Image-based Localization for Self-driving Vehicles Based on Online Network Adjustment in A Dynamic Scope

Guoyu Lu
Intelligent Vision and Sensing Lab
Rochester Institute of Technology
University of Georgia

2022 International Joint Conference on Neural Networks (IJCNN) | 978-1-7281-8671-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/IJCNN55064.2022.9892072

***Abstract*—Image-based localization provides an alternative solution for camera pose estimation, which is a crucial component for self-driving vehicles. Localization for vehicles requires continuous feedback. We propose a solution that can accurately estimate the vehicle position and orientation. In this solution, we provide a complete pipeline for self-driving vehicles, including map building and camera pose estimation. We first design a convolutional neural network and train the localization system based on the entire global map. During the real-time localization stage, we fine-tune the network regressor online through the training images in adjacent locations in the map, which can enhance the localization accuracy significantly. Depending on the vehicle motion, we adjust the scope of local training images dynamically. We demonstrate the superior performance of our method through experiments on benchmark dataset.**

I. Introduction

The vision-based localization algorithm is to match the query image against the map to estimate the camera pose. We utilize SLAM [26] for map generation because of its accuracy, and more importantly, the ability to generate a 3D map in real-time. Unfortunately, SLAM solution may involve accumulation error in the region that is far from the loop closure point. To further improve the map accuracy, we can build the map based on IMU and GPS data. A standard approach to determine the location of a locally captured image in the global map is to match their features and search for a scene that yields maximal matches. A nicely structured map allows us to utilize extra information to reduce the search space and thus improves the efficiency of searching for correct matches. The objective of the proposed algorithm is to localize a road vehicle on a map. Under the road driving condition, the images and corresponding camera poses are stored in a tree structure, where the location of the corresponding branch is the image location in the map given by the SLAM and GPS solution, which increases the localization accuracy by focusing on a local region.

We train a convolutional neural network based on images and their camera poses. The camera pose is obtained from the map building process. The trained network is a regression framework with 6 degrees of freedom. As many SLAM algorithms suffer from accumulation error issues, the map and camera pose labels may be inaccurate. To make our algorithm robust and widely applicable, our network can be trained based on maps built by any SLAM/SfM methods. To achieve this objective, we not only apply the camera pose obtained from map generation process to learn the model but also enforce geometric and photometric consistency during the training process, which can help mitigate the inaccurate camera pose label problems. We calculate the camera motion based on the difference between camera poses associated with images in a selected local keyframe set. Based on the camera motion, we transform the image from one position to another and examine the intensity difference between the overlayed images. Meanwhile, as we have obtained the 3D point cloud as a map, we back project the 3D points to the keyframes and reduce the distance between back-projected points. In such a case, we can rely on both spatial and temporal information to learn the localization system. During the localization process, the vehicle's camera captures an image as the query. Our localization procedure first utilizes the last known vehicle camera pose as the initial guess to select a subset of image frames with their camera poses. The selected frames are applied to fine-tune the network online. Once the network is updated, it will output the camera pose of the query image using the trained regressor. We assume that the vehicle's last known location is close to its current location. The significant error in determining the vehicle's previous location could lead to diverging result at the present time step. To overcome this issue, when the last frame's motion is mainly rotation, we enlarge the search scope to increase the chance of accurately fine-tune the network. The range of the search scope is set back to the original configuration when the motion is mainly translation. The strategy of dynamic search scope control ensures the robustness of the algorithm to remove the error in prior guess. The entire localization framework is shown in Fig. 1.

Our contributions are summarized as follows: 1) we build a complete solution targeting at self-driving vehicle localization, including map building and vehicle pose estimation; 2) we apply a convolutional neural network to the vehicle localization problem; 3) we propose a strategy to dynamically control the

978-1-7281-8671-9/22/$31.00 ©2022 IEEE

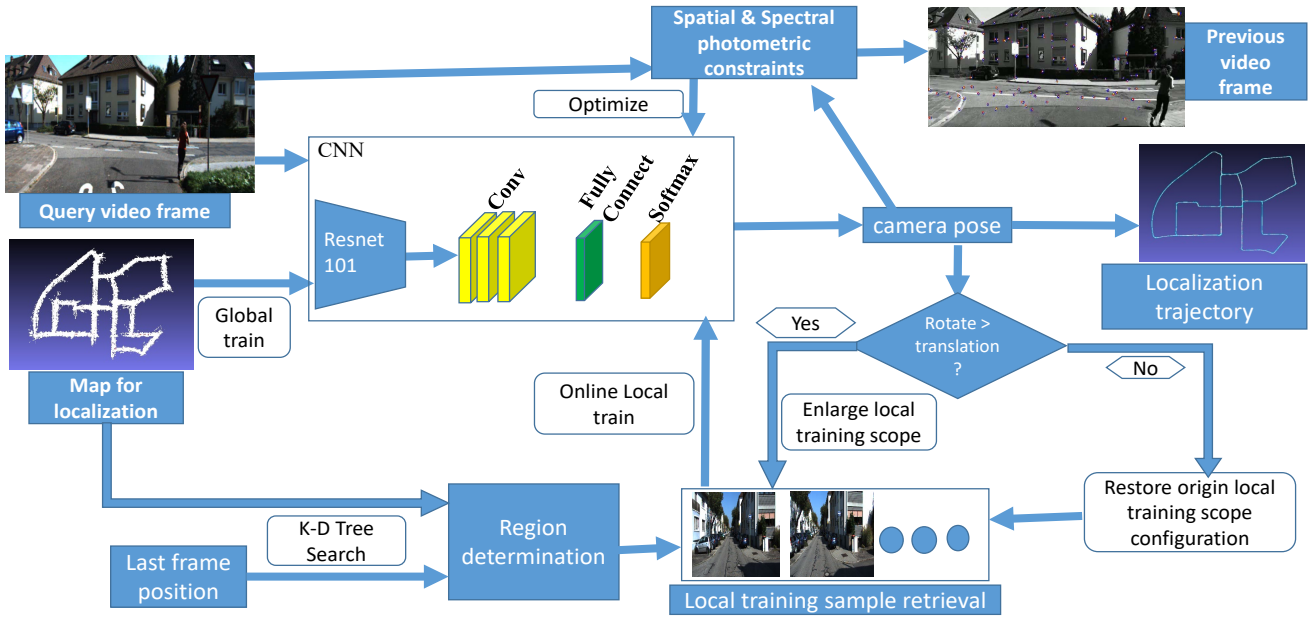


Fig. 1: The vehicle localization framework. A highly accurate map is generated for offline localization neural network training. Both spatial and spectral photometric clues are applied to constrain the neural network learning. During online localization, the system will be refined based on the adjacent images. The images used for online network optimization will be dynamically adjusted based on the vehicle motion.

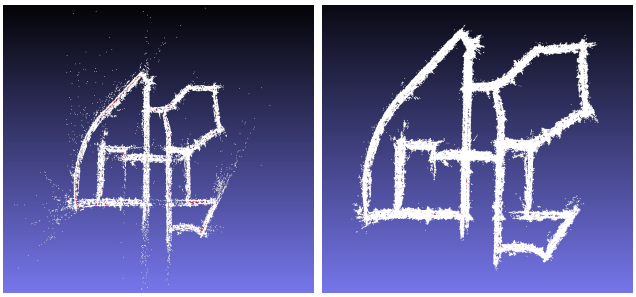
scope based on camera pose indexing to fine-tune the neural network online.

II. Related Work

Visual localization estimates the camera pose, including camera position and orientation, when given a query image as the input. The initial map for localization is a database of 2D images, each of which is associated with the position information. Correspondingly, image retrieval was initially applied to localize the query image [28]. In this process, vocabulary structure [31], hierarchical search [37], and holistic histogram features [36][15] are applied to enhance the image retrieval speed.

Based on the development of 3D modeling algorithms [32] [6], 3D point clouds are also used to localize the query image. By registering the query scene to the 3D reconstruction point cloud, the system can estimate both the camera position and orientation [14][18]. 2D images features are also directly matched to the 3D point cloud using Visual words to accelerate the matching process among the entire 3D point cloud [29][19][20]. Randomized tree [12], random forest [1] and embedded random ferns [9] were also used to obtain the correspondences between image and map. 3D-to-3D matching [21] and multitask learning frameworks [22] were applied to improve the matching accuracy in a fast speed. Cvivsic et al. [7] utilize stereo vision to track images features and build a pose graph to optimize the localization result. With the development of deep learning, CNN network was applied to localize the query image [16]. The training labels were from the camera pose estimation of structure-from-motion (SfM). The camera pose was estimated from the train regressor without specific feature matching. Expert Sample Consensus [2] explores deep neural network on RANSAC to enhance the feature matching accuracy and applied improved feature matching on camera localization problem. KFNet [41] incorporates Kalman filter in the localization process to regress the scene coordinate. Inloc [33] densely match the correspondences and synthesize views to build constraints for indoor localization. Recurrent neural network is also applied in global pose estimation [27]. However, Sattler et al. [30] pointed out that CNN-based camera pose regression do not consistently outperform hand-crafted image retrieval methods. When the map size is increased, the classification accuracy is also reduced due to the confusion of appearance. As the scope for autonomous driving is usually city-scale or even larger, the localization accuracy decreases with the increase of the map. To deal with this problem, we apply an online refinement based on our pre-trained neural network localization model.

To generate 3D maps in real-time, visual odometry (VO) [40] and Simultaneous Localization and Mapping (SLAM) algorithms [5] estimate the camera pose and 3D point cloud. Though with the help of local bundle adjustment and loop closure detection, the drift error can be reduced, the accumulation error is always the most significant issue for SLAM and VO algorithms. The deep neural network is also applied in VO [24][35][38]. However, existing deep neural network-based methods mainly focus on the adjacent frames' depth and camera pose estimation and have not realized the loop closure detection issue, leading to the entire trajectory largely differ from the ground truth. Different from the 3D point cloud, our map models are built through 3D camera poses, which will be introduced in section III. To localize the query image, we apply a convolutional neural network-based method, which is presented in section IV with basic



(a) (b)

Fig. 2: 3D map built by SLAM and adjusted based on IMU and GPS. (a) Original ORB-SLAM Map. (b) Map adjusted by high precision IMU and GPS. White points are the 3D points shown in the 2D map. Red lines are the ground truth.

network structure and training method in section IV.A. Relying on the vehicle location prior, we train neural network locally online introduced in section IV.B and dynamically change of the adjusting scope in section IV.C. The loss function to train the neural network is described in section IV.D.

III. Map Building

An accurate map is critical to provide a precise reference for the images to be localized, which can support to estimate the vehicle position and orientation accurately. As the road condition keeps changing, a fast and accurate map generation method is required. To realize this objective, we use IMU, or GPS or SLAM [26][25] as our map building method, as those methods can be used in real-time with satisfactory accuracy.

Most autonomous driving (e.g., Google, Baidu, Ford) and mapping service providers (e.g., HERE maps, TomTom) equip their vehicles with high precision GPS and IMU, especially for the map building purpose. Therefore, GPS and IMU or SLAM are used to build the map. As maps are provided by map companies, autonomous consumer cars are not required to be equipped with high precision GPS and IMU. Alternatively, we can also use the SLAM method to build the map, which relies on loop closure detection to decrease the drift error. We make the map based on each image's camera pose. To show the built map, we display the point cloud built by SLAM and IMU/GPS (Fig. 2), which is another output of the SLAM together with camera pose. The desired output for localization is camera pose. However, as we also apply 3D points to train the network, we also maintain the 3D map points together with camera poses.

Once each image obtains its camera pose, we index images into a K-D tree based on their camera poses. Our map building method utilizes a clustering structure that groups the map points in a specific window based on their locations. During the network training stage, we apply the back-project error as the loss function. Our framework of visual localization defines that all map points from the same image belong to a cluster. These clusters are indexed in a KD-tree based on their camera poses.

IV. Image-based Localization

A. Overall Localization System

Once we have created an accurate map, we can utilize this map to train a convolutional neural network to localize the query image. To train this network, we first rely on Resnet101 network as the base of our model and apply transfer learning to compensate for the training samples to learn our model. To better support our task, we modify the Resnet101 network based on the following steps: (1) remove the last 3 fully connected layers of Resnet101; (2) add a convolutional layer; (3) add another pooling layer; (4) add an additional fully connected layer; (5) normalize the output through softmax.

We tune the last four layers based on our entire mapping data. The training labels are the 6 degrees of freedom obtained from GPS and IMU or SLAM method, as well as the point cloud data. The translation matrix is a 3-dimensional vector. The rotation matrix is transformed into quaternion represented a 4-dimensional vector. Then the label for each image is a 7-dimensional vector. The trained network is the localization model that we will utilize to localize the image and obtain the camera pose.

B. Online Local Fine Training Network

As the network is trained based on the entire map, the network is targeted to provide satisfactory performance for the entire map region. Once we know the rough vehicle location, we can select a region of training samples to fine-tune the network, which may provide more accurate camera pose estimation for the current query image. As we choose just a tiny number of images to tune the newly added layers, this local training process can be conducted in the online process. As our proposed algorithm utilizes a clustering structure that groups the map points in a specific window based on their location, we can quickly extract the images close to the current query frame. 3D map generated from the vision-based algorithm can be conveniently clustered based on images where the features are extracted in SLAM. Our framework of visual localization defines that images are indexed in a KD-tree based on their camera poses.

During localization, we define the fine training scope as the number of images that are used to fine train the neural network. The scope defining the fine-tune region is approximated by the vehicle last known camera pose. Assuming the update interval of image measurement is sufficient, a moving vehicle operating under the normal driving condition cannot have a dramatic change in its ego-motion between two measurement images. Thus, we can also utilize the localization result in the last measurement to determine the network fine-tuning scope of the current frame. We can propagate the search scope location with vehicle kinematic model to further decrease the distance between vehicle location and search scope location and provide faster and more accurate scope prediction between the query image and the map. When there is no prior knowledge of the vehicle available, we apply the trained network based on the entire map without local training for the initial location. Once

the starting location is identified, we determine the search area based on the current location.

C. Dynamic Scope Control

The size of the online adjusting scope is dynamically updated based on estimation accuracy. When the previous result is accurate, a smaller search scope could be utilized, and vise versa. This paper utilizes an initial search scope of 5 images that cover the front, back, left and right direction of current image location. When the vehicle motion is determined, i.e., the moving direction of the vehicle is known, we can further narrow down the search scope. Experimentally, while the vehicle is moving forward or backward, which means the motion is mainly translation, there is a relatively substantial overlap between adjacent query frames. When the scene alters dramatically, such as significant rotation motion, we enlarge the search scope. It is practically difficult to determine the accuracy of the localization without a reference signal. Directly utilizing residual error for accuracy measurement can sometimes be biased. In this paper, we propose to utilize the ratio of the magnitude of translation and rotation vectors as a criterion to predict our estimation accuracy. When the ratio between translation and rotation magnitude is below a predefined threshold, we increase the size of the training scope. When the ratio increases to a limit, we can reduce the size of the search scope back to the default configuration. The threshold in our case is set to 2.

D. Loss Constraint

The result from the matching step gives a set of map points and their 2D projection coordinates in the current measurement image plane. The coordinate of a 3D point in camera body frame, $\mathbf{p}_c \in \mathbb{R}^3$, can be calculated from its known coordinate in the map frame, $\mathbf{p} \in \mathbb{R}^3$, by Eq. 1:

$$\mathbf{p}_c = R_w \mathbf{p} + \mathbf{t}_w \tag{1}$$

The projection coordinate of $\mathbf{p}_w$ in image plane is calculated with the perspective projection equation:

$$\begin{aligned} \frac{u}{f} &= \frac{R_{11}p_x + R_{12}p_y + R_{13}p_z + t_x}{R_{31}p_x + R_{32}p_y + R_{33}p_z + t_z} \\ \frac{v}{f} &= \frac{R_{21}p_x + R_{22}p_y + R_{23}p_z + t_y}{R_{31}p_x + R_{32}p_y + R_{33}p_z + t_z} \end{aligned} \tag{2}$$

We extract image feature coordinate $[u, v]$ from the query image and acquire the corresponding map points $\mathbf{p}_w$ from the map, which is available in SLAM map building process. From this given information, our objective is to estimate R_w and $\mathbf{t}_w$. We achieve this by solving an optimization problem that minimizes the following loss function:

$$\begin{aligned} \min_{R_w, \mathbf{t}_w} L = &\left(\begin{bmatrix} u \\ v \end{bmatrix} - Proj(R_w, \mathbf{t}_w, \mathbf{p}) \right)^T \\ &\left(\begin{bmatrix} u \\ v \end{bmatrix} - Proj(R_w, \mathbf{t}_w, \mathbf{p}) \right) \end{aligned} \tag{3}$$

where function $Proj$ is the perspective projection equation at the right side of Eq. 2 caused by the perspective camera model. Eq. 3 is the least square cost function.Our work parameterizes the rotation matrix with the quaternion. The quaternion is a 4×1 unit vector, $\mathbf{q} = [q_1, q_2, q_3, q4]$. It yields less number of parameters compared with the rotation matrix but does not experience the singularity issue like Euler angle and Classical Rodrigues parameters. Each element of a rotation matrix parametrized with the quaternion is a multivariate quadratic function of $\mathbf{q}$. Although the element of the rotation matrix is non-linear, we can show that Eq. 2 is linear in terms of rotation matrix elements after manipulated into form of Eq. 4.

$$\begin{aligned} &(uR_{31} - fR_{11})p_x + (uR_{32} - fR_{12})p_y \\ &+ (uR_{33} - fR_{12})p_z = ft_x - ut_z \\ &(vR_{31} - fR_{21})p_x + (vR_{32} - fR_{22})p_y \\ &+ (vR_{33} - fR_{32})p_z = ft_y - vt_z \end{aligned} \tag{4}$$

Replacing the elements of rotation matrix in Eq. 4 with quaternions leads to two multivariate 2^{nd} order polynomial equations. For the sake of compactness, we are not expanding the Eq. 4 explicitly with quaternion. However, it is clear that the estimation of a quaternion is reduced to solving two multivariate 2^{nd} order polynomial equations that have four parameters. The reason we use both camera pose and 3D map points from SLAM to train the network instead of just camera pose is that in this way, we can in the largest extent reduce the SLAM accumulation error in map building.

As SLAM/SfM may suffer from accumulation error, we further constrain the camera pose and the learned map to be consistent from both spatial and spectral perspectives. During the SLAM/SfM processes, we obtain the camera pose for each frame together with the depth map or 3D point cloud. To spatially optimize the camera pose, we project the map points to the different images. Meanwhile, we match the features for the last frame's points to the current image feature map. Ideally, the matched point and projected pixels on the current frame should be as close as possible, because the projected point and the matched pixel should be from the same map points. We plan to reduce the position difference between the matched pixel and the projected pixel, as shown in Fig. 3(a). In Fig. 3(a), p and p' are the two projected pixels from the same 3D point P on two different images. pm is the matching pixel to the left 3D projected pixel p. Ideally, p' and pm should be the same pixel as they both correspond to the same pixel p. Therefore, we optimize the camera pose to make the distance between the p' and pm to be as small as possible. The spatial constraining loss is as Eq. 5.

$$\begin{aligned} &L_{spatial} = \sqrt{pm^2 - p'^2} = \sqrt{pm^2 - (K(R2 \cdot P + t2))^2}, \\ &pm \to p = K(R1 \cdot P + t1) \end{aligned} \tag{5}$$

In Eq. 5, we define the loss function to be the distance between the 3D projected pixel on the current frame p' and the pixel pm on the current frame corresponding to the 3D projected pixel p of the previous frame. $R2$ and $t2$ are the orientation and position matrices of the current camera pose, which project the 3D point P from the world coordinate to

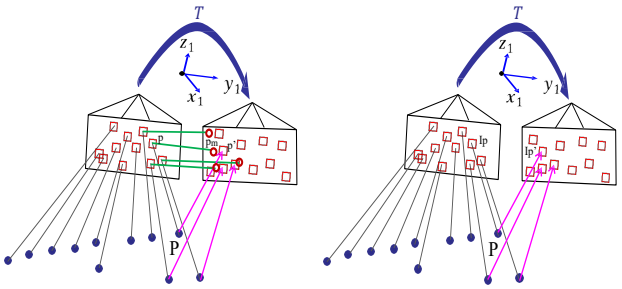


Fig. 3: Spatial (left) and spectral loss (right) in the network training process. Spatial loss constrains the distance between the matching features and back-projected points to be small. Spectral loss constrains corresponding pixels to be close in intensity.

the camera coordinate. The camera intrinsic matrix K further projects the 3D point in camera coordinate to the image plane. Similarly, $R1$ and $t1$ represent the orientation and position matrices of the previous camera pose. p' and pm are both the corresponding pixels of p from two different perspectives: one is from the feature matching, and another one is from the same 3D point re-projection. Through reducing the spatial distance between p' and pm by optimizing the camera pose, we can further improve the camera pose estimation accuracy.

In addition to the spatial constraints, we also optimize the network by spectral constraints. We project the 3D points to the 2D images based on estimated camera poses. The 2D pixels in different frames projected from the same 3D point should appear similar, which means their intensity values should be close to each other. Our objective is to reduce the spectral distance as small as possible, which can help us to optimize the camera pose, as shown in Fig. 3(b). In Fig. 3(b), the 3D point P is projected to p and p' in two frames. Ip and Ip' are the intensity of the two re-projected pixels. As the two pixels are corresponding to the same 3D point, their pixel intensity value should also be similar. We constrain the intensity represented by SSIM to be close to optimize the camera pose, as the following equation.

$$\begin{aligned} L_{spectral} &= 1 - SSIM(Ip, Ip') \\ p &= K(R1 * P + t1), \quad p' = K(R2 * P + t2) \end{aligned} \tag{6}$$

Similar to spatial constraint, $(R1, t1)$ and $(R2, t2)$ are extrinsics of the camera pose of the previous and current frame. K is the intrinsic of the camera. Extrinsics and intrinsics project the 3D point P to the image pixels p and p'. Through reducing the intensity value Ip and Ip' of the two corresponding pixels, we further optimize camera poses from the spectral perspective.

V. Experiments

To output camera poses, we first use a CNN to efficiently extract features and reduce feature representation dimensionality. The input of the Resnet is the query image followed by fine-tuning layers whose size of the receptive field in the network decrease from 7×7, 5×5 to 3×3 in order to extract finer region. Following a fully-connected layer and Softmax activation layer, the network returns a 7×1 transformation vector. During the test process, every single query image is

(a)

(b)

Fig. 4: 3D points back-projected to the images based on the camera pose inferred from the neural network after training. Blue points are the 3D points back-projected to the image. Red points are original pixels corresponding to the blue points. Yellow lines are the distance in corresponding pixel-to-3D point pairs.

Fig. 5: Vehicle paths generated without dynamic online training range determination function (left) and with this function (right). Red color marks the region of major difference.

the input for the network and only the new added layers are adjusted based on new inputs. The model is trained for 50 epochs. The batch size is 8 using Adam optimizer [17] where β_1 equals to 0.9 and β_2 is 0.999. The initial learning rate is λ =2×10^{-4} and set to be half after the first 30 epochs till to the end of training.

Our method is first tested on the public KITTI dataset [13], which is mainly for self-driving tasks. In KITTI dataset, the scenes are captured by two cameras under a stereo setting. During the map building process, the system maintains the left camera images and their associated camera poses in memory. The right images will be used to test the accuracy of our localization method. As left and right cameras share similarities in each pair of stereo images, we sample half (odd index) of the left camera images for map generation. The right camera images (even index) corresponding to non-selected left images will be applied for testing to avoid the same scene for testing. The training process is conducted only on the map generation data. Therefore, map generation and training process share the same sequence of data. And the testing data are different

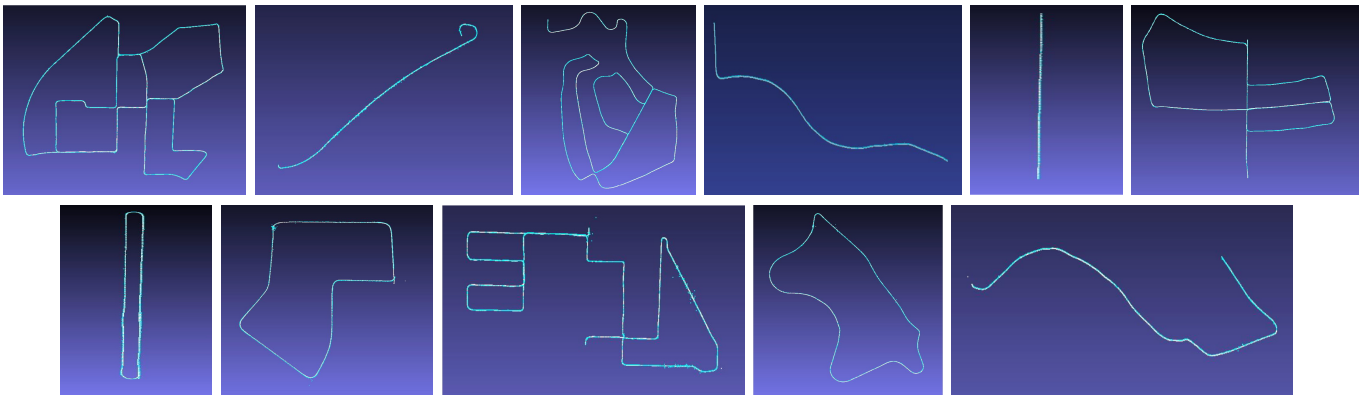


Fig. 6: Localization paths for all the 11 sequences on KITTI dataset. The white points are the ground truth, and the blue points are the predicted camera position from our localization system.

Performance	RMS error in position (m)			Variance for position			RMS error in attitude (rad)			Variance (Attitude)		
	x	y	z	x	y	z	x	y	z	x	y	z
Our framework	**0.3952**	**0.1381**	0.3491	**0.1538**	**0.0209**	0.1621	**0.0071**	**0.0101**	**0.0076**	**0.000019**	0.000061	**0.000022**
Superpoint [8]	0.4133	0.1458	0.3948	0.2268	0.03247	0.1763	0.0117	0.0161	0.0155	0.000047	0.000066	0.000053
D2-Net [10]	0.4056	0.1377	0.3329	0.2183	0.03347	0.1639	0.0114	0.0169	0.0147	0.000059	**0.000060**	0.000046
BRISK	0.7081	0.1876	0.4786	0.6052	0.0736	0.3104	0.0176	0.0281	0.0179	0.000069	0.00017	0.000074
ORB	0.4290	0.1555	**0.2760**	0.1820	0.0240	**0.0759**	0.0120	0.0163	0.0126	0.000036	0.000066	0.000039
SURF	0.6881	0.2492	0.5420	0.4711	0.0621	0.2936	0.0193	0.0302	0.0220	0.000093	0.00023	0.000121

TABLE I: Comparison of our localization framework and feature-based methods regarding the position and attitude estimation and variance.

video sequences compared with training data. To enlarge the variance of the testing data, each of these 11 sequences testing data is transformed with a homography matrix rotating the image from 0-20 degrees to change viewpoints. Random noise is added in the query video frames to change spot intensities and simulate unseen conditions. The same training and testing processes are conducted on all the methods to be compared with the same training and testing data. In

We train the neural network system based on camera pose optimization to reduce the 3D points back-projection spatial loss and spectral photometric error. We first visually verify the training effect based on the 3D back-projection error, as shown Fig. 4. In Fig. 4, 3D points are back-projected to the images using the camera pose inferred from the neural network represented by the blue points. Original pixels correspond to the 3D points during map generation stage are depicted by red color. Yellow lines between the red and blue points represent the distance between the back-projected 3D points and corresponding 2D pixels. As can be observed, the small yellow lines demonstrate the learning process is effective to output a correct camera pose. To avoid confusion, this verification is conducted on the training data split, which is used for map generation as well, as the testing data split does not involve the local feature extraction step. The rest experiments are conducted on testing sequences without overlapping with the training data.

To further verify the effect of dynamic scope to camera pose prediction, we apply the fixed online training scope without dynamic online training range determination, indicating the training scope is always kept the same size around the previous frame's location, as Fig. 5. From both Fig. 5(a) and 5(b), the offset points from ground truth mainly are from the large rotation area where the view changes dramatically, after turning off the dynamic range determination function. When vehicle turning at corners, the view has a relatively large change by rotation, keeping the updating region too tight may result in the loss of true positive training samples. From Fig. 5, dynamically adjusting the local training scope will positively affect the localization accuracy. A fixed training region may introduce bias to update the network. The localization accuracy of all the paths is displayed in Fig. 6.

We show our deep neural network's performance with regards to RMS error in position and attitude compared with other dominant features and deep learning features (Superpoint [8] and D2-Net [10]) fine-tuned on 10,000 random selected images on the testing datasets as Table I. The localization method of classical features is based on the same strategy of our deep neural network with local search (first global searching feature correspondences in 3D map and then refine camera pose based on local feature matching to neighboring images) and dynamic search region control based on the same rotation and translation ratio. From Table I, our deep neural network achieves the best performance in rotation and translation measurement. We can see that the local search and dynamic search scope control can lead to the high localization accuracy for feature-based methods as well. However, our convolutional neural network-based approach can achieve the best performance in terms of rotation and translation estimation, including the average precision and variance, indicating the stable performance of our method. Superpoint and D2-Net are better than BRISK and SURF, and better than ORB in several categories. In testing, D2-Net and Superpoint have the largest correct matching rate. The less detected features in

	Our method	V-LOAM [40]	LOAM [39]	SOFT2 [7]	GDVO [43]	Stereo DSO [34]	PMO [11]	MonoROCC [4]	SfMLearner [42]	Vid2depth [23]
Translation error	**0.28%**	0.63%	0.64%	0.65%	0.86%	0.93%	2.05%	1.11%	2.33%	1.86%
Rotation error (deg/m)	**0.000066**	0.0014	0.0014	0.0014	0.0031	0.002	0.0051	0.0028	0.0063	0.0057

TABLE II: Comparison with other top performed SLAM and VO methods on KITTI dataset

D2-Net and Superpoint than ORB affected final performance. Our method relieves from feature matching and achieves best overall accuracy.

We also test our method in comparison with other state-of-the-art localization methods. We show the position and attitude errors in Fig. 7. A smaller number indicates better performance. Among all the state-of-the-art localization methods, Probabilistic model (stereo and monocular) [3], Global co-visibility [19], Randomized tree with binary search [12], and 2D-to-3D [29] are the methods relying on feature matching and geometric verification without the use of deep neural network. PoseNet [16], Expert Sample Consensus [2], and KFNet [41] are based on convolutional neural networks to infer the camera pose. Benefitting the network training structure, and online adjustment under dynamic scope control, the proposed neural network system achieves the best accuracy for both translation and rotation estimation compared with other methods, indicating the effect of our neural network in terms of localization tasks.

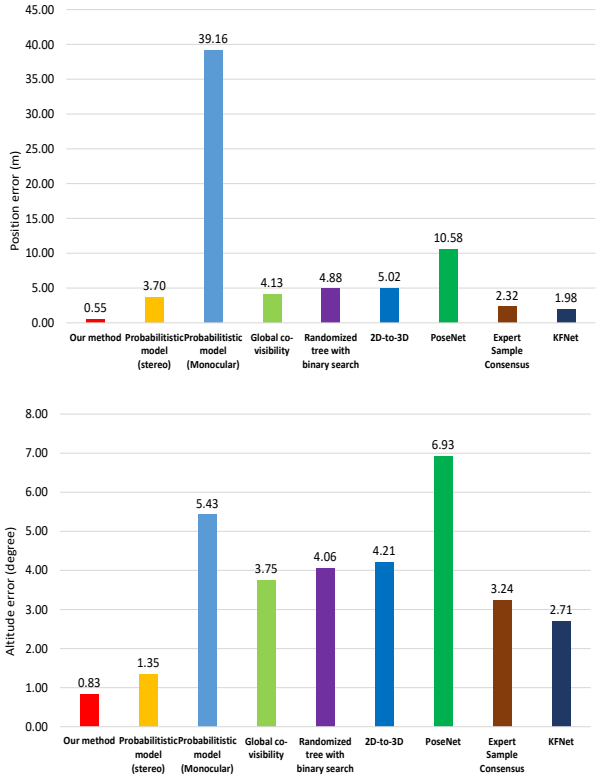


Fig. 7: The error of position (upper) and attitude (bottom) compared with state-of-the-art localization methods (Probabilistic model (stereo and monocular) [3], Global co-visibility [19], Randomized tree with binary search [12], 2D-to-3D [29], PoseNet [16]), Expert Sample Consensus [2], and KFNet [41].

To verify the effect of each component in our pipeline, a quantitative ablation study is added as Table III. From the table, local fine-tuning can avoid global confusion, which plays a more critical role than dynamic scope change.

Error for ablation study	Full pipeline	Without local fine-tuning	Without dynamic scope change
Position (m)	0.55	1.63	1.08
Attitude (deg)	0.83	1.22	0.97

TABLE III: Ablation study for quantitative analysis

We understand that SLAM and VO may contain the accumulation error problem. Here we also want to compare with the most advanced SLAM and VO methods to prove the accuracy of our method, as this is the closest comparison category in KITTI website, in addition to the comparison with localization methods with map priors that we have already shown in Fig. 7. We compare the most recent and best performed SLAM and VO algorithms on KITTI odometry category (V-LOAM (Lidar+camera) [40], LOAM (Lidar) [39], SOFT2 (stereo) [7], GDVO (stereo) [43], Stereo DSO [34], PMO [11], MonoROCC [4]), which mainly based on classical geometrical SLAM/VO methods, and deep neural network based methods (SfMLearner [42] and Vid2depth [23]), as Table II. Our method performs significantly better than the SLAM and visual odometry methods from translation and rotation error. It is noticeable that deep neural network based methods perform worse on camera pose estimation than classical SLAM/VO methods, which we consider it is because the optimization condition (e.g., loop closure) in epipolar geometry based methods maintain tighter constraints. This result is consistent with Sattler et al. [30]'s finding. This is also part of the reason we apply classical geometrical SLAM as the base to build the localization map. The overall performance demonstrates that our method can provide reliable and superb performance on vehicle ego-motion estimation.

VI. Conclusion

This paper presents a localization method for self-driving vehicles that can accurately estimate the vehicle position and orientation. We build maps composed of images with their associated camera pose and index all the camera poses of the map. We first train a CNN based on the training images, and their camera poses with the global map. We further extract the closest images and camera poses to train the network online locally. The current vehicle pose is estimated through the fine-tuned network globally and then locally. Based on the translation and rotation motion magnitude ratio, we determine the dominant motion of the vehicle and dynamically change the scope to update the network. Experiments demonstrate accurate localization effects.

Acknowledgement

This material is based upon work supported by the National Science Foundation under Award No. 2105257.

REFERENCES

[1] A. Bergamo, S. N. Sinha, and L. Torresani. Leveraging structure from motion to learn discriminative codebooks for scalable landmark classification. In *CVPR*, pages 763–770, 2013. 2

[2] E. Brachmann and C. Rother. Expert sample consensus applied to camera re-localization. In *ICCV*, pages 7525–7534, 2019. 2, 7

[3] M. A. Brubaker, A. Geiger, and R. Urtasun. Map-based probabilistic visual self-localization. *TPAMI*, 38(4):652–665, 2016. 7

[4] M. Buczko and V. Willert. Monocular outlier detection for visual odometry. In *IEEE IV*, pages 739–745, 2017. 7

[5] R. Castle, G. Klein, and D. W. Murray. Video-rate localization in multiple maps for wearable augmented reality. In *ISWC*, pages 15–22, 2008. 2

[6] D. Crandall, A. Owens, N. Snavely, and D. Huttenlocher. Discrete-continuous optimization for large-scale structure from motion. In *CVPR*, pages 3001 –3008, 2011. 2

[7] I. Cvišić, J. Ćesić, I. Marković, and I. Petrović. Soft-slam: Computationally efficient stereo visual slam for autonomous uavs. *Journal of field robotics*, 2017. 2, 7

[8] D. DeTone, T. Malisiewicz, and A. Rabinovich. Superpoint: Self-supervised interest point detection and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 224–236, 2018. 6

[9] M. Donoser and D. Schmalstieg. Discriminative feature-to-point matching in image-based localization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 516–523, 2014. 2

[10] M. Dusmanu, I. Rocco, T. Pajdla, M. Pollefeys, J. Sivic, A. Torii, and T. Sattler. D2-net: A trainable cnn for joint description and detection of local features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8092–8101, 2019. 6

[11] N. Fanani, A. Stürck, M. Ochs, H. Bradler, and R. Mester. Predictive monocular odometry (pmo): What is possible without ransac and multi-frame bundle adjustment? *Image and Vision Computing*, 68:3–13, 2017. 7

[12] Y. Feng, L. Fan, and Y. Wu. Fast localization in large-scale environments using supervised indexing of binary features. *TIP*, 25(1):343–358, 2016. 2, 7

[13] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012. 5

[14] A. Irschara, C. Zach, J.-M. Frahm, and H. Bischof. From structure-from-motion point clouds to fast location recognition. In *CVPR*, pages 2599–2606, 2009. 2

[15] H. Jégou, M. Douze, C. Schmid, and P. Pérez. Aggregating local descriptors into a compact image representation. In *CVPR*, pages 3304–3311, 2010. 2

[16] A. Kendall and R. Cipolla. Geometric loss functions for camera pose regression with deep learning. In *CVPR*, pages 2938–2946, 2017. 2, 7

[17] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[18] Y. Li, N. Snavely, and D. P. Huttenlocher. Location recognition using prioritized feature matching. In *ECCV*, pages 791–804, 2010. 2

[19] L. Liu, H. Li, and Y. Dai. Efficient global 2D-3D matching for camera localization in a large-scale 3D map. *ICCV*, pages 2372–2381, 2017. 2, 7

[20] G. Lu, N. Sebe, C. Xu, and C. Kambhamettu. Memory efficient large-scale image-based localization. *Multimedia Tools and Applications (MTA)*, 74(2):479–503, 2015. 2

[21] G. Lu, Y. Yan, L. Ren, J. Song, N. Sebe, and C. Kambhamettu. Localize me anywhere, anytime: A multi-task point-retrieval approach. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2434–2442, 2015. 2

[22] G. Lu, Y. Yan, N. Sebe, and C. Kambhamettu. Knowing where I am: Exploiting multi-task learning for multi-view indoor image-based localization. In *British Machine Vision Conference (BMVC)*, 2014. 2

[23] R. Mahjourian, M. Wicke, and A. Angelova. Unsupervised learning of depth and ego-motion from monocular video using 3d geometric constraints. In *CVPR*, pages 5667–5675, 2018. 7

[24] I. Melekhov, J. Ylioinas, J. Kannala, and E. Rahtu. Relative camera pose estimation using convolutional neural networks. In *ACIVS*, 2017. 2

[25] M. J. M. M. Mur-Artal, Raúl and J. D. Tardós. ORB-SLAM: a versatile and accurate monocular SLAM system. *ToG*, 31(5):1147–1163, 2015. 3

[26] R. Mur-Artal and J. D. Tardós. ORB-SLAM2: an open-source SLAM system for monocular, stereo and RGB-D cameras. *ToR*, 33(5):1255–1262, 2017. 1, 3

[27] E. Parisotto, D. Singh Chaplot, J. Zhang, and R. Salakhutdinov. Global pose estimation with an attention-based recurrent network. In *CVPRW*, pages 237–246, 2018. 2

[28] D. P. Robertson and R. Cipolla. An image-based system for urban navigation. In *British Machine Vision Conference (BMVC)*, pages 819–828, 2004. 2

[29] T. Sattler, B. Leibe, and L. Kobbelt. Efficient & effective prioritized matching for large-scale image-based localization. *TPAMI*, 39(9):1744–1756, 2017. 2, 7

[30] T. Sattler, Q. Zhou, M. Pollefeys, and L. Leal-Taixe. Understanding the limitations of cnn-based absolute camera pose regression. In *CVPR*, pages 3302–3312, 2019. 2, 7

[31] G. Schindler, M. Brown, and R. Szeliski. City-scale location recognition. In *CVPR*, 2007. 2

[32] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: exploring photo collections in 3D. *ACM Transition on Graphics (ToG)*, 25(3):835–846, 2006. 2

[33] H. Taira, M. Okutomi, T. Sattler, M. Cimpoi, M. Pollefeys, J. Sivic, T. Pajdla, and A. Torii. Inloc: Indoor visual localization with dense matching and view synthesis. In *CVPR*, pages 7199–7209, 2018. 2

[34] R. Wang, M. Schwörer, and D. Cremers. Stereo dso: Large-scale direct sparse visual odometry with stereo cameras. In *ICCV*. 7

[35] S. Wang, R. Clark, H. Wen, and N. Trigoni. Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks. In *ICRA*, 2017. 2

[36] J. Xiao, J. Chen, D.-Y. Yeung, and L. Quan. Structuring visual words in 3D for arbitrary-view object localization. In *European Conference on Computer Vision (ECCV)*, pages 725–737, 2008. 2

[37] K. Xuan, G. Zhao, D. Taniar, M. Safar, and B. Srinivasan. Voronoi-based multi-level range search in mobile navigation. *MTA*, 53(2):459–479, 2011. 2

[38] Z. Yin and J. Shi. Geonet: Unsupervised learning of dense depth, optical flow and camera pose. In *CVPR*, 2018. 2

[39] J. Zhang and S. Singh. Loam: Lidar odometry and mapping in real-time. In *RSS*, 2014. 7

[40] J. Zhang and S. Singh. Visual-lidar odometry and mapping: Low-drift, robust, and fast. In *ICRA*, pages 2174–2181, 2015. 2, 7

[41] L. Zhou, Z. Luo, T. Shen, J. Zhang, M. Zhen, Y. Yao, T. Fang, and L. Quan. Kfnet: Learning temporal camera relocalization using kalman filtering. In *CVPR*, pages 4919–4928, 2020. 2, 7

[42] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe. Unsupervised learning of depth and ego-motion from video. In *CVPR*, pages 1851–1858, 2017. 7

[43] J. Zhu. Image gradient-based joint direct visual odometry for stereo camera. In *IJCAI*, pages 4558–4564, 2017. 7