\Delta\theta \\ \sin\Delta\theta & \cos\Delta\theta \end{bmatrix} \begin{bmatrix} s_{1i_x} \\ s_{1i_y} \end{bmatrix} + \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}, \quad (4)$$

$$r_i = \| s'_{1i} - \mathrm{NN}_{s_0}(s'_{1i}) \|_2, \quad (5)$$

where $\mathrm{NN}_{s_0}(s'_{1i})$ is the nearest neighbor search of the transformed point s'_{1i} in the scan s_0 in the Cartesian space. Intuitively, the error vector r measures how well the two scans aligns given a geometrical transformation.

Inspired by [18], a robust t-distribution weighting w_i is applied to the alignment cost so that the alignment error caused by occlusion and noise etc. can be handled properly:

$$w_i = \frac{v+1}{v + (\frac{r_i}{\sigma})^2}. \quad (6)$$

σ in (6) is calculated iteratively using

$$\sigma^2 = \frac{1}{n}\sum_i r_i^2 \frac{v+1}{v+(\frac{r_i}{\sigma})^2}, \qquad (7)$$

where the tunable parameter $v = 5$ is a typical choice.

A robust alignment error $\rho(r) = \sum_i w_i r_i$ is computed for each positive classification result, to filter out those with high alignment error. The effect of such geometrical validation is two fold. The first is to avoid false positive classification of the loop detection ConvNet. The other situation when geometrical validation fails is that the loop closure classification gives correct prediction but the scan matching regression is inaccurate. The geometrical validation benefits the system by rejecting both incorrect and inaccurate loop closures.

IV. Experiments

The scan matching and loop closure detection ConvNets are tested in a simulation platform, as well as on a UAV flying in an indoor environment. The experiments focus on detecting and performing loop closure. Our scan-to-scan framework makes it easy to embed the ConvNets into any modern 2D LiDAR SLAM algorithms. In particular, we enhance the Hector SLAM proposed by Kohlbrecher et al. [4] with our proposed approach, which results in significant improvement in simulation and UAV navigation.

In the ConvNet enchanced SLAM, odometry is performed by the Hector SLAM and keyframes are created incrementally so that each pair of adjacent keyframes satisfies $\Delta x < 1.6m, \Delta y < 1.6m, \Delta\theta < 1.0rad$, which is the same as the setting in Section III-C. On the creation of each keyframe, it is checked against every keyframe created before using the loop closure detection ConvNet. The few keyframes created immediately before the new keyframe are skipped because it is unnecessary to perform loop closure for neighbor measurements. The possible loop closing keyframes reported by the loop detection ConvNet will go through the geometrical validation using the scan matching ConvNet. For the confirmed loop closures, the pose transformations are further refined with standard Iterative Closest Point (ICP) algorithm, based on the prediction of the scan matching ConvNet. A graph is built to capture the detected loops and the respective transformation.

A. Simulation

A diverse indoor environment, shown in the bottom-left of Figure 4(b), is built with ROS and Gazebo, with the size of around $60m \times 60m$. In addition, Gaussian noise $N(0, 0.05^2)$ is augmented into the simulated Hokuyo UTM-30LX range finder. The loop closure detection follows the brute-force searching strategy mentioned above. During the simulation shown in Figure 4(a), the simulated drone travels for 143m, resulting in 288 keyframes. In total 39340 laser scan pairs are tested with the ConvNet, shown in Table I.

In the simulation, we try performing SLAM with only the ConvNets, i.e., the scan matching ConvNet instead of the Hector SLAM plays the role of odometry. The resulting green trajectory is not perfect but still in accordance with our expectation. In odometry, any tiny transformation inaccuracy will accumulate and finally results in large error. Usually it is difficult to train a regression ConvNet to perfectly track a highly non-linear function like scan matching.

TABLE I
Performances of Loop Closure Detection

	Scan Pairs	TP	FP	Precision	Recall	Acc.
sim. w/o GV	39340	6	115	5.0%	100%	99.7%
sim. w/ GV	39340	6	0	100%	100%	100%
fly. w/o GV	780	13	26	33.3%	28.8%	92.5%
fly. w/ GV	780	13	0	100%	28.8%	95.9%

In Figure 4(a), the original Hector SLAM, represented by the blue line, begins drifting after a few meters because of the augmented scan noise and the challenging diverse environment. When the drone returns to the the starting point, loop closure is detected and performed by the ConvNets. The trajectory after loop closure is shown in the black line, which has competitive accuracy compared to the Google Cartographer [5] algorithm. Because of the modern deep learning framework, our approach performs thousands of detections per second, which is significantly faster than any traditional algorithms including the Google Cartographer.

B. Real-time Loop Closure on UAV

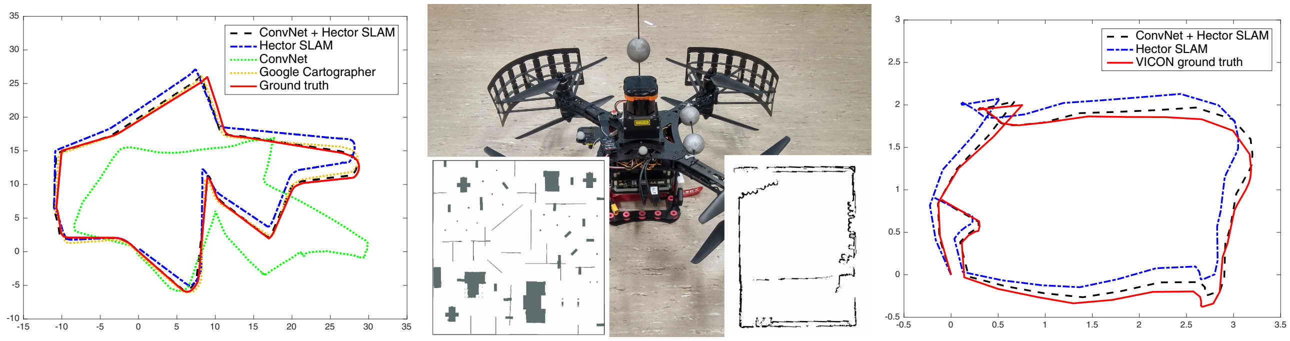
To evaluate the performance and robustness in practical environment, a UAV is navigated with the proposed ConvNet enhanced Hector SLAM. The occupancy map of the flight environment is shown in the bottom-right of Figure 4(b).

1) UAV Setup: As shown in Figure 4(b), the self-built UAV is equipped with a Hokuyo UTM-30LX range finder. An Intel NUC computer is mounted to run a ROS based navigation software, which includes the UAV localization and a A-star based path planning algorithm. The sensor data from the range finder are sent back to a workstation in real time via WiFi. The workstation runs the proposed CNN enhanced Hector SLAM algorithm and sends the localization result back to the UAV. Similarly, brute-force searching strategy is used, and loop closures are conducted using the ICP refined scan matching. In addition, a VICON system is used to record the ground truth trajectory of the UAV.

2) Results: Although the room is relative neat for LiDAR sensors, significant drift can be found for the original Hector SLAM algorithm. That is probably because the fluctuation in the height of the UAV, including the takeoff and landing process, violates the planar assumption of 2D SLAM. Or in other words, the changes of range measurements come from not only the planar movement of the UAV, but also the environment's structure that varies in different heights. Again, with the proposed ConvNet based loop closure, the optimized trajectory in black line is much closer to the ground truth, shown in Figure 4(c). In total there are 47 keyframes, and the results are illustrated in Table I.

C. Run Time

With a Nvidia GTX1080 GPU of the Pascal architecture, it takes about 1s to process 6000 laser scan pairs. In our case, loop closure can be detected as long as the relative location of any pair of scans satisfies $\Delta x < 1.6, \Delta y <$



(a) (b) (c)

Fig. 4. (a) The simulation shows effective loop closure with the proposed solution. (b) The hardware setup of the UAV equipped with a Hokuyo UTM-30LX range finder and the VICON system. (c) The proposed approach significantly improve the localization accuracy when navigating a UAV.

$1.6, \Delta\theta < 1$. Therefore, take an example that keyframes are created once the robot travels for $1m$ or steers for $0.6rad$, our approach is able to perform brute-force loop detection for 6km per second. This is infeasible previously given the range measurements as the only input. Moreover, if some other searching strategies, like those taking position uncertainty into account, are employed, the proposed approach can process a huge map in a short time. Also, such a deep learning based algorithm can be even faster in the near future because of the rapid development of GPUs.

V. Conclusion

In this paper, we proposed a deep learning based framework to solve the loop closure problem for 2D LiDAR. To the best of our knowledge, this is the first work that performs scan matching and loop closure detection using ConvNets. According to experiments in simulation and on a UAV, our approach achieves the accuracy of 98.2% and false positive rate of around 0.001%, even in unseen and noisy environment. With a contemporary GPU, our approach is able to process 6000 pairs of laser scans per second. In addition, the visualization and analysis of ConvNets show inspiring results for understanding the types of features that are critical for geometrical perception.

Although the deep learning based framework achieves satisfying loop detection performance, the scan matching ConvNet is relatively inaccurate compared to traditional scan-to-scan or scan-to-map methods. In loop closure detection, any scan-to-scan approach always suffers from the intrinsic ambiguity that similar scan measurements may not belong to the same place. To utilize the modern ideas of scan-to-map and submap-to-submap, combining Recurrent Neural Networks (RNNs) or Long Short Term Memory (LSTM) Networks with ConvNets should be a promising direction.

Acknowledgements

This work was partially supported by a NUS start-up grant R-252-000-636-133.

References

[1] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. D. Reid, and J. J. Leonard, "Simultaneous localization and mapping: Present, future, and the robust-perception age," *arXiv preprint arXiv:1606.05830*, 2016.

[2] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. MIT press, 2005.

[3] M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit, *et al.*, "Fastslam: A factored solution to the simultaneous localization and mapping problem," in *Aaai/iaai*, 2002, pp. 593–598.

[4] S. Kohlbrecher, J. Meyer, O. von Stryk, and U. Klingauf, "A flexible and scalable slam system with full 3d motion estimation," in *Proc. IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*, November 2011.

[5] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2d lidar slam," in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, 2016, pp. 1271–1278.

[6] G. D. Tipaldi and K. O. Arras, "Flirt-interest regions for 2d range data," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, 2010, pp. 3616–3622.

[7] M. Himstedt, J. Frost, S. Hellbach, H.-J. Böhme, and E. Maehle, "Large scale place recognition in 2d lidar scans using geometrical landmark relations," in *Intelligent Robots and Systems (IROS), 2014 IEEE/RSJ International Conference on*, 2014, pp. 5030–5035.

[8] C. Stachniss, G. Grisetti, D. Hähnel, and W. Burgard, "Improved rao-blackwellized mapping by adaptive sampling and active loop-closure," in *Proceedings of the Workshop on Self-Organization of AdaptiVE behavior (SOAVE)*, 2004, pp. 1–15.

[9] J. Neira, J. D. Tardós, and J. A. Castellanos, "Linear time vehicle relocation in slam," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, 2003, pp. 427–433.

[10] M. Bosse and R. Zlot, "Map matching and data association for large-scale two-dimensional laser scan-based slam," *The International Journal of Robotics Research*, vol. 27, no. 6, pp. 667–691, 2008.

[11] ——, "Keypoint design and evaluation for place recognition in 2d lidar maps," *Robotics and Autonomous Systems*, vol. 57, no. 12, pp. 1211–1224, 2009.

[12] G. D. Tipaldi, L. Spinello, and W. Burgard, "Geometrical flirt phrases for large scale place recognition in 2d range data," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, 2013, pp. 2693–2698.

[13] E. Olson, "M3rsm: Many-to-many multi-resolution scan matching," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, 2015, pp. 5815–5821.

[14] K. Granstrom, J. Callmer, F. Ramos, and J. Nieto, "Learning to detect loop closure from range data," in *Robotics and Automation (ICRA), 2009 IEEE International Conference on*, 2009, pp. 15–22.

[15] A. Nicolai, R. Skeele, C. Eriksen, and G. A. Hollinger, "Deep learning for laser based odometry estimation," in *Deep Learning Workshop at Robotics: Science and Systems Conference 2016*, Oct 2016.

[16] M. Pfeiffer, M. Schaeuble, J. Nieto, R. Siegwart, and C. Cadena, "From perception to decision: A data-driven approach to end-to-end motion planning for autonomous ground robots," *arXiv preprint arXiv:1609.07910*, 2016.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[18] C. Kerl, J. Sturm, and D. Cremers, "Robust odometry estimation for rgb-d cameras," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, 2013, pp. 3748–3754.