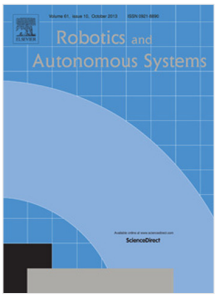
Journal Pre-proof



LiDAR-based vehicle localization on the satellite image via a neural network

Mengyin Fu, Minzhao Zhu, Yi Yang, Wenjie Song, Meiling Wang

PII: S0921-8890(19)30520-2
DOI: https://doi.org/10.1016/j.robot.2020.103519
Reference: ROBOT 103519

To appear in: *Robotics and Autonomous Systems*

Received date : 29 June 2019
Revised date : 4 March 2020
Accepted date : 30 March 2020

Please cite this article as: M. Fu, M. Zhu, Y. Yang et al., LiDAR-based vehicle localization on the satellite image via a neural network, *Robotics and Autonomous Systems* (2020), doi: https://doi.org/10.1016/j.robot.2020.103519.

LiDAR-based Vehicle Localization on the Satellite Image via a Neural Network

Mengyin Fu[1], Minzhao Zhu[1], Yi Yang[*,1], Wenjie Song[1], Meiling Wang[1]

***Abstract*—We present a novel method to localize the vehicle on an easily accessible geo-referenced satellite image based on LiDAR. We first design a neural network to extract and compare the spatial-discriminative feature maps of the satellite image patch and the LiDAR points, and obtain the probability of correspondence. Then based on the outputs of the network, a particle filter is used to obtain the probability distribution of the vehicle pose. This method can use LiDAR points and any type of odometry as input to localize the vehicle. The experimental results show that our model can generalize well on several datasets. Compared with other methods, ours is more robust in some challenging scenarios such as the occluded or shadowed area on the satellite image.**

***Keywords*—localization, deep learning, LiDAR, satellite image matching.**

I. Introduction

Vehicle localization is a key issue in the field of autonomous driving, and one of the predominant methods of estimating the vehicle position is via Global Navigation Satellite System (GNSS). However, GNSS may suffer from a decrease in positioning accuracy, or even a failure, when its signal is blocked or reflected by surrounding buildings (the so-called urban canyon effect).

In order to improve the reliability and accuracy of the localization system, a prior map can be used as a complement to the GNSS. One way to acquire the prior map is to use Mobile Mapping System [1]. In this system, a vehicle equipped with the Light Detection and Ranging (LiDAR) or other sensors travels along the road to construct the map [2]. The process of constructing the map can be defined as a Simultaneous Localization and Mapping (SLAM) problem in which a vehicle constructs the map of an unknown environment while simultaneously locates itself within the map [3]. However, due to the huge amount of work required to build the map, the cost of implementing this process in a large-scale environment is very high.

In contrast, satellite images are easy to access, and they almost cover the world. Therefore, many approaches seek to use satellite image as the prior map. These works register ground-level image or LiDAR points acquired by the vehicle to the satellite map in order to obtain the geo-referenced position from the map. However, the registering process is challenging because of sensor modality, time of acquisition, varying viewpoints, illuminations change, frequent object occlusions, motions, etc [4]. Compared with cameras, LiDAR is less sensitive to light, so some works use LiDAR as a sensor to localize the vehicle on the satellite image. These traditional methods are not particularly generalizable because due to illumination changing in different regions or time of the satellite image, it is hard to find the parameter of image preprocessing that performs well in various scenarios. Besides, trees, buildings or bridges cause occlusions or shadows on the satellite image, bringing about noise which may lead to a mismatch between satellite image and the LiDAR grid-map.

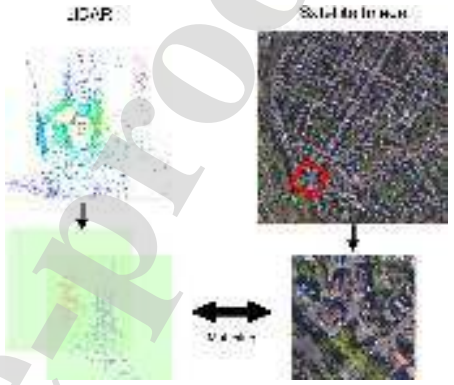


Fig. 1. Our approach localizes the vehicle by comparing the LiDAR points and the geo-referenced satellite image.

In this paper, we propose a learning-based method in order to address the challenges mentioned above. It uses a frame of LiDAR points to estimate the position and heading of the vehicle on a satellite image. We design a neural network that learns to compare the spatial-discriminative feature maps of the satellite image patch and the LiDAR points, and outputs their probability of correspondence. This output then serves as observations of a particle filter that estimates a distribution of the vehicle's pose. We also present a way to train this neural model for better performance. We evaluate our method with several datasets and the result shows that the learned neural model generalizes well to different environments. Compared with other methods, ours can achieve stable localization even in some challenging scenarios mentioned above (occlusion or shadowed area on the satellite image).

II. Related Work

There are many works seeking to achieve localization by matching satellite images with the sensor data from the vehicle. Many of these works are based on a particle filter framework [5] that obtains an estimation of the vehicle's pose. The main difference among these works is the method to obtain the weights of particles.

[1]School of Automation, Beijing Institute of Technology, Beijing, China
[*]Corresponding author: Y. Yang, Email: yang_yi@bit.edu.cn

Many approaches compare the image collected by the camera installed on the vehicle with the satellite image patches at the location of the vehicle. For example, camera images are projected to the top-view and extract features such as SURF [6] or SIFT [7] to match with the satellite image. Some works extract features from the original camera image, using a neural network [8, 9, 10] or associating with vertical structures [11]. In addition to image feature points, it is also feasible to match the edge or area of the processed overhead-view images and processed satellite images, such as the threshold image [12], the edge map of sidewalks [13] and the semantic segmentation results [14, 15, 16].

Methods using cameras can achieve satisfactory results under ideal lighting conditions but do not perform well in dim or backlighting conditions. LiDAR, on the other hand, is less sensitive to light, and some works use LiDAR as a sensor to localize the vehicle on the satellite image. Since it is difficult to match the satellite image with the raw LiDAR points directly, a preprocessing procedure is generally used to convert the 3D LiDAR points into a 2D image. These methods can be sorted into two categories based on the treatments of grid-maps they employ, namely, geometric-edge-based methods and intensity-based methods.

Edge-based methods compare grid-map with processed satellite images containing the contours of buildings or road edges, etc. Kmmerle R et al. [17] transfer 3D points to the overhead view to get a grid-map and then match with edges of satellite image generated by Canny edge extraction. Instead of using satellite image, some works choose to match the LiDAR grid-map with an available map such as OpenStreetMap [18, 19]. In these works, they match the grid-map to the building contours or the roads on the map. Similar to these methods, our method also compares grid-map to static objects on the map such as trees, roads, and buildings. The difference is that we train a neural network to directly match the grid-map with the original satellite image instead of a processed map.

Intensity-based methods use echo intensity of the laser to help match grid-maps with the satellite image. Paula Veronese et al. [20] propose a system to localize a vehicle on the urban environment by matching satellite images to the short-term history re-emission maps built from the infrared reflectance information of the LiDAR. In their system, Normalized Mutual Information(NMI) is used to compare re-emission maps and satellite images. Javanmardi M et al. [21] propose a method that extracts road features such as road markings in the 3D point cloud and aerial data, and then performs registration of the LiDAR data to the aerial map using the normal distribution transform. These two methods use the intensity information of the multi-frame point cloud accumulated according to dead-reckoning, while our method only uses single frame LiDAR points, and consequently lowers the accuracy requirement of the odometry method.

When performing LiDAR-based localization on raw satellite images, traditional methods such as [17],[20] can achieve localization with high accuracy in some scenarios, but the possible illumination changing in different regions of the satellite image makes it hard to find the parameter of image preprocessing that generalize well across various scenarios. For example, the parameter of the Canny edge detector tuned in one scenario might not be suitable for another. Besides, occlusions caused by trees, buildings or bridges are likely to render part of the road structure invisible from the satellite image. Excessive occlusion may disable intensity-based methods from finding intensity correlations between satellite image and the LiDAR grid-map. At the same time, these occlusions might generate extra edges, which brings more noise and affects the performance of edge-based methods. The same holds true for shadows in satellite images. Compared with traditional methods, our learning-based approach extracts deep features which should be more stable and helpful to guide the judgment of matching in spite of the misleading edges of occluders or shadows, thus our neural model can generalize better and would be less affected by shadows and occlusions compared with traditional methods.

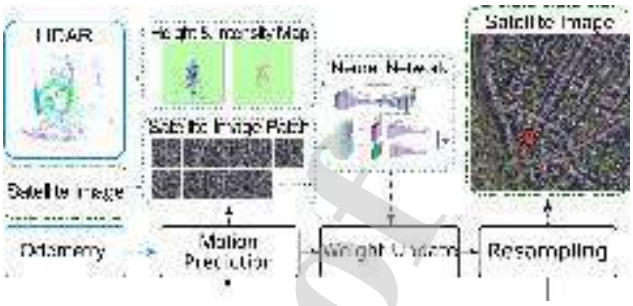


Fig. 2. Flow chart of our localization method. In our method, a particle filter is employed for the estimation of the pose of the vehicle. The weight of the particles is obtained by comparing the similarity between the satellite image patch and the LiDAR grid-map by a neural network.

III. PROPOSED METHOD

As shown in Fig. 2, our method takes a single scan of LiDAR and the odometry data (e.g. from IMU, a wheel odometer or the LiDAR odometry) as input. A georeferenced satellite image is the only prior information in this method. The output of our method is an estimation of the vehicle's position and orientation relative to this map. Our method can be divided into two main parts: a neural network for judging the matching degree of a satellite image patch with the LiDAR points, and a particle filter localization algorithm [5] that takes multiple outputs of the neural network as the observation to obtain a distribution of the pose of the vehicle. We now give a detailed description of our method.

A. Neural Network

Our neural network matches LiDAR points to the corresponding satellite image patch. In order to feed into the network, the point cloud is projected to the overhead view to build a grid-map. This grid-map is 144×144 m with 0.3 m per cell (i.e. 480×480 cells). It consists of 2 channels: the height channel and the intensity channel. Each cell in the map stores the height of the highest point falling within the cell's range as well as the reflection intensity value of that point. The

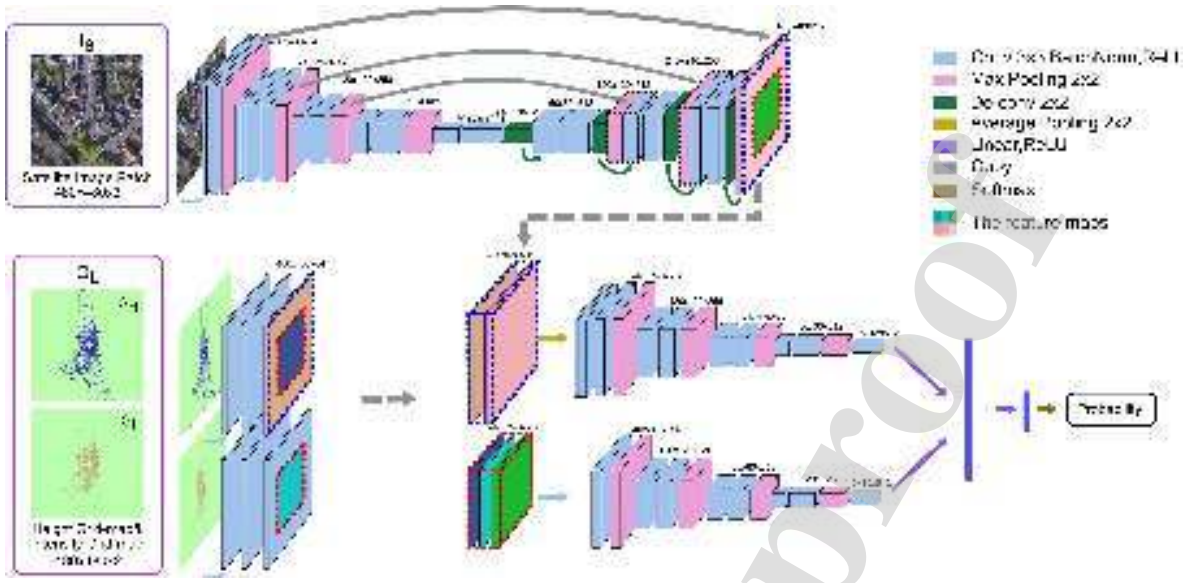


Fig. 3. The architecture of our network. Each cube in the figure represents a layer. The color of the cube or arrow represents the upcoming operation of the layer (shown in the upper right corner of the figure). The kernel size of convolutional layers is 3×3, the stride is 1 and padding is 1. The kernel size of pooling layers and de-convolutional layers are 2×2, the stride is 2. Our network consists of two cascade convolutional neural networks. The first part of the network extracts the feature maps(blue dotted boxes) of the satellite image and the LiDAR grid-map. These feature maps and their central part(red dotted boxes) are regrouped into two feature maps(the blue and red dotted cube) according to their sizes.

satellite image patch is also 480×480 pixels, and it is cropped from the satellite image of the whole interested region.

Our network takes a pair of satellite image patch I_S and a grid-map $G_L = \{G_H, G_I\}$ as input, where G_H, G_I denote the height channel and the intensity channel respectively. Our network consists of two cascade convolutional neural networks (see Fig. 3). The first part of the network is designed to generate the feature maps of the I_S and G_L in two parallel branches separately. The satellite image branch of the network is adapted from the U-Net [22]. The last three convolution layers of the U-Net are removed so that the high-level and low-level feature maps in the network can be output directly. Though high-level feature is helpful for determining whether I_S and G_L belong to the same place, spatial accuracy is limited due to the spatial invariance of the pooling layers [23]. As a result, the small spatial difference between I_S and G_L is difficult to distinguish when solely using the high-level feature map. In contrast, low-level feature map may tend to retain geometric information such as edges, which is more helpful for improving the spatial matching precision in a small range. This part of the network is fully convolutional, so the extraction of the satellite feature maps can be done offline. The grid-map branch consists of several convolutional and ReLU layers and outputs the low-level feature map of G_H and G_I separately.

The second part of our network is a classification network architecture with only one output. It consists of two parallel branches, each branch takes a certain region of the feature maps generated from the first part of the network as input. One of the branches receives the whole image region, and the other branch receives the central part of the image region. The first branch down-samples the feature maps before processes. The other branch receives the center-region of the feature maps which is cropped from the feature maps. The output of the two branches is concatenated and followed with two fully connected layers. This form of architecture combines two different resolution of the feature map, which could reduce the number of parameters needed with limited impact on the precision of matching compared with the typical one-branch network [24]. The whole-region-branch in the architecture has a wider receptive field on I_S and G_L. In contrast, the center-region-branch focuses more on the region near the vehicle (72 ×72 m) where the laser scanning lines are dense. After that, the fully connected layers serve as a decision function to consider both the result of the two branches. At the end of the network, a Softmax function is used to calculate the probability of matching (the network should output 1 if the two maps match).

B. Particle Filter

Although the neural network can match the LiDAR maps and the corresponding satellite image patch, it is possible that the matching hypothesis does not correspond to the correct location. Therefore, a particle filter [5] is used to efficiently choose the candidate satellite image patches and mitigate the noise of error matches. Particle filter can indicate a probabilistic distribution of the vehicle's pose by a set of weighted particles. Each particle $P_t^i = \{X_t^i, w_t^i\}$ includes a pose hypothesis $X_t^i = \{x_t^i, y_t^i, \psi_t^i\}$, and the particle's weight w_t^i. There are several reasons why only the horizontal position and heading are estimated in the particle filter. First, the satellite image is in a 2-D plane so it only helps to mitigate the 3-DoF pose error (horizontal position and yaw), but is not helpful for the other three dimensions (height, pitch and, roll). Second, since the vehicle moves on the plain ground and we assume there is no suspension, unbalanced tires, or vibration,

the localization on the two-dimensional plane is sufficient to meet its navigation needs.

During the initialization phase, particles are spread over the whole map or on the position given by information from external sources such as GNSS. At each time step, each particle is sampled from the prior distribution, which is computed from the motion model (depending on the selected odometry method) of the vehicle with Gaussian noise. In this paper, the motion model is formulated as

$$\begin{pmatrix} x_t^i \\ y_t^i \\ \psi_t^i \end{pmatrix} = \begin{pmatrix} x_{t-1}^i \\ y_{t-1}^i \\ \psi_{t-1}^i \end{pmatrix} + \begin{pmatrix} v_f \Delta t \cos \psi_{t-1}^i - v_l \Delta t \sin \psi_{t-1}^i \\ v_f \Delta t \sin \psi_{t-1}^i + v_l \Delta t \cos \psi_{t-1}^i \\ \Omega \Delta t \end{pmatrix} \tag{1}$$

where x_t^i, y_t^i represents the position in Universal Transverse Mercator (UTM) coordinate, ψ_t^i represents the heading angle, v_f, v_l represent the forward and leftward velocity, Ω represents the angular velocity. After that, we sample a 144×144 m satellite image patch centered on the position x^i, y^i with the orientation ψ^i for each particle. Then the weight of each particle is calculated using the output of the network

$$\tilde{w}_t^i = w_{t-1}^i \cdot Net(I_S^i, G_{H_t}, G_{I_t}) \tag{2}$$

where $\tilde{w}_t^i$ is unnormalized weight at this time step, w_{t-1}^i is the normalized weight of particle P_{t-1}^i at last time step, $Net(I_S^i, G_{H_t}, G_{I_t})$ is the output of the network, I_S^i is the satellite image patch of particle P_t^i. After having calculated and normalized the importance weights w_t^i, resampling is performed if the maximum value of $Net(I_S^i, G_{H_t}, G_{I_t})$ for all particles is greater than a threshold and the effective number of particles

$$\mathrm{N}_{eff} = \frac{1}{\sum_{i=1}^{n} w_t^{i\,2}} \tag{3}$$

is below a threshold. Finally, the average of the particles' pose is calculated to estimate the pose at this time step.

IV. Experiment

A. Network Training

We obtain the data from the KITTI dataset [25], collected by a vehicle driving through different scenes in Karlsruhe, Germany (The data is drawn from synced raw data instead of the odometry benchmark in KITTI). The vehicle is equipped with an OXTS-RT-3003 Integrated Navigation System, a Velodyne HDL-64E LiDAR and several other sensors. We sample 22 sequences as the training set, 10 sequences as the validation set. Satellite images at different times are drawn from Google Map which means there are multiple satellite images correspond to one sequence.

The training samples include positive pairs that G_L matches I_S and the negative pairs that G_L and I_S correspond to different places. For each frame of the LiDAR data, we obtain the ground truth position of the vehicle from fused GNSS/INS data and crop the patch from the satellite image. This image patch is used as a positive pair together with the LiDAR grid-map. We found that the performance of our method improves when the satellite image patches in the negative pairs are sampled similar to the particle distribution in the particle filter, as compared to randomly sampling on the whole satellite image. Therefore, we construct three different sets to sample the negative satellite image patches (see Fig. 5). In the first set, the satellite image patches are randomly sampled near the ground truth position of G_L. The role of this set is to improve the ability of the network to distinguish the minute position difference between G_L and I_S. The second set is sampled along the road near the ground truth position of G_L. This set is designed to let the network learn the difference between LiDAR grid-map and satellite image at different locations along the road. In the third set, the image patches are randomly sampled on the whole satellite image.

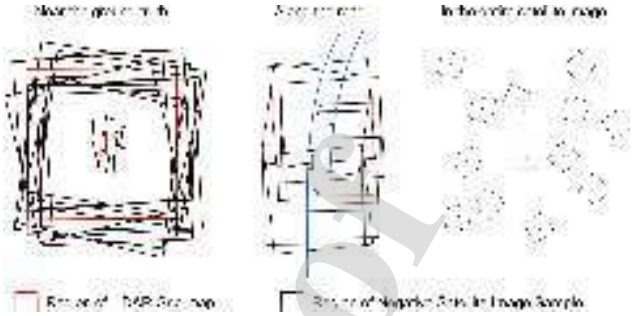


Fig. 5. The satellite image patches in the negative pairs are sampled in three ways. The first is random sampling near the ground truth position (left), the second is random sampling along the road (center), and the third is random sampling on the entire satellite image (right). The red box in the figure represents the range of the LiDAR grid-map, the black boxes represent the range of the satellite image patches, the blue line represents the roads.

The first set and the second set are randomly sampled from Gaussian distributions, and we determine the standard deviation parameters of the distributions as follows: we first try different distribution parameters to generate the dataset and train the network, then we adopt the trained models in the validation sequences and find the appropriate parameters with minimum variance of particle distribution. The choice of appropriate parameters requires further consideration of balancing between accuracy and robustness. In practice, if the standard deviation parameters are too high, the variance of the particle distribution would be large. On the other hand, if the standard deviation parameters are too low, the training of the neural network may be hard to converge and the method may be unstable. Finally, we choose the following parameters: the first set is randomly sampled with a standard deviation of 5 meters in both the east and north directions and 5 degrees in yaw angle near the ground-truth pose. The second set is sampled with a standard deviation of 15 meters in the direction along the road (we use the GNSS/INS data of several adjacent frames in the data set to approximate the direction of the road), 5 meters in the direction perpendicular to the road, 5 degrees in yaw angle. In addition, the negative satellite image samples are kept more than 2 meters apart from the ground truth position.

In order to augment training data, we use satellite images of different times and randomly rotate (with random rotate angle) or mirror satellite images and LiDAR point clouds. Overall, about 74,000 pairs of positive samples and 670,000 negative samples were collected for the training set, about

Fig. 4. Examples of occlusion and shadow scenes in satellite images.

21,000 positive samples and 190,000 negative samples were collected for the validation set. During training, we sample equally from positive and negative samples by using simple oversampling method.

We use the Cross-Entropy loss function to train our network. In the training phase, Adam [26] with learning rate 1e-7 are used to train the model. Dropout with the dropout rate 0.7 is used in the last 3 convolutional layers of the second part of our network. The size of the mini-batches in training and validation is 4. Weights are initialized randomly and trained from scratch in this experiment. The training is stopped after the loss in the verification set begins to increase for several epochs.

B. Experimental Setup

We evaluate our method on 4 sequences. KITTI-City is obtained from the city category in KITTI, and it is 1058 frame (105.8 sec), 238 m. KITTI-Residential is 4076 frame (407.6 sec), 3220 m. KITTI-Road is 817 frame (81.7 sec), 2446 m. We also collect a sequence that has 4300 frames (430.0 sec), 1399 m. Our vehicle is equipped with a Velodyne HDL-64E LIDAR and an Integrated Navigation System with 0.02-meter position accuracy and 0.06-degree heading accuracy. There is no overlap between the region of the training set and the test sequences in order to test the generalization ability of the neural network. Besides, Fig. 4 shows some challenging scenarios where there are occlusion and shadow in satellite images.

We construct two baselines to compare with our method. The first baseline builds a height grid-map from the overhead view after discarding the points with a height lower than a threshold (set as 1.7 m). Then the grid-map is compared with the edges of the satellite image generated by Canny edge extraction [17]. The second baseline builds a short-term history re-emission map and uses Normalized Mutual Information(NMI) to compare it with satellite images [20]. The parameters of the baselines are manually adjusted in the validation set.

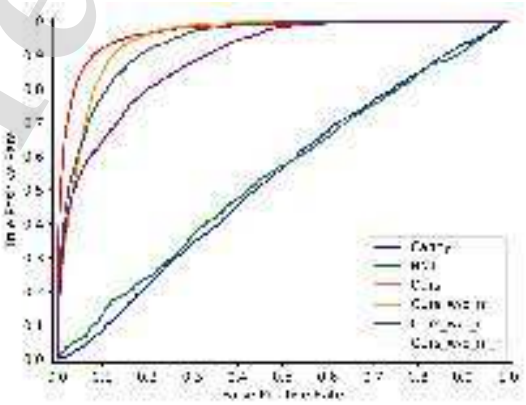


Fig. 6. The ROC curve of the baseline methods and our model on the test set. *Ours_w/o_m* represents training without mirror, *Ours_w/o_r* represents training without rotating the satellite image patches and grid-maps, *Ours_w/o_m_r* represents training without rotate and mirror.

We evaluate our method using two different odometry method. The first uses Inertial Navigation System (INS), the message we use in the experiment is the forward acceleration a_f, leftward acceleration a_l and angular velocity Ω. We obtain current velocity according to

$$\begin{aligned} v^t_{f_{INS}} &= v^{t-1}_f + a^t_f \Delta t \\ v^t_{l_{INS}} &= v^{t-1}_l + a^t_l \Delta t \end{aligned} \tag{4}$$

In order to mitigate the adverse effect of the accumulated error of two integral processes of acceleration, we add a feedback to adjust the velocity calculated using INS, which is formulated as

Fig. 7. The localization trajectory when using INS as odometry. From left to right are KITTI-City, KITTI-Residential, KITTI-Road, Our-Dataset. The yellow star in each figure indicates the starting point of the sequence.

TABLE I
AVERAGE POSITION ERROR AND HEADING ERROR WHEN USING INS

Category	Method	Position Error (meter)		Heading Error (degree)	
		Average Error	Standard Deviation	Average Error	Standard Deviation
KITTI-City	INS	287	247	**0.17**	**0.12**
	Canny [17]	237	185	1.32	1.28
	Ours	**1.82**	**1.01**	0.59	0.27
KITTI-Residential	INS	1015	847	5.04	4.16
	Canny [17]	1205	968	4.69	2.94
	Ours	**3.43**	**1.72**	**2.23**	**1.37**
KITTI-Road	INS	99.9	90.0	2.49	3.40
	Canny [17]	213	168	3.43	1.22
	Ours	**2.88**	**3.00**	**0.96**	**1.26**
Our-Dataset	INS	153	109	3.98	2.58
	Canny [17]	128	79.9	4.77	2.31
	Ours	**3.27**	**2.70**	**0.75**	**1.21**

$$\begin{aligned} v_f^t &= \alpha_1 v_{f_{INS}}^t + (1-\alpha_1)(\tilde{v}_e \cos\psi_{t-1}^i + \tilde{v}_n \sin\psi_{t-1}^i) \\ v_l^t &= \alpha_2 v_{l_{INS}}^t + (1-\alpha_2)(-\tilde{v}_e \sin\psi_{t-1}^i + \tilde{v}_n \cos\psi_{t-1}^i) \end{aligned} \tag{5}$$

where $\tilde{v}_e = (x_{t-1}^i - x_{t-n-1}^i)/(n\Delta t), \tilde{v}_n = (y_{t-1}^i - y_{t-n-1}^i)/(n\Delta t)$ is the average speed estimated by history positions, $\alpha_1 = 0.7, \alpha_2 = 0.4, n = 10$ in the experiment. This is performed only when the max weight of the particle is higher than 0.85, and have an at least one-second interval. The standard deviation of Gaussian noise in the particle filter is 3.0 m in the longitudinal direction, 1.0 m in the lateral direction, and 10^{-7} rad in yaw angle. In addition, the INS used in our dataset has high accuracy. Therefore, in order to simulate the performance of lower accuracy INS, we add Gaussian noise with -0.0001m/s^2 mean and 0.0012 m/s^2 standard deviation to the leftward acceleration, 0.0007 m/s^2 mean and 0.004 m/s^2 standard deviation to the forward acceleration, 0.00004 rad/s mean and 0.00025 rad/s standard deviation to the angular velocity. The second odometry method is LiDAR Odometry and Mapping method [27], with a formulation similar to INS. We take a similar approach to mitigate the effect of possible abruptly increased error of odometry, where the velocity is calculated by $v_{f_{LO}}^t = \Delta x_f/\Delta t, v_{l_{LO}}^t = \Delta x_l/\Delta t$. The standard deviation in the particle filter is 0.4 m in the longitudinal direction, 0.2 m in the lateral direction, and 10^{-7} rad in yaw angle.

Note that the model of INS is very coarse since the biases of gyro and accelerometer are not estimated. Ideally, it is better to estimate the bias because if the particle filter's odometry input could be the result which is corrected by the estimated bias, the prior distribution of the particle filter will be more accurate and the variance will be smaller. But the main purpose of constructing the INS experiment is to explore the performance of our neural model under the circumstance with relatively higher odometry input noise (which will cause a larger deviation between the predicted prior and actual distributions). If our method could localize under this condition, the weights should be calculated accurate enough during the weight update step, which indicates our neural network should have the ability to identify the pose where satellite image correspond with grid-map, as well as the ability to distinguish it when there is a spatial difference between the two maps. Therefore, the bias in INS is not a major concern in this work.

During the initialization phase, the particles of the particle filter are sampled from the uniform distribution within a radius of 5 meters on the position of the starting point, and within 5 degrees around the ground-truth heading angle. Note that

Fig. 8. The localization trajectory when using LiDAR Odometry. From left to right are KITTI-City, KITTI-Residential, KITTI-Road, Our-Dataset. The yellow star in each figure indicates the starting point of the sequence.

TABLE II
AVERAGE POSITION ERROR AND HEADING ERROR WHEN USING LIDAR ODOMETRY

Category	Method	Position Error (meter)		Heading Error (degree)	
		Average Error	Standard Deviation	Average Error	Standard Deviation
KITTI-City	LeGO-LOAM [27]	2.13	1.31	0.47	0.35
	Canny [17]	2.33	1.25	**0.23**	**0.07**
	NMI [20]	4.71	2.55	0.72	0.08
	Ours	**1.45**	**0.77**	0.27	0.16
KITTI-Residential	LeGO-LOAM [27]	20.05	14.36	2.38	1.52
	LeGO-LOAM(CL) [27]	8.00	2.70	1.01	0.99
	Canny [17]	9.95	3.32	2.02	0.12
	NMI [20]	20.23	14.19	5.03	4.01
	Ours	**2.36**	**1.27**	**0.79**	**0.36**
KITTI-Road	LeGO-LOAM [27]	237	172	2.70	3.92
	Canny [17]	77.01	41.43	4.33	3.59
	NMI [20]	22.95	16.96	2.33	3.06
	Ours	**3.90**	**3.78**	**2.39**	**1.23**
Our-Dataset	LeGO-LOAM [27]	4.89	2.61	3.18	5.93
	Canny [17]	3.31	1.52	1.96	0.06
	NMI [20]	4.54	2.23	**0.76**	**0.15**
	Ours	**2.99**	**1.33**	1.05	0.29

the NMI method [20] needs to build a short-term history re-emission map, therefore we only compare this method when using LiDAR Odometry and Mapping, because the result of INS has a very large error (since the output of the position is obtained by two integral processes through the result of the acceleration measurement).

V. RESULTS AND DISCUSSION

Fig. 6 shows the ROC curve of our neural model and the baseline methods. It can be found that our model outperforms the baselines. This means our neural model generalizes better when compared with baselines. When comparing the data augmentation methods, the performance of training without mirroring and rotating is the worst, and the performance improvement of training with mirroring is smaller than that of training with rotating.

Fig. 7 and Fig. 8 show the localization trajectory on these sequences when using two different odometry methods. Table I and Table II shows the position and heading error. Note that in Table II, loop closures are detected by LeGO-LOAM in KITTI-Residential and that result is shown as LeGO-LOAM(CL). Our method has smaller average errors since the baseline methods cannot stably determine whether the grid-map and the satellite image patch correspond to the same position, the cumulative error of the odometry cannot be corrected in some scenarios. Besides, the test result on our dataset proves that our model has generalization capability in a different environment. The average errors when using LiDAR Odometry are basically smaller than using INS, except the KITTI-Road sequence where the error of our method slightly increases, since in some scenes there are less geometric features and the LiDAR Odometry has larger error than INS.

Fig. 9 shows the scenes with relatively low positioning error using our method. It can be seen from the grid-map and the satellite image that these scenes have obvious structural features, such as the edge of the building, the trees, the intersection, or the edge of the curved road. In addition, in some challenging scenarios such as the area shadowed by

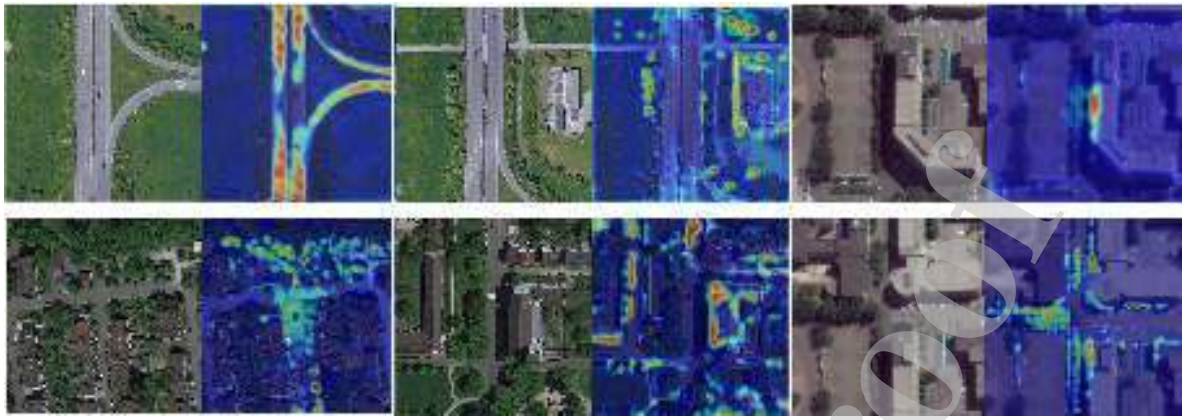


Fig. 11. We use Grad-CAM [29] to generate heat maps in order to understand which areas of image mostly influence the network's judgment. It can be seen that the elements affecting the judgment of the neural network are different in different scenarios.

Fig. 9. Examples of the scenes that our method has low positioning error. Each row in the figure is the satellite image (left), the height grid-map (center), and the intensity grid-map (right) of different scenes. The red dot in the satellite image is the ground truth position, the green dot is the estimated position. The blue and black dots are the particles spread by the particle filter. Deeper color of a particle corresponds to a lower weight.

building (The first row in Fig. 9) or occluded by trees (The fourth row in Fig. 9), our method still performs well. This may partly because of the neural network's ability to learn appropriate feature representations for a certain problem which gives it a competitive edge over traditional methods [23]. The deep features in the fully convolutional neural network are encoded with location and semantics in a nonlinear local-to-

Fig. 10. Examples of buildings in satellite images. Note that the view of the satellite image has varying degrees of oblique.

global pyramid [28]. Deep features are more stable and should be helpful to guide the judgment of matching. Without deep features, traditional methods based on basic features (such as edge or intensity) consider all the elements indiscriminately. In contrast, according to the heat maps generated by Grad-CAM [29] (See Fig. 11), in many cases, the contours of buildings or trees affect the judgment of the neural network more than the edges of occluders or shadows in scenes. This can explain, to some extent, why the neural network is less affected by shadows and occlusions. Meanwhile, since the training data contains shadow and occlusion scene, neural networks can learn strong prior information about this kind of scenario [30]. If the network is trained to learn it perfectly, then when it faces similar situations when testing, it is likely to judge accurately.

Although our neural model learns to match with the grid-map according to the structural features in the satellite image, the matching error still exists. In other words, our model cannot distinguish slight deviations between the two maps. There are several reasons that might be responsible. First, the satellite images' angle of view is slightly oblique instead of vertically downward. As a result, the side of the building can be seen in the satellite image (see Fig. 10). Therefore, the exact position where LiDAR scanline hit the building is hard to find on the satellite image. Second, if the roof of a building is larger than the bottom of the building, accurate building edges cannot be obtained in the satellite image. Third, although the trees in the satellite image help with matching, they do not correspond exactly to the grid-map, because the

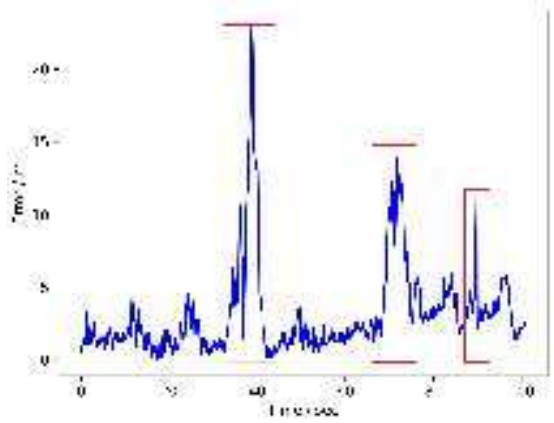


Fig. 12. The position error curve of our method on KITTI-Road when using INS as odometry. The error in the red boxes increases sharply.

shape formed by LiDAR scanline hitting the treetop or trunk is usually different from the shape of the tree in the satellite image. At last, the tilting of the vehicle caused by the terrain undulation will deform the grid-map, resulting in an imperfect match between the far-away LiDAR scan and the satellite image. Nevertheless, we think there are still some measures that can alleviate the detrimental effect of these issues. If more scenes and more satellite images of different times is included in the training data, there will be sufficient samples containing various styles of roofs and trees, and different view angles, from which our model can learn from the more diversified and increased amount of data to infer possible shapes of the bottom of buildings and trunk of trees. Meanwhile, the neural network might find out the law that which elements are more accurate in determining the correspondence of grid-map and satellite image when considering these mentioned issues (for example, maybe it can learn a confidence priority ordering of available elements).

In addition, our method has a large positioning error on the straight road that lacks structural features helpful for matching. We use KITTI-Road sequence as an example to illustrate this. The position error curve when using INS is shown in Fig. 12. It can be found that the position error increases sharply at several moments (the red boxes in Fig. 12). The reason for the increase in error at these times is the same, therefore we take the scene corresponding to about 40 seconds as an example to illustrate the reason (see Fig. 13). At first, the positioning error of our method is low, because there is a branch road near the location of the vehicle. This branch road provides a reference for our model to match the two maps. At this time, the scattering range of the particles in the figure is small, which means the variance of the position distribution is small. As the vehicle travels, it moves away from the branch road and enters the straight road where the LiDAR points are mostly within the road due to the occlusion of the road edge. Some similar scenes are shown in Fig. 14. In these scenarios, the correspondence between the satellite image and grid-map can be clearly found in the direction perpendicular to the road, while few structural features can be used as cues in the direction parallel to the road. Therefore, the variance

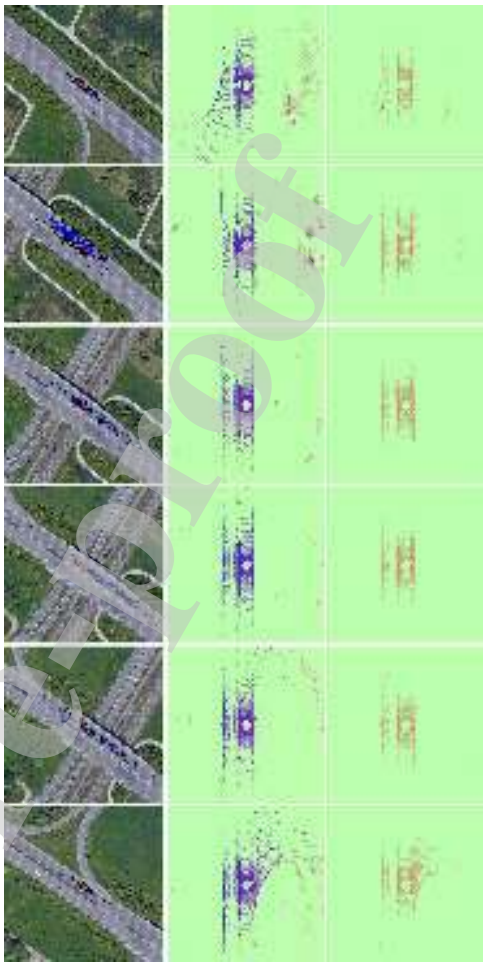


Fig. 13. The scenes correspond to the KITTI-Road sequence at around 40 sec (one of the red boxes in Fig. 12). The position error reaches the maximum on the fourth row.

of the position distribution in the direction along the road is large. In the end, when the vehicle leaves this area, useful structural features reappear, the positioning error decreases, and the variance of the particles becomes smaller.

In addition to the above factors, the resolution of the satellite image also affects the accuracy of positioning. However, a higher image resolution could improve the positioning accuracy only with specific conditions. Suppose the satellite images' field of view in real meters remains unchanged and the resolution is increased, the size of the neural network needs to expand, which increases the number of parameters in the neural network. The depth of the neural network would also need to increase for maintaining the receptive field of the high-level layers, otherwise, the ability to extract the semantic information will diminish. Therefore, only with enough training data to train the larger network, the positioning accuracy of our method might be improved, because the edges of buildings, trees and roads are clearer, making it easier to distinguish a smaller spatial difference between the satellite image and grid-map. In addition, one may consider decreasing the satellite images' field of view to maintain the size of the neural network. However, this approach does more harm than

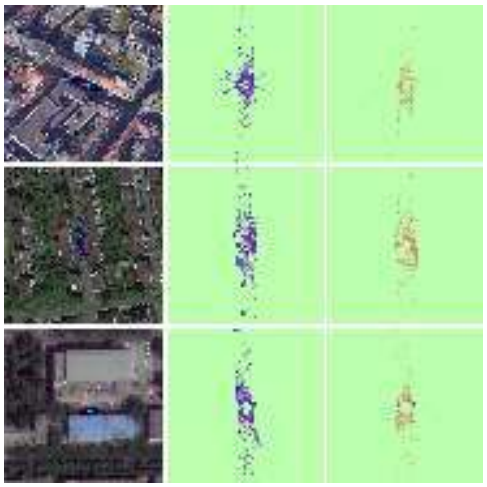


Fig. 14. Examples of the scenes that our method has high positioning error.

good because it lowers the probability of using some critical structural feature that locates far away from the vehicle, which harms the robustness of our method. Therefore, to increase the resolution of satellite image, there are some adverse effects attributed to increased parameters, such as more required training data, difficulty of training, and lower running speed, which might be alleviated by modifying the architecture of our neural network using new techniques in deep learning like depth-wise separable convolutions [31], ENAS[32], etc.

The localization accuracy of our method might not meet the requirements of stable lane-level localization, but our method has the ability to perform road-level localization in the GNSS-denied area or urban canyon. With the help of satellite images, our method is able to serve as a low-cost localization method without the requirement of pre-built maps. Furthermore, the characteristics of our method ensure high accuracy in intersections, which are critical places for the navigation of autonomous vehicles. While on a straight road where our method has a higher error, the road-level localization might be enough to navigate to the next road node.

In this paper, we only match the single-frame LiDAR scan with the satellite image. The accuracy of our method could be improved by accumulating multi-frame LiDAR scans to build a short-term grid-map before feeding to the network, but this requires sufficiently high accuracy of the odometry method. As for computational efficiency, our method is currently not up to real-time, and the time used on comparing a pair of satellite image and grid-map is about 86 ms using PyTorch on a Nvidia 1080Ti. In our experiments, the feature map extraction of the satellite image patch is done every time the network predicts the matching degree. To improve the computational efficiency, the feature maps of the entire satellite image with multiple rotation angles can be extracted offline with the satellite image branch, and the matching operation can be performed at intervals according to the accuracy of the odometry method.

VI. Conclusion

We propose a learning-based method to localize the vehicle on a geo-referenced satellite image. Our method takes a single scan of LiDAR as input, and outputs the position and heading of the vehicle. In this method, a neural network is designed to learn to match the LiDAR points with their corresponding satellite image patch. This network outputs the probability of correspondence which serves as observations in a particle filter that estimates a distribution of the vehicle's pose. We also propose a way to train this neural model for high performance. We evaluate our method against baseline methods on several datasets and experimental result shows that the learned model generalizes better to different environments and our method can achieve more stable localization even in some challenging scenarios.

In this paper, we only use single-frame LiDAR points because we do not require the accuracy of the odometry method to enable multi-frame point cloud accumulation. In our future work, we will apply our neural network on multi-frame points generated by LiDAR odometry and perform matching operation only when there are full of structural features (such as in road intersections) in order to achieve accurate real-time localization. Besides, the possible deviation of actual road structures from the history satellite image creates a potential problem. Theoretically, our method may suffer from mismatching in this scenario, because in the training data the satellite image is in accordance with the actual road structures. In the future, we will construct a dataset with samples that road structures are different from the history satellite images to figure out whether our method can handle this kind of challenge. Also, we will also explore its performance improvement with these kinds of training data.

ACKNOWLEDGEMENTS

This work was supported by the National Natural Science Foundation of China (NSFC, 61973034, U1913203, 61473042 and 61903034).

References

[1] K. Novak, "Mobile mapping systems: new tools for the fast collection of gis information," in *State-of-the-Art Mapping*, vol. 1943, pp. 188–198, International Society for Optics and Photonics, 1993.

[2] I. Puente, H. González-Jorge, J. Martínez-Sánchez, and P. Arias, "Review of mobile mapping and surveying technologies," *Measurement*, vol. 46, no. 7, pp. 2127–2145, 2013.

[3] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: part i," *IEEE robotics & automation magazine*, vol. 13, no. 2, pp. 99–110, 2006.

[4] S. Lefèvre, D. Tuia, *et al.*, "Toward seamless multiview scene analysis from satellite to street level," *Proceedings of the IEEE*, vol. 105, no. 10, pp. 1884–1899, 2017.

[5] S. Thrun, D. Fox, W. Burgard, and F. Dellaert, "Robust Monte Carlo localization for mobile robots," *Artificial intelligence*, vol. 128, no. 1-2, pp. 99–141, 2001.

[6] M. Noda, T. Takahashi, D. Deguchi, I. Ide, H. Murase, *et al.*, "Vehicle ego-localization by matching in-vehicle camera images to an aerial image," in *Asian Conference on Computer Vision*, pp. 163–173, Springer, 2010.

[7] A. Viswanathan, B. R. Pires, and D. Huber, "Vision based robot localization by ground to satellite matching in gps-denied situations," in *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pp. 192–198, IEEE, 2014.

[8] N. N. Vo and J. Hays, "Localizing and orienting street views using overhead imagery," in *European Conference on Computer Vision*, pp. 494–509, Springer, 2016.

[9] D.-K. Kim and M. R. Walter, "Satellite image-based localization via learned embeddings," in *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pp. 2073–2080, IEEE, 2017.

[10] S. Hu, M. Feng, R. M. Nguyen, and G. Hee Lee, "Cvm-net: Cross-view matching network for image-based ground-to-aerial geo-localization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7258–7267, 2018.

[11] X. Wang, S. Vozar, and E. Olson, "Flag: Feature-based localization between air and ground," in *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pp. 3178–3184, IEEE, 2017.

[12] T. Senlet and A. Elgammal, "A framework for global vehicle localization using stereo images and satellite and road maps," in *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pp. 2034–2041, IEEE, 2011.

[13] T. Senlet and A. Elgammal, "Satellite image based precise robot localization on sidewalks," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pp. 2647–2653, IEEE, 2012.

[14] A. Viswanathan, B. R. Pires, and D. Huber, "Vision-based robot localization across seasons and in remote locations," in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pp. 4815–4821, IEEE, 2016.

[15] A. Gawel, C. Del Don, R. Siegwart, J. Nieto, and C. Cadena, "X-view: Graph-based semantic multi-view localization," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1687–1694, 2018.

[16] G. Máttyus, S. Wang, S. Fidler, and R. Urtasun, "Hd maps: Fine-grained road segmentation by parsing ground and aerial images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3611–3619, 2016.

[17] R. Kümmerle, B. Steder, C. Dornhege, A. Kleiner, G. Grisetti, *et al.*, "Large scale graph-based slam using aerial images as prior information," *Autonomous Robots*, vol. 30, no. 1, pp. 25–39, 2011.

[18] P. Ruchti, B. Steder, M. Ruhnke, and W. Burgard, "Localization on openstreetmap data using a 3d laser scanner," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pp. 5260–5265, IEEE, 2015.

[19] O. Vysotska and C. Stachniss, "Exploiting building information from publicly available maps in graph-based slam," in *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, pp. 4511–4516, IEEE, 2016.

[20] L. de Paula Veronese, E. de Aguiar, R. C. Nascimento, J. Guivant, F. A. A. Cheein, *et al.*, "Re-emission and satellite aerial maps applied to vehicle localization on urban environments," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pp. 4285–4290, IEEE, 2015.

[21] M. Javanmardi, E. Javanmardi, Y. Gu, and S. Kamijo, "Towards high-definition 3d urban mapping: Road feature-based registration of mobile mapping systems and aerial imagery," *Remote Sensing*, vol. 9, no. 10, p. 975, 2017.

[22] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241, Springer, 2015.

[23] A. Garcia-Garcia, S. Orts-Escolano, S. Oprea, V. Villena-Martinez, and J. Garcia-Rodriguez, "A review on deep learning techniques applied to semantic segmentation," *arXiv preprint arXiv:1704.06857*, 2017.

[24] S. Zagoruyko and N. Komodakis, "Learning to compare image patches via convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4353–4361, 2015.

[25] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[26] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[27] T. Shan and B. Englot, "Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4758–4765, IEEE, 2018.

[28] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3431–3440, 2015.

[29] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *Proceedings of the IEEE international conference on computer vision*, pp. 618–626, 2017.

[30] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[31] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1251–1258, 2017.

[32] H. Pham, M. Y. Guan, B. Zoph, Q. V. Le, and J. Dean, "Efficient neural architecture search via parameter sharing," *arXiv preprint arXiv:1802.03268*, 2018.

Mengyin Fu

received the B.S. degree from Liaoning University, China, the M.S. degree from the Beijing Institute of Technology, China, and the Ph.D. degree from the Chinese Academy of Sciences.

He was elected as the Yangtze River Scholar Distinguished Professor in 2009. He was a recipient of the Guanghua Engineering Science and Technology Award for Youth Award in 2010 and the National Science and Technology Progress Award for several times in recent years.

Prof. Fu is the President of the Nanjing University of Science and Technology. His research interest covers integrated navigation, intelligent navigation, image processing, learning, and recognition and their applications.

Minzhao Zhu

received the B.S. degree from Beijing Institute of Technology, Beijing, China. He is currently pursuing the M.S. degree with the State Key Laboratory of Intelligent Control and Decision of Complex Systems, and also the School of Automation, Beijing Institute of Technology.

His research interests include autonomous vehicle, computer vision, visual SLAM, and air-ground collaborative perception.

Yi Yang

received the Ph.D. degree in control science and engineering with the School of Automation, Beijing Institute of Technology, Beijing, China.

He is currently a Professor with the School of Automation, Beijing Institute of Technology. He is author/co-author of more than 40 conference and journal papers in the area of unmanned ground vehicle. His research interests are environment perception of unmanned ground vehicle, motion planning and control, and robot design and development.

Wenjie Song

received the Ph.D. degree in control science and engineering with the School of Automation, Beijing Institute of Technology, Beijing, China. He is currently an Assistant Professor with the School of Automation, Beijing Institute of Technology.

His research interests include autonomous vehicle, computer vision, visual SLAM, path planning, and motion controlling. He has taken part in China Intelligent Vehicle Future Challenge several times as the Captain or Core Member.

Meiling Wang

received the B.S. degree in automation from the Beijing Institute of Technology, China, in 1992, and the M.S. and Ph.D. degrees from the Beijing Institute of Technology, China, in 1995 and 2007, respectively. She has been teaching at the Beijing Institute of Technology since 1995, and worked at the University of California San Diegoas a Visiting Scholar in 2004.

She was elected as the Yangtze River Scholar Distinguished Professor in 2014. She is currently the Director of the Integrated Navigation and Intelligent Navigation Laboratory, Beijing Institute of Technology, China. Her research interests include advanced technology of sensing and detecting and vehicle intelligent navigation.

Mengyin Fu

Minzhao Zhu

Yi Yang

Wenjie Song

Meiling Wang

Declaration of interests

■The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☐The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: