Mixture Density Networks for Multipath Assisted Positioning-based Fingerprinting

Markus Ulmschneider, Christian Gentner and Armin Dammann
German Aerospace Center (DLR), Institute of Communications and Navigation
{markus.ulmschneider,christian.gentner,armin.dammann}@dlr.de

***Abstract*—In multipath assisted positioning schemes, the spatial information contained in multipath propagation of wireless radio systems is exploited for localization of a receiver. However, such schemes suffer from a high computational complexity. We have proposed before a fingerprinting localization system based on multipath assisted positioning, where the fingerprinting database is encoded in a deep neural network (DNN). Within this paper, we propose and evaluate a mixture density network approach in our DNN to analyze ambiguities among fingerprints at different locations. We show that our scheme shows a very good positioning performance with an error of around 2m for the most part, while having a low computational complexity in the online stage and a very low effort compared to traditional fingerprinting schemes.**

***Index Terms*—cooperative Channel-SLAM, deep learning, fingerprinting, localization, simultaneous localization and mapping**

I. Introduction

A promising alternative to global navigation satellite systems (GNSSs) for indoor localization are terrestrial radio frequency (RF) systems such as cellular or wireless local area network (WLAN) systems. While such systems are typically deployed for communication, they can be used for localization as well. Nevertheless, multipath propagation tends to deteriorate the localization performances.

With multipath assisted positioning, the spatial information contained in multipath components (MPCs) is exploited by regarding each MPC as a line-of-sight (LoS) signal from a so-called *virtual* transmitter. The locations of the virtual transmitters are typically unknown, but can be estimated jointly with the receiver position with simultaneous localization and mapping (SLAM) [1]–[3]. Physical transmitters can be WLAN routers or cellular base stations. With cooperative Channel-SLAM [4], [5], we have introduced such an approach, where multiple users estimate their own position jointly with the locations of physical and virtual transmitters in the scenario. While cooperative Channel-SLAM is a promising approach with good positioning performance, it suffers from a very high computational complexity, making a practical implementation infeasible. In the following, the term transmitter includes both physical and virtual transmitters. The term user refers to either a mobile user or the radio receiver the user is equipped with, depending on the context. We assume the physical transmitter and the receiver to be perfectly time-synchronized.

A different approach to positioning are fingerprinting schemes [6]. In contrast to model-based positioning schemes, fingerprinting schemes are data-driven and work in two stages. In the offline stage, fingerprints are collected at known locations and stored in a database with the respective locations. In the online stage, users can be localized by matching measured fingerprints against the database. Fingerprints can be channel features, magnetic signatures or visual features, for example.

A big drawback of fingerprinting schemes is the huge effort in the offline stage. A third-party positioning system is typically necessary for obtaining the locations of the fingerprints. When the environment changes, the database needs to be updated and the third-party positioning system deployed again.

We have proposed a fingerprinting scheme named DNN-CC-SLAM in [7], where the fingerprints are times of arrival (ToAs) of received signal components, including the LoS path and MPCs. The locations of the collected ToAs are estimated with cooperative Channel-SLAM in the offline stage without the need of a third-party positioning system. A deep neural network (DNN) is trained in the offline phase to estimate the user position and covariance based on the ToAs of signal components. A Kalman filter tracks the user position based on these estimates from the DNN in the online stage.

Within this paper, we implement this DNN as a mixture density network (MDN) [8] to gain insight into the ambiguities among fingerprints at different user locations.

The remainder of this paper is organized as follows. In Section II, we introduce cooperative Channel-SLAM and fingerprinting. Section III presents DNN-CC-SLAM and our MDN architecture. The user tracking is explained in Section IV. We perform simulations to evaluate our scheme in Section V, and conclude the paper in Section VI.

II. Cooperative Channel-SLAM and Fingerprinting

A. Cooperative Channel-SLAM

Fig. 1 illustrates the principal idea behind multipath assisted positioning in a simplified scenario with one reflecting wall and one point scatterer. Omitting the LoS path from the physical transmitter Tx to the two user locations for clarity, there are two possible propagation paths shown. In the case of a reflection of the signal at the wall, the virtual transmitter vTx1 is at the location of the physical transmitter mirrored at the wall. The corresponding propagation paths are drawn blue. The case of scattering at the point scatterer is illustrated by the red propagation path. The corresponding virtual transmitter vTx2 is located at the point scatterer. In the case of scattering,

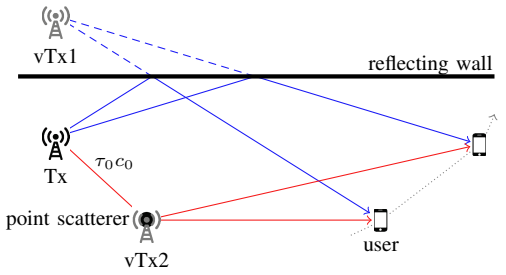

Fig. 1. The reflected and scattered signal components are regarded as LoS signals from virtual transmitters.

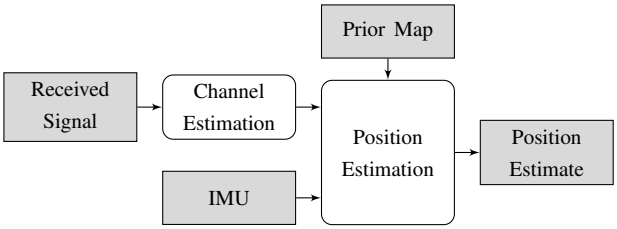

Fig. 2. The two stages of Channel-SLAM are channel and position estimation.

there is a delay offset τ_0 between the physical and the virtual transmitter, corresponding to the Euclidean distance $\tau_0 c_0$ between the two, where c_0 is the speed of light. This delay offset can be interpreted as clock offset.

The cases of single reflections and scattering can be extended to multiple interactions of the transmit signal with objects in the environment in a straightforward manner [1].

Note that the locations of the virtual transmitters are independent from the user location. With Channel-SLAM, we jointly estimate the user location and velocity with the positions and clock offsets of the transmitters. We do not distinguish between physical and virtual transmitters, i.e., between the LoS path and MPCs. Each signal component corresponds to one transmitter, and the physical and virtual transmitters are covered by the same model. Hence, localization of the user is possible with a single physical transmitter. Channel-SLAM works in two steps as indicated by Fig. 2.

In the first step, a channel estimator tracks the parameters of signal components over time. Within this paper, we use the Kalman Enhanced Super Resolution Tracking (KEST) [9] algorithm. At each time instant k, the parameter estimates are stored in the vector $\boldsymbol{z}_k$. Tracking the parameters inherently yields a data association among signal components, and thus transmitters, for neighboring time instants.

In the second step, the estimates from the channel estimator are used as measurements in a recursive Bayesian estimation scheme to jointly estimate the locations and clock offsets of the transmitters and the user position and velocity with SLAM. We define the joint user and transmitter states from time instants zero to k by $\boldsymbol{x}_{0:k}$ and seek the corresponding minimum mean square error (MMSE) estimator $\hat{\boldsymbol{x}}_{0:k,\mathrm{MMSE}}$. If information on the transmitter states from a prior map is available, this information can be incorporated as well.

Due to the high non-linearity of the estimation problem, the second step of Channel-SLAM is implemented as a particle filter, which is a non-optimal solution to recursive Bayesian estimation. The idea is to represent the involved probability density functions (PDFs) with weighted samples, so-called particles, in the state space. To reduce the very high complexity of the estimation problem, we split the overall state $\boldsymbol{x}_{0:k}$ into the user state $\boldsymbol{x}_{\mathrm{u},0:k}$ and the transmitter state $\boldsymbol{x}_{\mathrm{TX},0:k}$, each of them from time instants zero to k,

$$\begin{aligned} \mathrm{p}\left(\boldsymbol{x}_{0:k}|\boldsymbol{z}_{1:k},\boldsymbol{u}_{1:k}\right) = \mathrm{p}\left(\boldsymbol{x}_{\mathrm{u},0:k}|\boldsymbol{z}_{1:k},\boldsymbol{u}_{1:k}\right) \\ \times \mathrm{p}\left(\boldsymbol{x}_{\mathrm{TX},0:k}|\boldsymbol{z}_{1:k},\boldsymbol{x}_{\mathrm{u},0:k}\right), \end{aligned} \tag{1}$$

where $\boldsymbol{z}_{1:k}$ are measurements obtained from KEST from time instants one to k. Additional sensors such as an inertial measurement unit (IMU) can be incorporated in the second step as control input $\boldsymbol{u}_{1:k}$ as indicated by Fig. 2.

While an extension to multiple physical transmitters is straightforward, in particular if their transmit signals are separable in one domain such as time, frequency or code, we restrict ourselves to the case of one physical transmitter within this paper. Furthermore, switching roles to a transmitting mobile user and static physical receivers is straightforward.

Note that we model point scatterers such that they distribute the energy of an impinging signal perfectly uniformly to all directions. Nevertheless, other effects such as diffraction or imperfect scatterers can typically be well modeled as point scatterers in Channel-SLAM.

In [5], we have presented cooperative Channel-SLAM, where users jointly estimate the states of the transmitters and therefore improve their positioning performance drastically.

B. *Fingerprinting for Localization*

In contrast to model-based approaches such as Channel-SLAM, fingerprinting approaches for localization are based on collected data. They work in two stages.

In the offline stage, fingerprints are taken at known locations and stored in a database together with these locations. Fingerprints can be any location-specific features, and are typically features of the wireless radio channel, such as channel state information (CSI) or received signal strength indicator (RSSI). In the online stage, a user can be localized by matching a measured fingerprint against the database.

One advantage of fingerprinting is that no complex models are needed. Accordingly, it tends to be of very low complexity and often shows good positioning performances. In addition, it naturally works with different radio technologies, such as WLAN, ultra-wideband (UWB) or Bluetooth. However, fingerprinting schemes come with major drawbacks. First, a third party positioning system typically needs to be deployed in the offline stage to obtain the locations where fingerprints have been collected. Second, the fingerprints need to be collected, which takes some effort before localization is possible. Third, since no models are assumed, the fingerprints need to be collected in a dense grid for a good localization performance. Fourth, an appropriate or even optimum metric for the similarity of two fingerprints may be very hard to find.

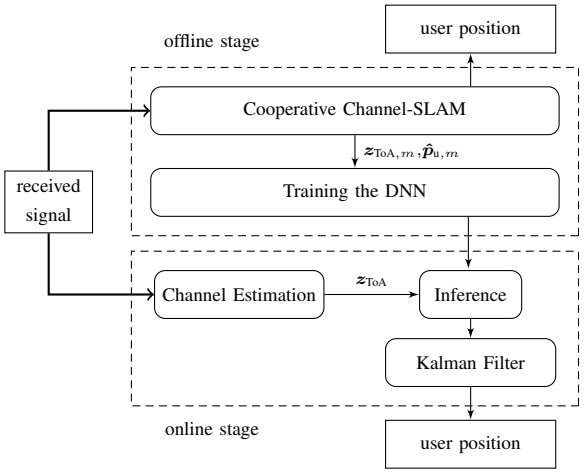

Fig. 3. The flow chart of DNN-CC-SLAM.

These challenges hinder a wide-spread use of fingerprinting methods in large-scale systems. Nevertheless, more recent publications on fingerprinting try to deal with these challenges with various methods using machine learning [10], [11]. For example, encoding the fingerprinting database in a DNN can solve the problem of finding a metric for the similarity of fingerprints. In addition, a DNN is able to generalize to some extent, leading to a better interpolation and extrapolation.

III. Multipath Assisted Positioning-based Fingerprinting

In [7], we have proposed a new fingerprinting scheme named DNN-CC-SLAM that can overcome some disadvantages of classical fingerprinting approaches and the high complexity of cooperative Channel-SLAM. In contrast to classical fingerprinting, the locations of the taken fingerprints in the offline stage are obtained with cooperative Channel-SLAM. A flowchart of DNN-CC-SLAM is presented in Fig. 3.

In the offline stage, one or more users roam through a scenario localizing themselves with cooperative Channel-SLAM. The estimated user locations $\hat{\boldsymbol{p}}_{\text{u},m}$ and ToAs $\boldsymbol{z}_{\text{ToA},m}$ of signal components from the m^{th} user are used to train the feedforward DNN to predict the user location based on the fingerprints, i.e., the ToAs of signal components.

In the online stage, a channel estimator processes the received signal and passes the estimated ToAs $\boldsymbol{z}_{\text{ToA}}$ to the DNN for inference. Since a single antenna is assumed at the user and no tracking, but only a point estimation of the ToAs is required, a very efficient channel estimator can be used. Within the scope of this paper, we use the superfast line spectral estimation (SFLSE) [12]. A Kalman filter then tracks the user position and velocity.

Compared to classical fingerprinting, a huge advantage is that no third party positioning system needs to be deployed, neither for creating the database in the first place, not for re-collecting fingerprints in case of changes in the environment. In addition, fingerprint collection can be easily automated, for example by equipping a mobile robot with a RF receiver. While the offline stage of DNN-CC-SLAM, containing cooperative Channel-SLAM and training the DNN, is computationally complex, the necessary operations can be performed offline on a server or in the cloud. In the online stage, the complexity is very low. First, an efficient channel estimator can be used. Second, inference of the DNN is of low complexity. Third, the Kalman filter implementation is very efficient, since each measurement is a position, which is part of the state space. Encoding the fingerprinting database in a DNN does not only improve interpolation and extrapolation, but also avoids the difficulty of finding an appropriate metric between two fingerprints, i.e., two sets of ToAs.

We train the DNN to learn the user position from fingerprints, which are a vector of ToAs of signal components. We denote the input vector corresponding to the fingerprints by $\boldsymbol{q}$ and the vector corresponding to the user position by $\boldsymbol{p}$. In particular, we seek to learn the underlying distribution $\mathrm{p}\left(\boldsymbol{p}|\boldsymbol{q}\right)$ based on a set of training data, i.e., samples $\boldsymbol{p}_j$ and $\boldsymbol{q}_j$.

Fingerprinting relies on the assumptions that (i) two fingerprints taken at two user positions close to each other are close in some dedicated metric, and that (ii) fingerprints taken at two locations far away from each other are far away in the same metric. However, these assumptions are sometimes not satisfactorily fulfilled due to symmetries in the scenario as well as noise and its influence on channel estimates.

In order to deal with corresponding ambiguities, we implement a MDN following [8], where $\mathrm{p}\left(\boldsymbol{p}|\boldsymbol{q}\right)$ is modeled as a Gaussian mixture model (GMM) consisting of $N_{\mathcal{N}}$ normally distributed components with normalized weights α_i,

$$\mathrm{p}\left(\boldsymbol{p}|\boldsymbol{q}\right) = \sum_{i=1}^{N_{\mathcal{N}}} \alpha_i(\boldsymbol{q}) \mathcal{N}(\boldsymbol{p}; \boldsymbol{\mu}_i(\boldsymbol{q}), \boldsymbol{C}_i(\boldsymbol{q}))\,, \tag{2}$$

where $\mathcal{N}(\boldsymbol{p}; \boldsymbol{\mu}_i(\boldsymbol{q}), \boldsymbol{C}_i(\boldsymbol{q}))$ is the PDF of a normal distribution in $\boldsymbol{p}$ with mean $\boldsymbol{\mu}_i(\boldsymbol{q})$ and covariance matrix $\boldsymbol{C}_i(\boldsymbol{q})$, which explicitly depend on $\boldsymbol{q}$.

The loss function in the DNN training is the negative log-likelihood function regarding the PDF in Eq. (2). We write the loss function $L\left(\boldsymbol{w}\right)$ explicitly as a function of the DNN parameters $\boldsymbol{w}$, i.e., weights and biases,

$$L\left(\boldsymbol{w}\right) = -\sum_{j=1}^{N_S} \log \sum_{i=1}^{N_{\mathcal{N}}} \alpha_i(\boldsymbol{q}_j, \boldsymbol{w}) \mathcal{N}(\boldsymbol{p}_j; \boldsymbol{\mu}_i(\boldsymbol{q}_j, \boldsymbol{w}), \boldsymbol{C}_i(\boldsymbol{q}_j, \boldsymbol{w})), \tag{3}$$

where N_S is the number of samples available for training.

The output layer of the network consists of $6N_{\mathcal{N}}$ neurons, corresponding to a weight, a two-dimensional mean, and three entries of the 2×2 symmetric covariance matrix for each component in a GMM. If $N_{\mathcal{N}} = 1$, the weight α_i and the index i can be dropped.

IV. User Tracking

In the online stage of DNN, the estimated ToAs of signal components are passed to the DNN for inference, and the resulting estimates to the Kalman filter for tracking of the

user. The state vector in the Kalman filter at time instant k is given by

$$\boldsymbol{x}_k = \left[\boldsymbol{p}_k{}^T \;\; \boldsymbol{v}_k{}^T\right]^T, \tag{4}$$

where $\boldsymbol{p}_k$ and $\boldsymbol{v}_k$ are the position and velocity of the user. The movement model is a linear, pedestrian random walk model.

The output of the GMM at time instant k are the parameters $\boldsymbol{\mu}_{i,k}$, a vector of length two, $\boldsymbol{C}_{i,k}$, a matrix of size 2×2, and $\alpha_{i,k}$ for each of the $N_{\mathcal{N}}$ components. For initialization of the Kalman filter, the initial position $\hat{\boldsymbol{p}}_0$ and position covariance matrix $\hat{\boldsymbol{L}}_0$ are chosen to be from the Gaussian component with highest of the weights $\alpha_{i,0}$ from the DNN. The full initial state estimate mean and the corresponding covariance matrix at time instant 0 are then given by

$$\hat{\boldsymbol{x}}_0 = \left[\hat{\boldsymbol{p}}_0^T \;\; \boldsymbol{0}^T\right]^T \quad \text{and} \quad \hat{\boldsymbol{P}}_0 = \begin{bmatrix} \hat{\boldsymbol{L}}_0 & \mathbb{0} \\ \mathbb{0} & \sigma_{v,0}\mathbb{1} \end{bmatrix}, \tag{5}$$

respectively, where $\sigma_{v,0}$ denotes the initial variance on the velocity, $\mathbb{0}$ is the all-zero matrix and $\mathbb{1}$ is the identity matrix. The posterior state estimate mean at time instant k is denoted by $\hat{\boldsymbol{x}}_k$ and the corresopnding covariance matrix by $\hat{\boldsymbol{P}}_k$.

In the Kalman filter, we interpret the output of the DNN as a measurement of the user position. In particular, the position measurements in the filter are the mean $\boldsymbol{\mu}_{i,k}$ and the measurement covariance the corresponding covariance matrix $\boldsymbol{C}_{i,k}$ for some index i. Hence, no complicated measurement model needs to be assumed, but the observation model corresponds to an identity matrix of corresponding dimensions. The innovation covariance in the Kalman filter is a 2×2 matrix, who's inverse can be obtained efficiently in closed form.

If $N_{\mathcal{N}} = 1$, the output of the network is directly passed on to the Kalman filter as measurement. If $N_{\mathcal{N}} > 1$, we obtain from the DNN multiple weighted hypotheses for measurements of the user position. For the sake of a low computational complexity, we take a hard-decision in favor of the hypothesis with lowest distance towards the current user position in the Kalman filter. In particular, at time instant k, we decide for the Gaussian component with index $\hat{i}_k$, for which the Kullback–Leibler divergence (KLD) between the corresponding Gaussian component $\mathcal{N}(\boldsymbol{\mu}_{i,k}, \boldsymbol{C}_{i,k})$ and the current user position PDF in the Kalman filter $\mathcal{N}\left(\hat{\boldsymbol{p}}_k, \hat{\boldsymbol{L}}_k\right)$ with mean $\hat{\boldsymbol{p}}_k$ and covariance matrix $\hat{\boldsymbol{L}}_k$ is the lowest. Note that in line with Eqs. (5), $\hat{\boldsymbol{p}}_k$ corresponds to the first two entries of $\hat{\boldsymbol{x}}_k$ and $\hat{\boldsymbol{L}}_k$ is the upper left 2×2 submatrix of $\hat{\boldsymbol{P}}_k$, i.e., the matrix containing the first two rows and columns of $\hat{\boldsymbol{P}}_k$. In addition, we weight this KLD by the respective inverse of the weight $\alpha_{i,k}$ of the Gaussian component, and thus decide for the Gaussian component with index

$$\hat{i}_k = \underset{i \in \{1,\ldots,N_{\mathcal{N}}\}}{\arg\min} \frac{1}{\alpha_{i,k}} \mathrm{KLD}\left(\mathcal{N}(\boldsymbol{\mu}_{i,k}, \boldsymbol{C}_{i,k}), \mathcal{N}\left(\hat{\boldsymbol{p}}_k, \hat{\boldsymbol{L}}_k\right)\right). \tag{6}$$

V. Simulations

A. Simulation Environment and Parameters

We evaluate our approach with simulations in the scenario in Fig. 4, showing the top view of an indoor mall with one physical transmitter depicted by the red triangle labeled Tx. The black lines and points represent walls and scatterers, respectively. We refer to the user track depicted by the blue line running from START to END as reference track.

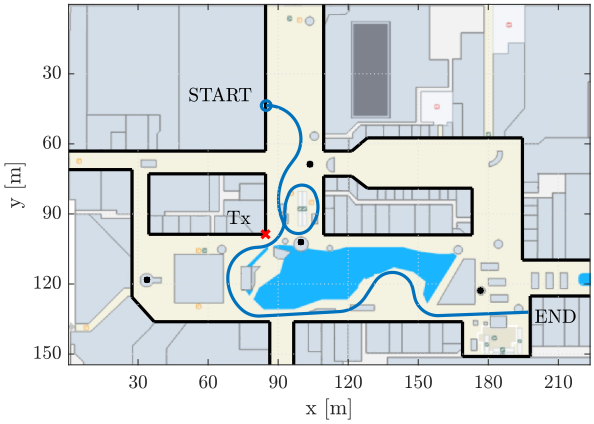

Fig. 4. Top view of the simulation scenario.

The physical transmitter continuously transmits a signal which is known to the users. The center frequency of the signal is 1.9 GHz and its bandwidth 100 MHz. Based on the walls and scatterers, a channel impulse response (CIR) is simulated with ray-tracing for every user position. In Channel-SLAM, the users record a snapshot of the received signal every 10 ms as input for the channel estimator to obtain a position estimate.

In Channel-SLAM, the users are equipped with an antenna array consisting of nine elements arranged uniformly in a 3×3 grid. The channel estimator tracks both the ToAs and angles of arrival (AoAs) of signal components, which are then used as measurements in the particle filter. Turn rates from a gyroscope that is rigidly mounted to the receiver and thus aligned with the user heading are incorporated as control input in the particle filter. In the online stage of DNN-CC-SLAM, no IMU is used, and a single antenna at the receiver is assumed.

Our DNN is a fully connected feedforward neural network with sigmoid activation functions. The input layer is of size 32, corresponding to the number of ToA estimates of signal components. If the model order, i.e., the number of signal components estimated by the channel estimator, is less or more than 32, we apply zero-padding or truncation, respectively. In the DNN, the number of hidden layers is six with 1000, 300, 200, 100, 50 and 30 neurons. The loss function is given by Eq. (3) and the DNN is trained with the Adam optimizer.

B. Simulation Results

As depicted in Fig. 3, the DNN is trained with data, i.e., estimated ToAs and user positions, from multiple users going through the scenario with cooperative Channel-SLAM. For our evaluations, the training data is obtained from 44 different users with a total traveled distance of approx. 8.86 km. Given

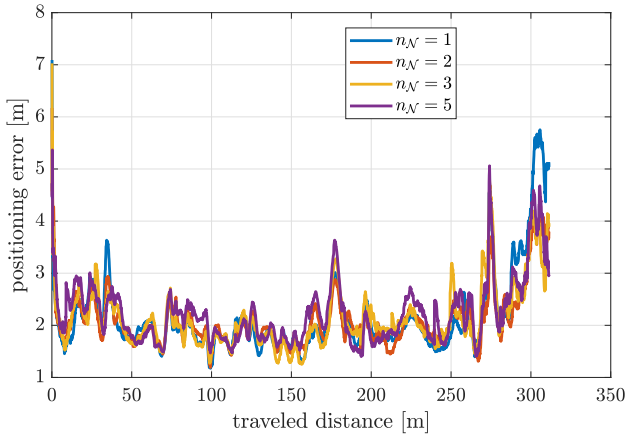

Fig. 5. Positioning error for the reference user using different numbers of Gaussian components in the MDN.

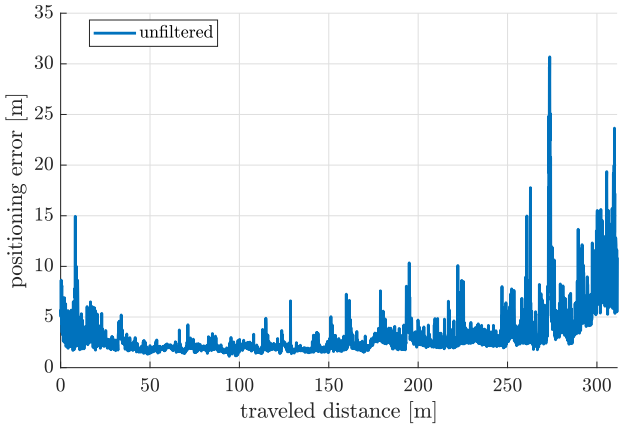

Fig. 6. Positioning error for $N_{\mathcal{N}} = 1$ before the Kalman filter.

a speed of $1\,\mathrm{m/s}$, there are 88,640 data points, of which 90% are used for training and 10% as validation data.

Fig. 5 shows the positioning error of a user going along the reference track depicted in Fig. 4 for different numbers of components $N_{\mathcal{N}}$ in the GMM. All results in Fig. 5 are averaged over 50 repetitions due to the stochastic nature of the particle filter in cooperative Channel-SLAM and of the optimization in training the DNN. Near the end of the reference track, there is less training data available compared to the rest of the track, leading to a higher error in general. The positioning error averaged over the entire traveled distance of the user are $2.80\,\mathrm{m}$, $2.41\,\mathrm{m}$, $2.57\,\mathrm{m}$ and $2.60\,\mathrm{m}$ for $N_{\mathcal{N}} = 1$, $N_{\mathcal{N}} = 2$, $N_{\mathcal{N}} = 3$ and $N_{\mathcal{N}} = 5$, respectively. We see that all curves are mostly very similar. Only towards the end of the track, where fewer training data is available, the error for $N_{\mathcal{N}} = 1$ is considerably higher than for the other cases.

Fig. 6 shows the positioning error for $N_{\mathcal{N}} = 1$ before tracking, i.e., the raw output of DNN predicting the user position. There are considerable outliers in the beginning and especially near the end of the track. The lack of training data near the end leads to an offset from the true user position.

We conclude for our scenario, that in regions with relatively few training data, ambiguities regarding the user position are more likely, and a MDN with more than one component is beneficial. In contrast, if enough training data is available, ambiguities are unlikely. Since the averaged error increases from $N_{\mathcal{N}} = 2$ to $N_{\mathcal{N}} = 5$ in Fig. 5, we suspect that if such ambiguities arise, they occur mainly between two, but not more user positions. Increasing $N_{\mathcal{N}}$ beyond two does not improve the performance, and with $N_{\mathcal{N}} = 2$, our approach can well handle ambiguities in the prediction from the DNN.

VI. Conclusion

We have presented an MDN approach for our multipath assisted positioning-based fingerprinting scheme DNN-CC-SLAM and analyzed ambiguities arising in the DNN. As expected, these ambiguities arise if the user is in an area with few training data. For the most part, ambiguities arise only between two different locations in our scenario. Hence, MDNs with two components are sufficient, and more components do not improve the performance in our scenario.

References

[1] C. Gentner, T. Jost, W. Wang, S. Zhang, A. Dammann, and U.-C. Fiebig, "Multipath Assisted Positioning with Simultaneous Localization and Mapping," *IEEE Trans. Wireless Commun.*, vol. 15, no. 9, pp. 6104–6117, Sep. 2016.

[2] H. Wymeersch, N. Garcia, H. Kim, G. Seco-Granados, S. Kim, F. Wen, and M. Fröhle, "5G mm Wave Downlink Vehicular Positioning," in *IEEE Global Communications Conference (GLOBECOM)*, Dec. 2018, pp. 206–212.

[3] R. Mendrzik, H. Wymeersch, G. Bauch, and Z. Abu-Shaban, "Harnessing NLOS Components for Position and Orientation Estimation in 5G Millimeter Wave MIMO," *IEEE Trans. Wireless Commun.*, vol. 18, no. 1, pp. 93–107, Jan. 2019.

[4] M. Ulmschneider, S. Zhang, C. Gentner, and A. Dammann, "Multipath Assisted Positioning With Transmitter Visibility Information," *IEEE Access*, vol. 8, pp. 155 210–155 223, 2020.

[5] M. Ulmschneider, C. Gentner, and A. Dammann, "Cooperative Estimation of Maps of Physical and Virtual Radio Transmitters," in *Proceedings of the 34th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2021)*, Sep. 2021.

[6] S. He and S. G. Chan, "Wi-Fi Fingerprint-Based Indoor Positioning: Recent Advances and Comparisons," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 466–490, 2016.

[7] M. Ulmschneider and C. Gentner, "User Tracking with Multipath Assisted Positioning-based Fingerprinting and Deep Learning," in *17th European Conference on Antennas and Propagation (EuCAP)*, 2023.

[8] C. M. Bishop, "Mixture Density Networks," Aston University, WorkingPaper, 1994.

[9] T. Jost, W. Wang, U. Fiebig, and F. Perez-Fontan, "Detection and Tracking of Mobile Propagation Channel Paths," *IEEE Trans. Antennas Propag.*, vol. 60, no. 10, pp. 4875–4883, Oct. 2012.

[10] L. Xiao, A. Behboodi, and R. Mathar, "A Deep Learning Approach to Fingerprinting Indoor Localization Solutions," in *27th International Telecommunication Networks and Applications Conference (ITNAC)*, Nov 2017, pp. 1–7.

[11] K. S. Kim, S. Lee, and K. Huang, "A scalable deep neural network architecture for multi-building and multi-floor indoor localization based on Wi-Fi fingerprinting," *Big Data Analytics*, vol. 3, no. 1, Apr. 2018.

[12] T. L. Hansen, B. H. Fleury, and B. D. Rao, "Superfast Line Spectral Estimation," *IEEE Trans. Signal Process.*, vol. 66, no. 10, pp. 2511–2526, May 2018.