Preprints of the 21st IFAC World Congress (Virtual)
Berlin, Germany, July 12-17, 2020

Deep-Learning-based Relocalization in Large-Scale outdoor Environment

Shikuan Yu, Fei Yan, Wenzhe Yang, Xiaoli Li, Yan Zhuang

S. Yu, F. Yan, W. Yang and Y. Zhuang are with the Key Laboratory of Intelligent Control and Optimization for Industrial Equipment of Ministry of Education and the School of Control Science and Engineering, Dalian University of Technology, Dalian 116024, China (e-mail: yushikuan@mail.dlut.edu.cn; fyan@dlut.edu.cn; ywz@mail.dlut.edu.cn; zhuang@dlut.edu.cn). X. Li is with the Faculty of Information Science Beijing University of Technology, Beijing 100083, China (e-mail:lixiaolibjut@bjut.edu.cn).

Abstract: For the issue of relocalization, this paper proposed a deep-learning-based method for outdoor large-scale environment. In the first step, we projected a 3D Light Detection and Ranging(LiDAR) scan onto three 2D images from top to bottom. Then a densenet-based neural network structure was designed to regress a 4-DOF robot pose. These images are then stacked together, fed into the proposed DCNN architecture, and the output is the predicted robot pose. Extensive experiments have been conducted in practice with a real mobile robot, verifying the effectiveness of the proposed strategy. Our network can obtain approximately $3.5m$ and $4°$ accuracy outdoors.

Keywords: 3D LiDAR Scan, DCNN, Relocalization, Mobile Robot, Outdoor Environment

1. INTRODUCTION

In the past few decades, the research towards outdoor self-driving robots has picked up a staggering pace (see Lingemann et al. [2005], Csorba [1997]). The ability to acquire accurate pose information of the robot in real-time is the premise and basis for the robot to perform various tasks. Relocalization is crucial for an outdoor robot in navigation as well as other tasks.

Most strategies proposed over the past years for robot relocalization outdoors are basically based on Feature Point Matching (see Rusu et al. [2008], Rusu et al. [2010]). Methods based on local point cloud feature extraction were proposed in Belongie et al. [2001] and Tombari et al. [2010], these methods are used for scene recognition (see Steder et al. [2011]) and closed-loop detection through similarity metrics. The feature extraction method based on the global point cloud extracts the global descriptor of the point cloud from the point cloud in the global coordinate system (see Aldoma et al. [2011], Wohlkinger and Vincze [2011], Muhammad and Lacroix [2011]). Compared with the word bag method, the descriptor does not need to perform local keypoint detection, so the calculation is faster.

PoseNet (see Kendall et al. [2015]) was proposed in 2015 and was considered as a state-of-the-art method to regress a robot pose. In Kendall et al. [2015], a network structure based on GoogLeNet (see Szegedy et al. [2015]) was used to estimate a 6-DOF pose. However, it suffers from insufficient illumination and costs a heavy computational burden. Similar idea is also introduced in Kuse and Shen [2019], which is able to recover from complicated kidnap scenes and failures in real-time.

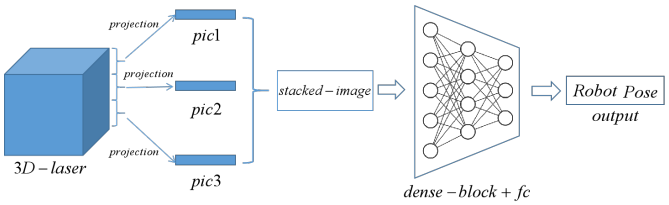

Fig. 1. System overview of the proposed strategy. We converted 3D LiDAR scan into three 2D images, and stacked them together, which was used as the input of a DCNN to predict a robot pose.

To get the robot pose when robot restarts from any point in a large-scale outdoor environment while consuming less computing resources, we proposed a convolutional neural network (DCNN) architecture based strategy for the task of relocalization in this paper. Recently, DCNN based methods have lead to breakthroughs in several vision tasks, such as classification (see Simonyan and Zisserman [2014], He et al. [2016a]), detection (see Ren et al. [2017], Redmon et al. [2016]) and segmentation (see Long et al. [2015], Chen et al. [2017a], Ronneberger et al. [2015]). Majority of these methods are based on camera images and few methods have focused on using 3D LiDAR scans (see Wu et al. [2017], Zhou and Tuzel [2017]).

Our proposed network structure takes a stacked image as the input and regresses a 4-DOF robot pose relative to a scene. Fig.1 demonstrates the proposed strategy frame-

⋆ This work was supported by the National Natural Science Foundation of China under Grant U1913201 and Grant 61503056.

work. It consists of a 3D LiDAR scan preprocessing algorithm and a dense blocks (see Huang et al. [2016]) based robot pose estimation. The laser preprocessing method is designed to convert one frame of 3D LiDAR scan into three 2D-images, these images are then stacked into a multi-channel image, which passed through our proposed DCNN for pose regression(as shown in Fig.1). The DCNN is designed to extract features from multi-channel images. The last convolutional block is followed by fully connected layers, and the output presents the predicted robot pose. Our main contributions are as follows:

- A method to project 3D LiDAR scan into stacked multi-channel images.
- A DCNN structure to regress robot pose outdoors.

The rest of this paper is organized as follows. Section II reviews related work. Our method is presented in Section III. Experimental results are given in Section IV. The conclusions are drawn in the last.

2. RELATE WORKS

Recent years have witnessed the efforts made by researchers to optimize the performance of robot relocalization outdoors. Multiview 2D projection (M2DP) was proposed in He et al. [2016b], which projects the point cloud in the global coordinate system to the 2D plane and extracts a 192-dimensional feature vector. The similarity between the two scenes is judged by the cosine between the 192-dimensional vectors of the two scenes. Giseop Kim et al. proposed the Scan Context (see Kim and Kim [2018]) algorithm, which was based on the non-histogram global descriptor of the laser point cloud as the basis for judging the similarity of the scenes. Dube et al. [2017] proposed the SegMatch algorithm, which extracted and described the segmentation block in a 3D point cloud, then matched it with the segmentation block in the traversed scenes, and used the steps of geometric verification to find the closed-loop candidate. ORB-SLAM2 (see Mur-Artal et al. [2015]) based robot outdoor relocalization algorithm was used and improved in Mur-Artal and Tardós [2017a], Engel et al. [2014], Yang et al. [2016], Mur-Artal and Tardós [2017b]. Orb-slam based algorithms could extract more observation features and those algorithms have better robustness. However, Orb-slam based algorithms not only consume a lot of computing resources but also are affected by insufficient illumination.

Most of the Deep Learning-based robot relocalization methods are vision-based (see Kuse and Shen [2019], Zhou et al. [2017]). Other machine learning techniques were employed for the loop-closure problem (see Yin et al. [2018], Chen et al. [2017b]). The PointNet architecture proposed by Qi et al. Charles et al. [2017] is a popular choice for learning from unordered pointcloud. A multi-layer perceptron was proposed to learn features from individual points and then use an asymmetric function to combine features learned from points, as a global representation. PointNet++ (see Qi et al. [2017]) was also proposed by them as an extending PointNet. The extension included hierarchical learning, where a set of points were sampled from the input point set and then points in the neighborhood of the centroids are grouped together, which is then followed by the PointNet architecture. Paper Dewan and Burgard [2019] proposed a deep convolutional neural network for the semantic segmentation of a LiDAR scan into four classes, their architecture was based on dense blocks and efficiently utilizes depth separable convolutions to limit the number of parameters while still maintaining state-of-the-art performance. Paper Kendall et al. [2015] introduced a novel algorithm named PoseNet based on a convolutional neural network, this strategy regressed a 6-DOF camera pose from a single RGB image. A novel Deep Learning-based relocalization method to extract the features of LiDAR data was introduced in Cao et al. [2017], this method classified these features in order to reduce the randomness of the relocalization, and this method could avoid some limitation of manual features compared with other methods based on matching the manual feature points.

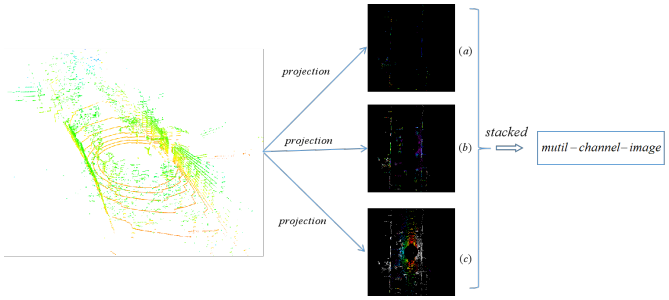

Fig. 2. Three RGB-images converted from a set of 3D LiDAR scan. These three pictures are all projections of the scan from top to bottom but represent different heights. We just keep point clouds within $1.5m$ above and below the LiDAR. (a), (b) and (c) represent those point cloud data in the range of$[0.5, 1.5)$, $[-0.5, 0.5)$, $[-1.5, -0.5]$ from the LiDAR respectively.

3. STRATEGY PROCEDURE

This section presents the proposed deep-learning-based algorithm in detail. This part is used to predict the global robot pose outdoors. We demonstrate the method to convert 3D LiDAR scan firstly, then, the network structure is introduced. The cost function designed for training the DCNN is also described.

3.1 3D LiDAR Scan Projection

3D LiDAR scan is used as the observation data in the outdoor robot research field widely. Its robustness and the characteristic that is not affected by insufficient illumination make it used extensively. As we know, the HDL-64E lidar sensor gets up to 2.2 million points per second, the HDL-32E lidar sensor gets 1.39 million points per second, and the VLP-16 lidar sensor gets 0.3 million points per second. So many point clouds contain enough feature points to complete the outdoor relocalization tasks of the robots, but processing such a large amount of point cloud data consumes huge computing resources. So the first step of our method is to project the scan onto 2D images and each such image encodes a specific modality. These images are then stacked together and passed through our proposed DCNN for robot relocalization outdoors.

The VLP-16 Lidar sensor features up to 16 lasers across a 30° vertical field of view and a 360° horizontal field of view.

Table 1. Configuration of the DCNN

conv-blocks	Kernel1	Kernel2	Kernel3	Nunber of Channerls
conv-blocks0	7	*	*	9
conv-blocks1	7	5	*	64
conv-blocks2	5	5	*	128
conv-blocks3	5	5	3	256
conv-blocks4	3	3	3	512
conv-blocks5	3	3	3	512

The height of our robot is $1.5m$, so we just keep the point data within the field of $[-1.5, 1.5]$ in verticality. As shown in Fig.2, a single-set 3D LiDAR scan is converted into three RGB-images (a), (b), (c), which represent the height of $[0.5, 1.5)$, $[-0.5, 0.5)$, $[-1.5, -0.5]$ relative to LiDAR respectively. All of these RGB-images have the size of $448 * 448 * 3$, which means Width*Height*Channel. (a), (b) and (c) were stacked into one image with nine channels in turn. This 9-channel image is used as the input of the DCNN.We have tested the results when one frame of laser data is processed into multiple RGB images. The results show that when the image processed by one frame of laser is less than 3, there are many missing features of the point cloud, and the prediction pose accuracy is low. When the image processed by a frame of laser is higher than 3, the pose predicted by the network will be saturated with precision.

3.2 CNN based Pose Regression

Network structure We proposed a convolutional DCNN architecture for the task of robot relocalization outdoors. In the reason that convolutional layers can be interpreted as transforming inputs into feature representation effectively, we designed a convolutional neural network to map multi-channel images to robot poses. Our network is comprised of an encoder for learning the features required for the task and fully connected layers for mapping the learned distributed features representation to the sample tag space. Our architecture is similar to other DCNN architecture proposed for the task of semantic segmentation (see Jegou et al. [2017], Shelhamer et al. [2017]), which is based on dense blocks and is shown in Fig.3. Our network structure consists of 6 convolutional blocks, $conv-blocks0$ includes one convolutional layer and one pooling layer, $conv-blocks1$ and $conv-blocks2$ include two convolutional layers and one pooling layer, $conv-blocks3$, $conv-blocks4$ and $conv-blocks5$ include three convolutional layers and one pooling layer. The output of the last fully connected layers was a vector with the size $1 * 4$ including pose and orientation.

The configuration of the DCNN is outlined in TABLE 1. The input of the network is a 9-channel image converted from a 3D LiDAR scan. As shown in Fig.3, there are six convolutional blocks, each of which is normalized and has a rectified linear unit (leaky-ReLU Xu et al. [2015]) activation. We connected all the blocks to each other, specifically, each block will accept all the blocks in front of it as its additional input. In order to ensure that the input of each block is the same as the size of the feature map of its extra input, we pass the input of each block through a resize module. The resize module consists of a convolutional layer and a pooling layer. Following the last convolutional layer, three fully connected layers are used to extract the global feature and regress the robot pose.

Table 2. Parameters of two scenes

DUT	Scene Features	Road Length	Training	Test
Area A	Architecture	470m	4607	965
Area B	Vegetation	420m	4396	685
Total Area	*	1590m	12830	4630

Cost Function The cost function is designed to train the DCNN so that its predicted poses are close to the ground truth. In this paper, the network outputs a pose vector V_o, given by a 3D robot position $[x, y, z]$ and orientation represented by angle θ:

$$V_o = [x, y, z, \theta] \tag{1}$$

We chose yaw angles as our orientation representation because arbitrary 1D values is a simpler process than the 4D values normalization and normalization required of rotation matrices. Specifically, the cost function contains two parts: a pose error and an angle error.

3.2.2.1. Pose Error Pose error is formulated as the Euclidean distance between the predicted positions and the ground truths. The function aims to minimize the Euclidean distance between the ground truth and the predicted position by the network. The error function is shown as follows:

$$L_{err} = \frac{1}{n}\sum_{i=1}^{n}\{||\hat{x} - x||_2^2 + ||\hat{y} - y||_2^2 + \lambda||\hat{z} - z||_2^2\} \tag{2}$$

where $||\cdot||_2^2$ is the 2-norm, n is the batch size, and λ is a factor to balance the weight of x, y and z for the reason that the robot does not have too much positional fluctuation in the vertical direction.

3.2.2.2. Orientation Error The method to calculate the orientation error in this paper is followed. The way is shown in Formula.3, we used the form of angle to calculate the error between the predicted value and the ground truth.

$$\Theta = \frac{1}{n}\sum_{i=1}^{n}\beta||\hat{\theta} - \theta||_2^2 \tag{3}$$

where $||\cdot||_2^2$ is the 2-norm, n is the batch size, θ means the angle representing the orientation, and β is a factor to balance the weight of positions and orientations.

4. EXPERIMENT RESULTS

4.1 Experimental Scene and Equipment

As shown in Fig.5, a large-scale scene was selected to prove the performance of our algorithm. As shown in Table.2, the area marked by the blue dotted line and yellow dotted line indicate the *Area A* and the *Area B* respectively. The *Total Area* means the all paths labeled by black lines. Fig.6 shows the robot we used in the experiment, we choose a mobile robot mounted with a VLP-16 3D laser scanner. The VLP-16 Lidar sensor features up to 16 lasers across

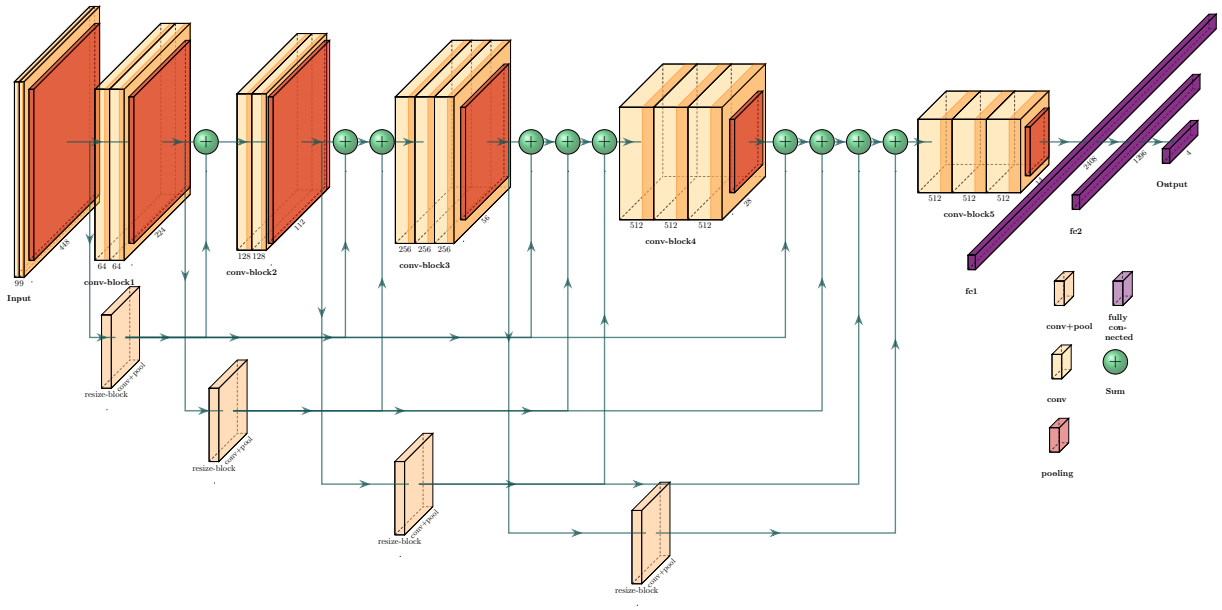

Fig. 3. The architecture of the CNN network in the proposed strategy. The network structure is based on dense-blocks and consists of fourteen convolutional layers with three fully connected layers. The output of the network is the 4-DOF posture including pose and orientation.

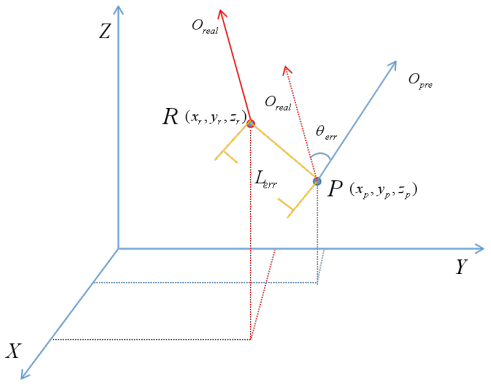

Fig. 4. Illustration of the way to calculate the errors between the predicted pose and the ground truth. Point R (x_r,y_r,z_r) and point P $(x_p, y_p, z_p,)$ represent the real robot position and the predicted robot position respectively, arrow O_{real} and O_{pre} represent the real robot orientation and the predicted robot orientation respectively. L_{err} means the euclidean distance between the predicted position and the ground truth, θ means the robot yaw angle.

a 30° vertical field of view and 360° horizontal field of view. The maximum linear and angular velocities of the robot are 1.0m/s and 1.0rad/s respectively. The densenet-based-model was trained using data collected from a true environment and tested on a real mobile robot in the same environments. We chose TensorFlow (see Abadi et al. [2015]) to implement our complete network architecture. Our dataset consists of 12830 scans for training and 4630 scans for testing. We used Huber loss and used Adam optimizer with learning rate $1e^{-3}$ with decay of $2e^{-4}$ and batch size of 16. Our network was trained using an Nvidia GTX2080ti GPU.

Fig. 5. Areas we choose to test our method and PoseNet. The area marked by the blue dotted line and yellow dotted line indicate the *Area A* and the *Area B* respectively. The *Total Area* means the all paths labeled by black lines.

4.2 Results in Real Environments

To prove the effectiveness of our strategy, we conducted repeated experiments with an IPC(Industrial Personal Computer) (Intel Core TM Dual Core i7H-6500 and 6GB RAM) without the GPU in the areas (shown in Fig.6). We tested the effectiveness of our strategy and compared our method with PoseNet both in daytime and nighttime.

Table 3. Results in Real Environment

	Training Set	Testing Set	Average Error in Daytime	Average Error in Nighttime
PoseNet	12830	4630	3.86m, 3.84°	Disabled
Ours	12830	4630	3.24m, 3.67°	3.78m

TABLE.3 shows the compares results of PoseNet and our method in the *Total Areas*. The 3D LiDAR scans used in our method and the images used to train PoseNet correspond to each other on the time stamp, so we had

Fig. 6. The setup of the mobile robot equipped with a VLP-16 Lidar sensor. The computer is IPC(Industrial Personal Computer) (Intel Core TM Dual Core i7H-6500 and 6GB RAM) without the GPU, the GPS is NovATel OEM718D.

the same number of training sets to train PoseNet and our model. Finally we collected the same number of test sets to test the both models. The results show that our model has higher precision than the PoseNet during the daytime period and can still maintain the same accuracy at night, while the PoseNet cannot be used at night.

5. CONCLUSION AND FUTURE WORK

This paper proposed a method to relocalize robots in Large-Scale outdoor Environment based on Deep-Learning. Firstly, a laser scan data preprocessing method is proposed to project a 3D LiDAR scan onto three 2D images from top to bottom, and a densenet-based neural network structure was designed to regress a 4-DOF robot pose. Through extensive evaluation in real environment, we demonstrated that ample pose information was preserved in such networks and it is feasible to relocalization relying on the projection of 3D LiDAR scan.

The reason why our strategy can realize relocalization is because the topology of the environment contained in the high-dimensional point cloud is mapped to the robot pose space by the neural network. The laser data preprocessing step does not destroy the topology of the point cloud. The regression characteristics of the neural network make sure that the point cloud features can be mapped into the continuous robot pose space.

In future work, we will consider the issue of sensor fusion. We consider using fused data from cameras and 3D lasers as the input of neural networks, which will increase the input characteristics of neural networks. We will continue to improve the accuracy of relocalization and will strive to simplify the structure of the neural network.

REFERENCES

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., et al. (2015). Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv: Distributed, Parallel, and Cluster Computing*.

Aldoma, A., Vincze, M., Blodow, N., Gossow, D., Gedikli, S., Rusu, R.B., and Bradski, G.R. (2011). Cad-model recognition and 6dof pose estimation using 3d cues. 585–592.

Belongie, S.J., Malik, J., and Puzicha, J. (2001). Shape matching and object recognition using shape contexts. Technical Report UCB/CSD-01-1128, EECS Department, University of California, Berkeley.

Cao, J., Zeng, B., Liu, J., Zhao, Z., and Su, Y. (2017). A novel relocation method for simultaneous localization and mapping based on deep learning algorithm. *Computers & Electrical Engineering*, 63, 79–90.

Charles, R.Q., Su, H., Kaichun, M., and Guibas, L.J. (2017). Pointnet: Deep learning on point sets for 3d classification and segmentation. 77–85.

Chen, L., Papandreou, G., Schroff, F., and Adam, H. (2017a). Rethinking atrous convolution for semantic image segmentation. *arXiv: Computer Vision and Pattern Recognition*.

Chen, Z., Jacobson, A., Sünderhauf, N., Upcroft, B., Liu, L., Shen, C., Reid, I., and Milford, M. (2017b). Deep learning features at scale for visual place recognition. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 3223–3230. IEEE.

Csorba, M. (1997). *Simultaneous localisation and map building*. Ph.D. thesis, University of Oxford Oxford.

Dewan, A. and Burgard, W. (2019). Deeptemporalseg: Temporally consistent semantic segmentation of 3d lidar scans. *arXiv: Robotics*.

Dube, R., Dugas, D., Stumm, E., Nieto, J.I., Siegwart, R., and Cadena, C. (2017). Segmatch: Segment based place recognition in 3d point clouds. *international conference on robotics and automation*, 5266–5272.

Engel, J., Schöps, T., and Cremers, D. (2014). Lsd-slam: Large-scale direct monocular slam. In *European conference on computer vision*, 834–849. Springer.

He, K., Zhang, X., Ren, S., and Sun, J. (2016a). Deep residual learning for image recognition. 770–778.

He, L., Wang, X., and Zhang, H. (2016b). M2dp: A novel 3d point cloud descriptor and its application in loop closure detection. 231–237.

Huang, G., Liu, Z., Weinberger, K.Q., and Der Maaten, L.V. (2016). Densely connected convolutional networks. *arXiv: Computer Vision and Pattern Recognition*.

Jegou, S., Drozdzal, M., Vazquez, D., Romero, A., and Bengio, Y. (2017). The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation. 1175–1183.

Kendall, A., Grimes, M., and Cipolla, R. (2015). Posenet: A convolutional network for real-time 6-dof camera relocalization. In *Proceedings of the IEEE international conference on computer vision*, 2938–2946.

Kim, G. and Kim, A. (2018). Scan context: Egocentric spatial descriptor for place recognition within 3d point cloud map. 4802–4809.

Kuse, M. and Shen, S. (2019). Learning whole-image descriptors for real-time loop detection andkidnap re-

covery under large viewpoint difference. *arXiv preprint arXiv:1904.06962*.

Lingemann, K., Nüchter, A., Hertzberg, J., and Surmann, H. (2005). High-speed laser localization for mobile robots. *Robotics and autonomous systems*, 51(4), 275–296.

Long, J., Shelhamer, E., and Darrell, T. (2015). Fully convolutional networks for semantic segmentation. 3431–3440.

Muhammad, N. and Lacroix, S. (2011). Loop closure detection using small-sized signatures from 3d lidar data. 333–338.

Mur-Artal, R., Montiel, J.M.M., and Tardos, J.D. (2015). Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5), 1147–1163.

Mur-Artal, R. and Tardós, J.D. (2017a). Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Transactions on Robotics*, 33(5), 1255–1262.

Mur-Artal, R. and Tardós, J.D. (2017b). Visual-inertial monocular slam with map reuse. *IEEE Robotics and Automation Letters*, 2(2), 796–803.

Qi, C.R., Yi, L., Su, H., and Guibas, L.J. (2017). Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv: Computer Vision and Pattern Recognition*.

Redmon, J., Divvala, S.K., Girshick, R.B., and Farhadi, A. (2016). You only look once: Unified, real-time object detection. 779–788.

Ren, S., He, K., Girshick, R.B., and Sun, J. (2017). Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6), 1137–1149.

Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. 234–241.

Rusu, R.B., Blodow, N., Marton, Z., and Beetz, M. (2008). Aligning point cloud views using persistent feature histograms. 3384–3391.

Rusu, R.B., Bradski, G.R., Thibaux, R., and Hsu, J.M. (2010). Fast 3d recognition and pose using the viewpoint feature histogram. 2155–2162.

Shelhamer, E., Long, J., and Darrell, T. (2017). Fully convolutional networks for semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4), 640–651.

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv: Computer Vision and Pattern Recognition*.

Steder, B., Ruhnke, M., Grzonka, S., and Burgard, W. (2011). Place recognition in 3d scans using a combination of bag of words and point feature based relative pose estimation. 1249–1255.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1–9.

Tombari, F., Salti, S., and Stefano, L.D. (2010). Unique signatures of histograms for local surface description. 356–369.

Wohlkinger, W. and Vincze, M. (2011). Ensemble of shape functions for 3d object classification. 2987–2992.

Wu, B., Wan, A., Yue, X., and Keutzer, K. (2017). Squeezeseg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3d lidar point cloud. *arXiv: Computer Vision and Pattern Recognition*.

Xu, B., Wang, N., Chen, T., and Li, M. (2015). Empirical evaluation of rectified activations in convolutional network. *arXiv preprint arXiv:1505.00853*.

Yang, S., Song, Y., Kaess, M., and Scherer, S. (2016). Pop-up slam: Semantic monocular plane slam for low-texture environments. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 1222–1229. IEEE.

Yin, P., He, Y., Xu, L., Peng, Y., Han, J., and Xu, W. (2018). Synchronous adversarial feature learning for lidar based loop closure detection. In *2018 Annual American Control Conference (ACC)*, 234–239. IEEE.

Zhou, T., Brown, M., Snavely, N., and Lowe, D.G. (2017). Unsupervised learning of depth and ego-motion from video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1851–1858.

Zhou, Y. and Tuzel, O. (2017). Voxelnet: End-to-end learning for point cloud based 3d object detection. *arXiv: Computer Vision and Pattern Recognition*.