

Attention-based Vehicle Self-Localization with HD Feature Maps

Nico Engel, Vasileios Belagiannis and Klaus Dietmayer

Abstract—We present a vehicle self-localization method using point-based deep neural networks. Our approach processes measurements and point features, i.e. landmarks, from a high-definition digital map to infer the vehicle’s pose. To learn the best association and incorporate local information between the point sets, we propose an attention mechanism that matches the measurements to the corresponding landmarks. Finally, we use this representation for the point-cloud registration and the subsequent pose regression task. Furthermore, we introduce a training simulation framework that artificially generates measurements and landmarks to facilitate the deployment process and reduce the cost of creating extensive datasets from real-world data. We evaluate our method on our dataset, as well as an adapted version of the Kitti odometry dataset, where we achieve superior performance compared to related approaches; and additionally show dominant generalization capabilities.

I. INTRODUCTION

The localization of autonomous agents in an unknown environment with a high-definition (HD) digital map as prior is a key component in state-of-the-art robotic systems, including self-driving cars [1]. It is important for other automated driving modules such as the human-vehicle interaction [2], trajectory prediction in perception [3] and tracking [4]. The goal is to infer the vehicle’s pose, which is comprised of a position and an orientation [5]. Normally, the pose is estimated in the global or local coordinate system relative to the digital map, which can then be used to extract useful information from the map [6]. Furthermore, the localization accuracy is expected to be around 50 cm [7] for real-world applications.

The standard vehicle self-localization approach is to rely on global navigation satellite systems (GNSS), e.g. GPS, to obtain the pose estimate. However, they fail to meet the required localization accuracy, especially in urban environments, and suffer from multi-path effects, and blocked line-of-sight to the satellites, which further deteriorates the localization quality [8]. Also, the GPS signal can be combined with correction data (dGPS) and inertial measurement units (IMU) to further improve the localization accuracy, but due to the high acquisition and operation costs, it is not sustainable for mass deployment. Alternatively, one can create high-definition digital maps that contain distinct and easily recognizable high or low-level features, extracted from the on-board sensor data, such as camera, laser and radar. The sensor measurements are registered with the features from the digital map to infer the vehicle’s relative pose inside the map frame. Several methods have been proposed

in the field of robotics to perform the inference, ranging from simple point to point registration approaches, e.g. ICP [9], to more sophisticated methods that utilize filtering approaches, such as the Extended Kalman-Filter (EKF) or the particle filter [10]. The association of the measurements to the map features, often called landmarks, is usually performed by assigning the most likely measurements to each landmark using either a heuristic or probabilistic approach. In urban environments, these algorithms suffer from erroneous associations from noise that is caused by the highly dynamic scenarios with numerous road participants [11].

Currently, the promising way for localization is the data-driven approach [12], [13]. An appropriate dataset has to be created that ideally captures most of the desired areas of operation for obtaining a model that generalizes well during deployment. However, it is not feasible to create and label datasets in hundreds of cities around the world and updating them whenever environmental changes or new scenarios emerge. We also follow the data-driven approach but propose a simulation framework to generate synthetic training data. We can train a deep neural network to perform localization without the necessity of acquiring a plethora of real-world data. Although DeepLocalization [12] is related to our approach, it not able to learn local data relations.

We define the localization process as two tasks, namely the landmark to measurement association and the point cloud registration. For the association process, we present an attention mechanism to score each landmark assignment for subsequently learning the most probable associations. Based on the weighted representation of the measurements and landmarks, we generate local features that are used for the point cloud registration process. Then, another attention operation combines all measurements and landmarks to predict the vehicle’s pose. For the inference process, we propose a GPS-based and a filter-based approach, that allows us to estimate a pose offset based on a previous pose similar to our training pipeline. We show that employing the attention mechanism for the different tasks greatly improves the localization accuracy on different datasets compared to the related work, especially in dynamic and complex environments. Additionally, we propose a simulation framework that enables us to train the network by artificially generating landmark and measurements. We do this by designing a probabilistic model that is inspired by the spatial occurrence of real map landmarks. The evaluation shows, that a network trained on artificial samples is able to generalize to real-world data, thus significantly reducing the resources needed to deploy our approach while still meeting the required accuracy in urban scenarios of about 50 cm.

Authors are with Institute of Measurement, Control and Microtechnology, Ulm University, Albert-Einstein-Allee 41, 89081 Ulm, Germany {firstname.lastname}@uni-ulm.de. Project Page: <https://github.com/engelnico/deeplocalization>.

II. RELATED WORK

In the following, we compare and discuss related approaches for vehicle localization. We distinguish between traditional model-based approaches, such as localization methods that use filter-based algorithms, and learning-based approaches, i.e. deep neural networks.

A. Model-based approaches

In the field of robotics, different model-based methods have been proposed to solve the task of self-localization by using a digital map as prior to estimate a pose or by simultaneously constructing a map while localizing itself in an unknown environment, i.e. SLAM.

Censi [14] proposes PLICP, an improvement to the iterative closes point algorithm by introducing a point-to-line metric instead of the originally used point-to-point metric that can be used to register measurements with a feature-based map. Moreover, PLICP improves the convergence properties while also converging in a finite number of steps. Contrary to this, Fontanelli *et al.* propose a RANSAC-based localization approach using lidar measurements that is both accurate and copes with noisy measurements [15]. Using an Extended Kalman-Filter, Teslic *et al.* introduce a localization framework that combines wheel encoders to predict the robot's motion and laser scans to correct the robots pose by matching the measurements with a digital map [16].

A very well-known approach by Dellaert *et al.* implements a probabilistic model by representing the robot's state space as a particle-based density [17]. The Monte-Carlo Localization (MCL) is able to efficiently represent arbitrary distributions, in this case the robot's pose, while being more accurate and requiring less memory compared to related methods. Later, Thrun *et al.* [5] improve the algorithm by introducing two methods of generating particles in the estimation and learning a kernel density tree to enable faster sampling. Based on the Monte-Carlo Localization, Montemerlo and Thrun develop the FastSLAM algorithm, which combines the MCL with an EKF to generate a landmark-based map while enabling a robot localization in unknown environments [18]. Additionally, Stuebler *et al.* propose the RFS-MCL [19], that combines the Random-Finite Set theory with a particle filter-based localization approach. Since then, many SLAM algorithms have been proposed, e.g. [20], [21], with a heavy focus on camera-based approaches, e.g. [22], [23]. This trend of using vision-based systems can also be observed for the localization task, i.e. visual odometry (VO) [24].

B. Learning-based approaches

Besides model-based approaches, recent work focused on learning-based methods using deep neural networks. Since traditional neural networks like convolutional neural networks (CNN) or simple multi-layer perceptrons (MLP) require the input to be structured and ordered, most approaches use a vision-based system, i.e. camera images, to perform the localization task.

Yang *et al.* [25] propose SANet, a scene agnostic framework for camera localization, where they separate the scenes

and model parameters and learn a hierarchical scene representation. Thus, SANet is independent of the scenes and can easily be deployed to online tasks, such as navigation and SLAM. Radwan and Valada introduce a network architecture called VLocNet++ [13], where they combine the learning of semantics, regressing the global pose of a camera and odometry to exploit the relationships between the tasks to increase the overall performance. On the other hand, Kendall *et al.* propose PoseNet [26] which is a real-time localization system based on convolutional neural networks, that is designed to regress the cameras pose from a single RGB image. Similar to the localization methods using ICP variants to register two point clouds, DeepICP [27] is an end-to-end trainable neural network, where a key-point detector is trained such that it focuses on stationary objects and avoids dynamic objects. This is achieved by generating corresponding points by matching possible candidates based on learned probabilities. Wang *et al.* introduce an attention-based camera relocation system [28] that is robust to outliers and dynamic illumination conditions. The attention mechanism is used to learn a geometrical representation that focuses on robust features. Contrary to our approach, AtLoc is restricted to camera images only, whereas we focus on a generic feature representation that can handle multi-modal sensor measurements. Finally, Lu *et al.* propose L3-Net [29], a learning-based lidar localization framework that incorporates multiple network structures, such as convolutional neural networks (CNN) and recurrent neural networks (RNN), to learn local descriptors for 3D point cloud data. However, due to multiple network stages that increase the overall complexity and model size, the computational time suffers and in some cases violates our real-time requirement of about 100 ms.

A comprehensive survey on the topic of deep-learning based localization methods can be found in [11].

III. METHODOLOGY

In this section, we introduce our methodology, the attention-based model as well as our training and inference process. We assume 2D multi-modal sensor measurements as one input set, denoted by $M = \{m_1, \dots, m_\nu\}$, $m_{(\cdot)} \in \mathbb{R}^2$. Furthermore, we consider landmarks from our digital map $L = \{l_1, \dots, l_\mu\}$, $l_{(\cdot)} \in \mathbb{R}^2$ as the second input to our network. Landmarks are easily recognizable and static objects, e.g. trees, traffic lights, poles, that were generated from sensor measurements during the map building process, that we describe in Section IV. Here, it is important to note that both input point sets are unordered and the cardinality of the sets, i.e. the number of measurements ν and landmarks μ , is not known in advance.

A. Problem Formulation

The goal of localizing an agent is to find the relative pose $p = [x, y, \varphi]$ within a given coordinate system, e.g. the map frame. For this, we consider current multi-modal sensor measurements with the aim of matching them to the landmarks from the digital map which are in the vehicle's

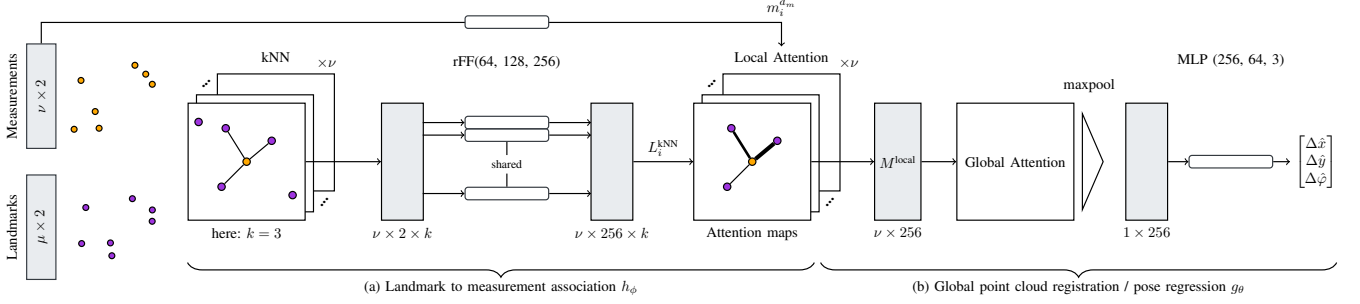


Fig. 1. Overview of our network architecture. First we group all landmarks in the vicinity of each measurement using the k-Nearest Neighbour algorithm. Then, we learn the most probable matching by employing the local attention mechanism, which weights the associations (a). Finally, we use this representation to register the measurements and the landmarks and infer a pose correction vector (b).

field of view. The matching set function is defined as following:

$$f(M, L) \rightarrow [x, y, \varphi]. \quad (1)$$

We split this set matching problem into two subtasks: a) The landmark to measurement association, where we try to match both input point sets such that the most probable landmarks are assigned to the corresponding sensor measurements. For this, we employ the attention mechanism, which learns to score each possible landmark to measurement association. b) A global point cloud registration process in higher dimensional space using the local associations. The result of the registration process is a spatial transformation that describes the mapping of the landmarks to the measurement set, i.e. the pose prediction. In this work, we approximate the matching function (1) with a deep neural network, given by:

$$f(M, L) \approx g_\theta(h_\phi(M, L)), \quad (2)$$

where we denote the set of learnable parameters by θ, ϕ . Furthermore, the landmark to measurement association process is defined as $h_\phi(\cdot)$ and the point cloud registration together with the subsequent pose regression is combined as $g_\theta(\cdot)$.

B. Attention-based model

The measurements M and the landmarks (features) L serve as input to our network, which is visualized in Fig. 1. In the first step of the landmark to measurement association process, see Fig. 1a), we employ a k-Nearest Neighbour (kNN) search to find the k closest landmarks to each measurement. We perform the kNN to take into account the spatial relationship between measurements and landmarks: Measurements ideally originate from landmarks, therefore we only consider landmarks in the vicinity of the measurements for the association process. Then, for every measurement we calculate the Euclidean distance of the k associated landmarks and transform the result into higher dimensional space $d_m = 256$ using a row-wise feed forward (rFF) network. A rFF is a feed-forward network that receives only one point as input but shares its weight with all subsequent points [30]. Then we employ a multi-head attention module [31], as it can capture context and

higher order dependencies of point sets [32]. Furthermore, we leverage the fact that the attention mechanism scores the input sets, thus we learn a weighting of the most probable association. For that reason, we define the attention function \mathcal{A} that describes a mapping of N queries $Q \in \mathbb{R}^{N \times d}$ and N_k key-value pairs $K \in \mathbb{R}^{N_k \times d}$, $V \in \mathbb{R}^{N_k \times d}$ to the output space $\mathbb{R}^{N \times d}$ as follows

$$\mathcal{A}(Q, K, V) = \sigma \left(\frac{QK^T}{1/\sqrt{d}} \right) V, \quad (3)$$

which is also known as scaled dot product attention $\mathcal{A}(Q, K, V) : \mathbb{R}^{N \times d_k}, \mathbb{R}^{N_k \times d}, \mathbb{R}^{N_k \times d} \rightarrow \mathbb{R}^{N \times d}$ [31]. The activation function $\sigma(\cdot)$ that also denotes the aforementioned learned score is usually given by the softmax function

$$\sigma_{\text{softmax}}(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)}, \quad (4)$$

with $\sigma(\cdot) : \mathbb{R}^{N \times d}, \mathbb{R}^{N_k \times d} \rightarrow \mathbb{R}^{N \times N_k}$. Instead of performing a single attention operation, we follow the ideas of [31] and employ multi-head attention, where the queries, keys and values are first linearly projected h times using independent feed-forward networks to incorporate spatial relations in different subspaces. Then, the attention function (3) is applied in parallel to each of the h projections and the result is concatenated and linearly projected again. The operation is described by:

$$\text{Multihead}(Q, K, V) = (\text{head}_1 \oplus \dots \oplus \text{head}_h)W^O, \quad (5)$$

where $\text{head}_i = \mathcal{A}(QW_i^Q, KW_i^K, VW_i^V)$ with learnable parameter matrices $W_i^Q \in \mathbb{R}^{d \times d}$, $W_i^K \in \mathbb{R}^{d \times d}$, $W_i^V \in \mathbb{R}^{d \times d}$ and $W^O \in \mathbb{R}^{d \times d}$. The \oplus operation denotes matrix concatenation. Finally, Vaswani *et al.* define the multi-head attention block that consists of the attention operation (3) and residual connections followed by layer normalization [33] as follows:

$$\mathcal{A}^{\text{MH}}(X, Y) = \text{LayerNorm}(S + \text{rFF}(S)), \quad (6)$$

where $\mathcal{A}^{\text{MH}} : \mathbb{R}^{N \times d}, \mathbb{R}^{N_k \times d} \rightarrow \mathbb{R}^{N \times d}$, and the sublayer S is defined as $S = \text{LayerNorm}(X + \text{Multihead}(X, Y, Y))$.

Furthermore, X and Y denote arbitrary input sets. In the following we set $d = d_m = 256$.

As visualized in Fig. 1, we employ local attention for subtask (a) and define it as follows:

$$\mathcal{A}_i^{\text{local}} := \mathcal{A}^{\text{MH}}(m_i^{d_m}, L_i^{\text{kNN}}), i = 1, \dots, \nu, \quad (7)$$

where we take each measurement m_i , project it to model dimension d_m using the rFF depicted in Fig. 1 and apply the multi-head attention mechanism (6) with the k associated landmarks from the kNN algorithm. Thus, we have $m_i^{d_m} \in \mathbb{R}^{1 \times d_m}$, $L_i^{\text{kNN}} \in \mathbb{R}^{k \times d_m}$ and $\mathcal{A}_i^{\text{local}} \rightarrow \mathbb{R}^{1 \times d_m}$. By employing the local attention operation (7) we generate latent features for every measurement that contain a weighted representation of all nearby landmarks, see Equation (3). Thus, our network is able to learn the most probable landmark to measurement association which is visualized by the weight of the connection in the attention maps in Fig. 1. After applying Equation (7) to each measurement and concatenating the result, we obtain the local feature matrix $M^{\text{local}} \in \mathbb{R}^{\nu \times d_m}$. Then, we employ another self-attention operation for subtask (b) to aggregate global information

$$\mathcal{A}^{\text{global}} := \mathcal{A}^{\text{MH}}(M^{\text{local}}, M^{\text{local}}), \quad (8)$$

with $\mathcal{A}^{\text{global}} : \mathbb{R}^{\nu \times d_m}, \mathbb{R}^{\nu \times d_m} \rightarrow \mathbb{R}^{\nu \times d_m}$. The global attention mechanism (8) relates the local features of the matched landmarks and measurements against each other, allowing the network to learn the point set registration process and capture higher order dependencies. Afterwards, a maxpooling operation [34] follows to produce global features of fixed length that are invariant to input point permutations and arbitrary input set cardinality. Here, it is important to note that the attention operation itself is invariant to input point permutations as well [31]. Finally, we infer the vehicle's pose using the learned global features with a simple feed forward output head as shown in Fig. 1.

C. Model training

For training, we follow the same protocol as in [12] to treat the problem as a regression task [35] and rely on the ground-truth pose from our dGPS system as a starting point. Then, all landmarks in the vehicle's field of view (FoV) are loaded from the digital map and are transformed from UTM coordinates into the vehicle's frame using the dGPS pose. At that point, both inputs, namely the landmarks and measurements, are available in the same coordinate system. To imitate real-world conditions during training, we additionally add a small translation and rotation to all landmarks by sampling both a position and a rotation offset from a uniform distribution \mathcal{U} on the interval $[-\sigma, \sigma]$. Thus, we systematically simulate the inaccurate GPS measurement p_{GPS} . The advantage of this method is that it is no longer necessary to determine the global pose, but instead we infer a synthetically generated pose offset. Therefore, Equation (1) becomes

$$f(M, L) \rightarrow [\Delta x, \Delta y, \Delta \varphi], \quad (9)$$

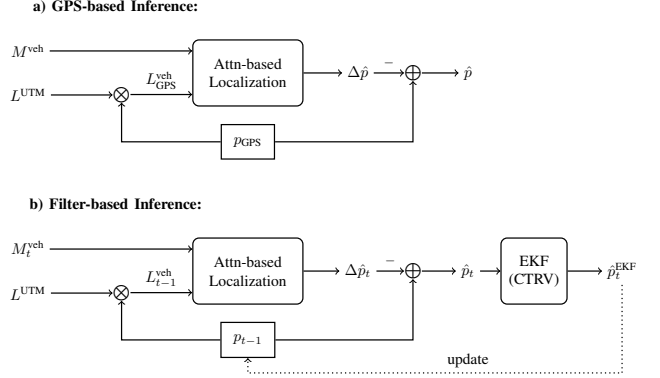


Fig. 2. Overview of the proposed inference methods.

with the pose offset $\Delta p = [\Delta x, \Delta y, \Delta \varphi]$. The predicted global pose \hat{p} can then be easily obtained by

$$\hat{p} = p_{\text{GPS}} - \Delta \hat{p}, \quad (10)$$

where the prediction of the network is denoted by $\Delta \hat{p}$. This method allows us to generate new training samples every epoch, as the randomly sampled pose offset directly serves as the training label. Furthermore, it simplifies the training of the network because we found that inferring a small offset to be more numerically stable in the optimization process compared to using the global UTM coordinate system. Since the rotation and translation offset are given in different units, i.e. rad and m, we train our network with the same loss function that learns a weighting factor for each of the loss parts, as proposed in [36], [12]. In particular, we employ the L2-Loss for each of the pose component predictions

$$L_{\text{tran}} = \mathbb{E}[(\Delta \hat{x} - \Delta x)^2] + \mathbb{E}[(\Delta \hat{y} - \Delta y)^2], \quad (11a)$$

$$L_{\text{rot}} = \mathbb{E}[(\Delta \hat{\varphi} - \Delta \varphi)^2], \quad (11b)$$

where L_{tran} is the translation loss and L_{rot} the rotation loss. For the total multi-task loss, we combine the loss terms

$$L_{\text{multi}} = L_{\text{tran}} e^{-s_{\text{tran}}} + s_{\text{tran}} + L_{\text{rot}} e^{-s_{\text{rot}}} + s_{\text{rot}}, \quad (12)$$

where $s_{\text{tran}} = \log \sigma_{\text{tran}}^2$, $s_{\text{rot}} = \log \sigma_{\text{rot}}^2$. Following the ideas from Kendall *et al.*, $\sigma_{\text{tran}}, \sigma_{\text{rot}}$ are the homoscedastic uncertainties, i.e. learnable parameters for weighting each loss function [36].

D. Inference

Similar to [12], we present two different inference approaches, which are also visualized in Fig. 2.

GPS-based inference is the default inference configuration that resembles the training process from Section III-C. Again, we load all landmarks from the digital map that are in the vehicle's field of view L^{UTM} and transform them into the vehicle's coordinate system $L_{\text{GPS}}^{\text{veh}}$ using the noisy GPS measurement p_{GPS} , which is usually supplied in global coordinates, e.g. UTM. Since the pose that is used for transforming the landmarks to the vehicle coordinate system is noisy and inaccurate, it induces a small shift and rotation to the landmarks, which resembles the synthetic and randomly



Fig. 3. Our test track in Ulm-Lehr. Landmarks are shown as red dots (•) and we visualize the alternative Train / Test split as ① / ②, respectively.

sampled pose offset that is applied to the ground-truth pose in the training process. As mentioned above, the goal is to infer a pose correction vector $\Delta\hat{p}$ that is applied to the initial pose estimate p_{GPS} in order to obtain the global pose \hat{p} , see Fig. 2 a). This inference approach can directly be applied without further adjustments and is our recommended configuration when a commercially available GPS sensor with noisy measurements is installed.

Filter-based inference extends our system architecture with an Extended Kalman-Filter in order to obtain a temporal filtered and smooth pose estimate. To highlight the incorporation of the time domain, we slightly change our notation as shown in Fig. 2 b). The algorithm requires an previous pose estimate p_{t-1} which can either be supplied by a commercially available and noisy GPS system for initialization or from the previous time step $t-1$. Similar to the GPS-based inference, we use p_{t-1} to transform the landmarks from the digital map L^{UTM} to the vehicle coordinate system L^{veh} and use it as input to our network together with the current measurements. Since landmarks that are transformed with the previous pose estimate together with current measurements are used as input to our network, we again obtain a small shift and rotation due to the vehicle's motion. The correction output $\Delta\hat{p}_t$ is applied to the previous pose estimate p_{t-1} , which is then used as measurement input to the Extended Kalman-Filter with a constant turn rate and velocity motion model (CTRV). The output \hat{p}_t^{EKF} is a smoothed estimate of the global pose. In the next time step, we use this estimate as the initial pose p_{t-1} . The advantage of the filter-based inference approach is that it requires only one GPS measurement for initialization and can then be used as a stand-alone localization method. Furthermore, in [12] we show that the computational overhead of the EKF is negligible.

IV. DATASET

In this section, we introduce our dataset which consists of a high-definition (HD) digital map with point features, i.e. landmarks, and multiple recordings with sensor mea-

surements on our test track in Ulm, Lehr, that is shown in Fig. 3. The test track is about 6 km long with urban and rural roads, intersections, roundabouts and merging lanes. For the measurement dataset, we recorded multiple runs in November 2018 using our autonomous vehicle which is equipped with a camera, radar and laser sensors, as well as a dGPS system that provides high precision pose information that we use for the training process and for evaluation purposes. Additionally, we employ a simple pre-processing pipeline where we cluster the raw measurements to obtain more stable and reliable point features. For this, we cluster the laser and radar measurements using the density-based DBSCAN [37] and for camera images, we extract features using the maximally stable extremal region algorithm (MSER) [38]. For every measurement time step, we additionally record the vehicle's high-precision pose using the dGPS system. Finally, for training our network we introduce two train / test splits: 1) a uniform distribution, in which we use six complete runs for the training split and two additional runs for the test split, and 2) a spatial split visualized as ① / ② in Fig. 3, to demonstrate the generalization capabilities of our approach in unseen environments. Our high-definition (HD) map was generated one year before the measurement dataset to incorporate a diverse data distribution that closely resembles the dynamic environment prevalent in most urban scenarios. As mentioned above, our map contains generic landmarks (features) that are created from our pre-processed sensor measurements. During our map building process, we classify a single measurement as static and easily recognizable when it is seen multiple times on different runs. For that, a Bernoulli-filtering approach is employed as proposed by Stuebler *et al.* [39], that assigns an existence probability to every landmark candidate. Finally, we only select landmarks that have a high existence probability to build our map. In total, the digital map consists of 3860 landmarks and has a size of only 600 kB. Furthermore, the landmarks are stored in the Universal Transverse Mercator (UTM) coordinate system, while the measurements are stored in the vehicle frame that has its origin at the center of the rear axle. An in-depth explanation of the map creation process and the pre-processing pipeline can be found in [39], [12]. Additionally, we use the Kitti odometry dataset [40] to evaluate our approach on a public dataset. Since our approach relies on a single feature for each object, we only use the provided camera images and extract landmarks for the digital map and measurements using the MSER algorithm as described above.

V. TRAINING SIMULATION FRAMEWORK

Besides our approach, we present a training simulation framework that allows us to synthetically generate training data without the need of recording real world measurements. Ideally, our landmark-based localization method can be deployed in a variety of environments and scenarios. This requires a diverse and extensive training dataset that enables the network to generalize to unseen data points. However, creating and maintaining a large-scale dataset in multiple

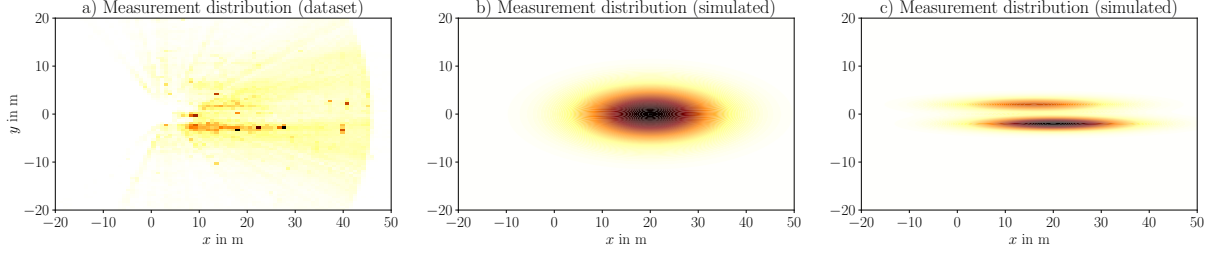


Fig. 4. Measurement distribution from our dataset (left) and two possible approaches for the simulated measurement model using 2D Gaussian distributions (center and right). Darker patches indicate areas where measurements appear more frequently.

urban and rural areas is both expensive and labor intensive. Instead, we propose a training simulation framework, which is based on real-world data. Since static objects are usually located at the side of the road, e.g. traffic signs, poles and trees, the landmarks are distributed mostly along the longitudinal axis, as we show in Fig. 4 a), where darker areas indicate a more frequent occurrence of landmarks in the vehicle’s field of view. Based on the measurement distribution of our dataset, we model the location using a multivariate normal distribution with mean $\mu \in \mathbb{R}^2$ and covariance matrix $\Sigma \in \mathbb{R}^{2 \times 2}$ given by:

$$\mathcal{N}(\mu, \Sigma) = \frac{1}{\sqrt{(2\pi)^2 |\Sigma|}} \exp \left(-\frac{1}{2} (x - \mu)^T \Sigma^{-1} (x - \mu) \right). \quad (13)$$

In Fig. 4 b) and c), we demonstrate two possibilities of modelling the measurement distribution. In b) we set $\mu = [20 \ 0]$ and $\Sigma = \text{diag}(100, 15)$. To better incorporate the fact that the measurements occur on the side of the road, we use a Gaussian mixture $\mathcal{N}(\mu, \Sigma) = \lambda_1 \mathcal{N}(\mu_1, \Sigma_1) + \lambda_2 \mathcal{N}(\mu_2, \Sigma_2)$, with $\mu_1 = [20 \ -2]$, $\mu_2 = [20 \ 2]$, $\Sigma_1 = \Sigma_2 = \text{diag}(120, 1)$ and $\lambda_2 = 0.6$. The resulting Gaussian is visualized in Fig. 4 c). Each training step, we randomly draw ν measurements from the Gaussian $M \sim \mathcal{N}(\mu, \Sigma)$, where again ν is a random variable $\nu \sim \mathcal{U}(\nu_{\min}, \nu_{\max})$. Considering ideal sensors that detect each landmark in the vehicle’s field of view, the set of measurements and landmarks are equal $M = L$, thus we duplicate all our synthetically sampled measurements. However, we identified three effects that deteriorate the quality of the input data: 1) *Clutter measurements* did not originate from a landmark either due to dynamic objects or a faulty sensor. 2) *Missed detections* are landmarks that were not seen by any sensor because the landmark was obscured or, again, caused by a faulty sensor. And finally, 3) *measurement noise* that causes an inaccurate landmark localization. We model these effects with a Poisson distribution $P(k; \lambda) = (\lambda^k \exp(-\lambda)) / (k!)$, which simulates the number of events with the mean occurrence rate λ . In our case, we add clutter measurements with a mean clutter rate of λ_{clutter} and we delete measurements with mean miss rate λ_{miss} . Finally, we add noise to the remaining measurements by sampling from a uniform distribution $\mathcal{U}(-\sigma_{\text{noise}}, \sigma_{\text{noise}})$.

To summarize, we first generate ν measurements from the spatial distribution, visualized in Fig. 4. These measurements are duplicated and then used as landmarks. To simulate

real-world scenarios and environments, we deteriorate the measurements by applying multiple effects, such as clutter, missed detections and measurement noise. The modified measurements and landmarks form the input to our network, following the training process from Section III-C.

VI. EXPERIMENTS AND EVALUATION

Here, we present the experiments and the evaluation results. In particular, we perform real-world experiments on our Ulm-Lehr datasets as well as the adapted version of the Kitti odometry dataset [40]. Then, we evaluate our network architecture using the simulation framework as introduced in Section V and compare the localization accuracy by changing the percentage of simulated and real-world training data. Our network is implemented using *PyTorch* [41] and we perform all experiments on a *Nvidia Geforce 2080Ti*. Furthermore, we set $k = 8$ as we found it to capture all landmarks that are in the vicinity of each measurement.

A. Real-World Experiments

The results of our real-world experiments are shown in Table I, where we report the root mean square error (RMSE). First, we compare the localization accuracy of the GPS-based inference with our prior work [12] for different GPS noise parameters, which are denoted by σ_x, σ_y and σ_φ in Table I a) - c). Our approach shows impressive improvements in every experiment we conducted both on our own and the Kitti odometry dataset. In some cases, we improve the localization accuracy up to 54 %. Furthermore, DeepLocalization [12] has about 1.8 M learnable parameters with an inference time of about 2 ms. Due to the attention mechanism, the complexity of our network increases to 8.4 M learnable parameters with an inference time of 26 ms, which still meets our real-time requirements.

Next, we compare the filter-based inference with two baseline approaches (ICP [9] and an EKF), as well as related state-of-the-art model-based localization methods in Table I d) - e). Here, it is important to note that we modified the FastSLAM [10] and the PHDSlam [42] to only include the localization algorithm, as the digital map is already provided by our dataset. Similar to the GPS-based inference, we achieve the best localization accuracy by a wide margin on the Ulm-Lehr dataset as well as the Kitti odometry dataset compared to the related work with a mean accuracy of about 0.16 m - 0.18 m for the position and 1.4° for the orientation.

TABLE I
RESULTS OF REAL-WORLD EXPERIMENTS.

| Method | Ulm-Lehr | | | Kitti Odometry [40] | | |
|---|---------------|---------------|-------------|---------------------|---------------|-------------|
| | x | y | φ | x | y | φ |
| a) GPS-based Inference: $\sigma_x, \sigma_y = 2$ m, $\sigma_\varphi = 10^\circ$ | | | | | | |
| DeepLoc [12] | 0.77 m | 0.70 m | 2.5° | 0.84 m | 0.82 m | 3.1° |
| Attn-based (ours) | 0.41 m | 0.51 m | 1.7° | 0.49 m | 0.55 m | 2.1° |
| b) GPS-based Inference: $\sigma_x, \sigma_y = 1$ m, $\sigma_\varphi = 4^\circ$ | | | | | | |
| DeepLoc | 0.44 m | 0.37 m | 1.3° | 0.48 m | 0.44 m | 2.3° |
| Attn-based (ours) | 0.20 m | 0.23 m | 0.9° | 0.29 m | 0.32 m | 1.7° |
| c) GPS-based Inference: $\sigma_x, \sigma_y = 0.5$ m, $\sigma_\varphi = 2^\circ$ | | | | | | |
| DeepLoc | 0.27 m | 0.23 m | 0.8° | 0.35 m | 0.33 m | 1.4° |
| Attn-based (ours) | 0.17 m | 0.18 m | 0.6° | 0.21 m | 0.24 m | 1.1° |
| d) Related approaches | | | | | | |
| ICP [9] | 1.17 m | 1.46 m | 4.9° | 1.91 m | 1.84 m | 6.1° |
| EKF + GPS | 0.59 m | 0.54 m | 6.5° | 1.07 m | 1.12 m | 6.5° |
| FastSlam [10] | 0.34 m | 0.32 m | 2.1° | 0.73 m | 0.77 m | 2.9° |
| PHDSlam [42] | 0.30 m | 0.32 m | 1.7° | — | — | — |
| RFS-MCL [19] | 0.28 m | 0.26 m | 1.9° | — | — | — |
| e) Filter-based Inference | | | | | | |
| DeepLoc + EKF | 0.27 m | 0.24 m | 0.8° | 0.45 m | 0.44 m | 2.1° |
| Attn-based + EKF (ours) | 0.18 m | 0.16 m | 0.6° | 0.31 m | 0.25 m | 1.4° |

Additionally, we evaluate the GPS-based inference with noise parameters $\sigma_x, \sigma_y = 1$ m, $\sigma_\varphi = 4^\circ$ on our alternative train/test split, visualized as ① / ② in Fig. 3, to show the generalization capabilities of our approach. With the uniform split our network achieves a localization accuracy of 0.20 m to 0.23 m and an orientation accuracy of 1.7°. When we train the network with the spatially divided dataset ① / ②, the localization accuracy only drops to 0.29 m and 0.33 m for the x and y component, respectively. We achieve an orientation accuracy of about 1.9°. These results show, that even though the network has never been trained on the area marked as ②, our approach is able to localize in an unknown environment and achieve an accuracy that meets the requirement for urban scenarios. Finally, the localization accuracy of an exemplary 2 min drive on our test track is shown in Fig. 5. We additionally highlight the RMSE as well as the maximum error for each pose component.

B. Simulated Experiments

The results of our simulated experiments are shown in Table II. Here, we report the RMSE for the GPS-based inference with noise parameters $\sigma_x, \sigma_y = 1$ m, $\sigma_\varphi = 4^\circ$ in Table II a), as well as the filter-based inference method in Table II b). For each experiment, we train the network with the synthetic measurements and landmarks, as explained in Section V, and perform the inference on our Ulm-Lehr dataset. For this experiment, we rely on the Gaussian mixture distribution shown in Fig. 4 c). Furthermore, we also add a small percentage of the real-world samples from our dataset to the training pipeline, indicated as 0%, 5% and 50%. Thus,

TABLE II
RESULTS OF SIMULATED EXPERIMENTS.

| Method | x | y | φ |
|---|--------|--------|-----------|
| a) GPS-based Inference: $\sigma_x, \sigma_y = 1$ m, $\sigma_\varphi = 4^\circ$ | | | |
| Attn-based 0% | 0.40 m | 0.45 m | 1.6° |
| Attn-based 5% | 0.37 m | 0.41 m | 1.5° |
| Attn-based 50% | 0.31 m | 0.34 m | 1.1° |
| b) Filter-based Inference: | | | |
| Attn-based 0% + EKF | 0.29 m | 0.33 m | 1.5° |
| Attn-based 5% + EKF | 0.29 m | 0.31 m | 1.5° |
| Attn-based 50% + EKF | 0.23 m | 0.25 m | 0.9° |

we demonstrate that a satisfying localization accuracy can be achieved with very few or even no real-world data at all, which still meets our required accuracy in urban scenarios. Especially when the EKF with the CTRV motion model is employed, the negative impact of the synthetic measurement is negligible, thus indicating promising potential for our method to be deployed in a variety of different environments and scenarios around the world.

VII. CONCLUSION

We presented a localization approach based on current measurements and generic landmarks from a HD digital map. Our attention mechanism learns the landmark to measurement association, which we use to register the input point clouds (measurements and landmarks) and, subsequently, infer the vehicle's pose. Furthermore, we proposed a training framework to generate synthetic training data. It facilitates the learning process of our method and improves generalization. Finally, we evaluate our approach on two datasets and show promising results compared to the related work.

REFERENCES

- [1] S. Thrun, "Probabilistic robotics," *Communications of the ACM*, vol. 45, no. 3, pp. 52–57, 2002.
- [2] J. Wiederer, A. Bouazizi, U. Kressel, and V. Belagiannis, "Traffic control gesture recognition for autonomous vehicles," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 10 676–10 683.
- [3] I. Hasan, F. Setti, T. Tsesmelis, V. Belagiannis, S. Amin, A. Del Bue, M. Cristani, and F. Galasso, "Forecasting people trajectories and head poses by jointly reasoning on tracklets and vislets," *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 4, pp. 1267–1278, 2019.
- [4] N. Engel, S. Hoermann, P. Henzler, and K. Dietmayer, "Deep object tracking on dynamic occupancy grid maps using rnn," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018, pp. 3852–3858.
- [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] F. Gies, J. Posselt, M. Buchholz, and K. Dietmayer, "Extended existence probability using digital maps for object verification," in *2020 IEEE 23rd International Conference on Information Fusion (FUSION)*, 2020, pp. 1–7.
- [7] J. Levinson, M. Montemerlo, and S. Thrun, "Map-based precision vehicle localization in urban environments," in *Robotics: science and systems*, vol. 4, no. Citeseer. Citeseer, 2007, p. 1.

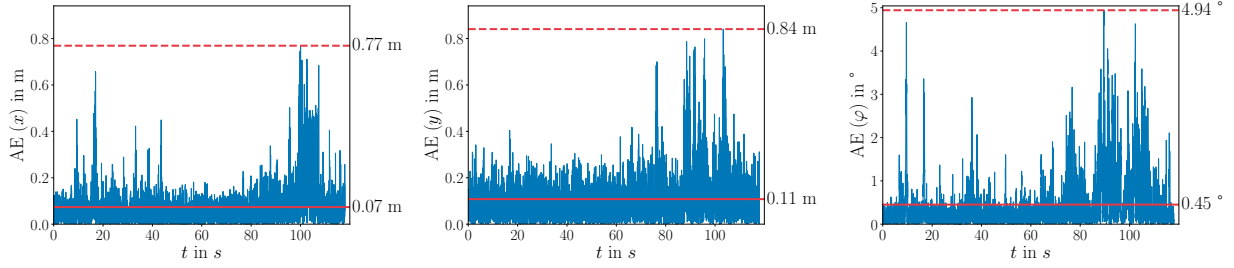


Fig. 5. Here, we report the absolute localization errors (AE) on our test track in Ulm-Lehr, the RMSE (—) and the maximum error (---) for the filter-based inference.

- [8] M. G. Wing, A. Eklund, and L. D. Kellogg, "Consumer-grade global positioning system (gps) accuracy and reliability," *Journal of forestry*, vol. 103, no. 4, pp. 169–173, 2005.
- [9] P. J. Besl and N. D. McKay, "Method for registration of 3-d shapes," in *Sensor fusion IV: control paradigms and data structures*, vol. 1611. International Society for Optics and Photonics, 1992, pp. 586–606.
- [10] M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit, *et al.*, "Fastslam: A factored solution to the simultaneous localization and mapping problem," *Aaai/iaai*, vol. 593598, 2002.
- [11] C. Chen, B. Wang, C. X. Lu, N. Trigoni, and A. Markham, "A survey on deep learning for localization and mapping: Towards the age of spatial machine intelligence," *arXiv preprint arXiv:2006.12567*, 2020.
- [12] N. Engel, S. Hoermann, M. Horn, V. Belagiannis, and K. Dietmayer, "Deeplocalization: Landmark-based self-localization with deep neural networks," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2019, pp. 926–933.
- [13] N. Radwan, A. Valada, and W. Burgard, "Vlocnet++: Deep multitask learning for semantic visual localization and odometry," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 4407–4414, 2018.
- [14] A. Censi, "An icp variant using a point-to-line metric," in *2008 IEEE International Conference on Robotics and Automation*. Ieee, 2008, pp. 19–25.
- [15] D. Fontanelli, L. Ricciato, and S. Soatto, "A fast ransac-based registration algorithm for accurate localization in unknown environments using lidar measurements," in *2007 IEEE International Conference on Automation Science and Engineering*. IEEE, 2007, pp. 597–602.
- [16] L. Teslić, I. Škrjanc, and G. Klančar, "EKF-based localization of a wheeled mobile robot in structured environments," *Journal of Intelligent & Robotic Systems*, vol. 62, no. 2, pp. 187–203, 2011.
- [17] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, "Monte carlo localization for mobile robots," in *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)*, vol. 2. IEEE, 1999, pp. 1322–1328.
- [18] M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit, *et al.*, "Fastslam 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges," in *IJCAI*, vol. 3, 2003, pp. 1151–1156.
- [19] M. Stübler, J. Wiest, and K. Dietmayer, "Feature-based mapping and self-localization for road vehicles using a single grayscale camera," in *2015 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2015, pp. 267–272.
- [20] J. Mullane, B.-N. Vo, M. D. Adams, and B.-T. Vo, "A random-finite-set approach to bayesian slam," *IEEE transactions on robotics*, vol. 27, no. 2, pp. 268–282, 2011.
- [21] H. Deusch, S. Reuter, and K. Dietmayer, "The labeled multi-bernoulli slam filter," *IEEE Signal Processing Letters*, vol. 22, no. 10, pp. 1561–1565, 2015.
- [22] R. Mur-Artal and J. D. Tardós, "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.
- [23] A. Pumarola, A. Vakhtov, A. Agudo, A. Sanfeliu, and F. Moreno-Noguer, "Pl-slam: Real-time monocular visual slam with points and lines," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 4503–4508.
- [24] S. A. Mohamed, M.-H. Hagbayan, T. Westerlund, J. Heikkonen, H. Tenhunen, and J. Plosila, "A survey on odometry for autonomous navigation systems," *IEEE Access*, vol. 7, pp. 97 466–97 486, 2019.
- [25] L. Yang, Z. Bai, C. Tang, H. Li, Y. Furukawa, and P. Tan, "Sanet: Scene agnostic network for camera localization," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 42–51.
- [26] A. Kendall, M. Grimes, and R. Cipolla, "Posenet: A convolutional network for real-time 6-dof camera relocalization," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2938–2946.
- [27] W. Lu, G. Wan, Y. Zhou, X. Fu, P. Yuan, and S. Song, "Deepicp: An end-to-end deep neural network for 3d point cloud registration," *arXiv preprint arXiv:1905.04153*, 2019.
- [28] B. Wang, C. Chen, C. X. Lu, P. Zhao, N. Trigoni, and A. Markham, "Atloc: Attention guided camera localization," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 06, 2020, pp. 10 393–10 401.
- [29] W. Lu, Y. Zhou, G. Wan, S. Hou, and S. Song, "L3-net: Towards learning based lidar localization for autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 6389–6398.
- [30] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," *Advances in neural information processing systems*, vol. 30, pp. 5099–5108, 2017.
- [31] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *arXiv preprint arXiv:1706.03762*, 2017.
- [32] N. Engel, V. Belagiannis, and K. Dietmayer, "Point transformer," *arXiv preprint arXiv:2011.00931*, 2020.
- [33] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.
- [34] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 652–660.
- [35] V. Belagiannis, C. Rupprecht, G. Carneiro, and N. Navab, "Robust optimization for deep regression," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2830–2838.
- [36] A. Kendall, Y. Gal, and R. Cipolla, "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7482–7491.
- [37] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Kdd*, vol. 96, no. 34, 1996, pp. 226–231.
- [38] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust wide-baseline stereo from maximally stable extremal regions," *Image and vision computing*, vol. 22, no. 10, pp. 761–767, 2004.
- [39] M. Stübler, S. Reuter, and K. Dietmayer, "A continuously learning feature-based map using a bernoulli filtering approach," in *2017 Sensor Data Fusion: Trends, Solutions, Applications (SDF)*. IEEE, 2017, pp. 1–6.
- [40] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.
- [41] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *arXiv preprint arXiv:1912.01703*, 2019.
- [42] J. Mullane, B.-N. Vo, and M. D. Adams, "Rao-blackwellised phd slam," in *2010 IEEE International Conference on Robotics and Automation*. IEEE, 2010, pp. 5410–5416.