Deep Tensor Learning for Reliable Channel Charting from Incomplete and Noisy Measurements

Ge Chen, Panqi Chen and Lei Cheng
College of Information Science and Electronic Engineering
Zhejiang University, Hangzhou, 310027, China
gechen@zju.edu.cn, 12431109@zju.edu.cn, lei_cheng@zju.edu.cn

***Abstract*—Channel charting has emerged as a powerful tool for user equipment localization and wireless environment sensing. Its efficacy lies in mapping high-dimensional channel data into low-dimensional features that preserve the relative similarities of the original data. However, existing channel charting methods are largely developed using simulated or indoor measurements, often assuming clean and complete channel data across all frequency bands. In contrast, real-world channels collected from base stations are typically incomplete due to frequency hopping and are significantly noisy, particularly at cell edges. These challenging conditions greatly degrade the performance of current methods. To address this, we propose a deep tensor learning method that leverages the inherent tensor structure of wireless channels to effectively extract informative while low-dimensional features (i.e., channel charts) from noisy and incomplete measurements. Experimental results demonstrate the reliability and effectiveness of the proposed approach in these challenging scenarios.**

***Index Terms*—Channel charting, noisy and incomplete channel, tensor decomposition , deep learning**

I. Introduction

Modern wireless systems are expected to support denser user equipments (UEs), larger antenna arrays, and an increased number of subcarriers [1]. These trends lead to growing volumes of high-dimensional channel data collected at base stations (BSs), creating new opportunities for intelligent data-driven learning methods to extract knowledge from these large datasets and redefine the potential of wireless sensing, localization, and communication.

To this end, an emerging technique called *channel charting* [2]–[5]maps high-dimensional channel data into low-dimensional representations (also known as pseudo-positions) that preserve the relative similarities of the original data. With a properly designed similarity measure, the resulting low-dimensional channel chart can provide valuable insights into the radio environment, such as user positions/trajectories within a region, and the clustering properties of channels, facilitating various downstream tasks [6], [7].

Earlier works in channel charting primarily rely on classical manifold learning algorithms, such as Multidimensional Scaling (MDS) [8] and t-Distributed Stochastic Neighbor Embedding (t-SNE) [9]. However, these methods are mostly non-parametric, making out-of-sample extensions computationally expensive. Specifically, obtaining the low-dimensional representations (i.e., pseudo-positions) for new channel data requires referencing historical data [10], which is inefficient. To address this limitation, recent approaches leverage deep learning-based [3], [11] dimensionality reduction techniques to learn a parametric mapping, enabling efficient out-of-sample predictions. Among these, the most representative methods are based on Siamese or Triplet network architectures [3], [11]. Their key idea is to train a neural network that takes channel data as input and outputs pseudo-positions, ensuring that the relative dissimilarities among these pseudo-positions closely match those computed from the high-dimensional channel data. Once trained, the neural network can rapidly generate low-dimensional pseudo-positions for any new channel samples, enabling fast channel charting.

Despite their effectiveness, current deep learning-based channel charting methods often *overlook non-ideal* factors encountered in practical scenarios, such as noisy communication environments and incomplete channel measurements (e.g., due to frequency hopping) [12], [13]. Additionally, existing feature extraction techniques for channel data tend to retain excessive information, leading to high-dimensional feature vectors (e.g., computed via auto-correlation) [5]. As a result, neural networks predominantly rely on Multi-Layer Perceptrons (MLPs), significantly increasing the number of parameters and the training burden. These challenges raise a critical question: *How can we design a neural network-based channel charting method that is robust to noisy and incomplete channel data while maintaining a concise model architecture to enable fast training and inference?*

To address this question, we make *the first attempt* to leverage the inherent *low-rank tensor structure* within wireless channel data to design a robust and efficient channel charting network. Recent studies have shown that wireless channels in Multiple-Input Multiple-Output Orthogonal Frequency Division Multiplexing (MIMO-OFDM) systems can be naturally represented using low-rank tensor decomposition models [14]–[17]. By leveraging tensor structures, more accurate channel estimations have been achieved while requiring fewer pilot signals. These findings motivate us to develop a charting network that integrates *tensor decompositions and computations* to mitigate the adverse effects of incomplete and noisy

Corresponding author: Lei Cheng (lei_cheng@zju.edu.cn). This work was supported by the National Natural Science Foundation of China under Grant 62371418.

channel measurements, while reducing the number of network parameters. Experimental results demonstrate the superior performance of our proposed method compared to state-of-the-art approaches.

II. Problem Formulation And Challenges Ahead

A. Channel Model

We consider a MIMO-OFDM system. Following [18], the channel between the BS and the UE is modeled using the 3GPP Cluster Delay Line (CDL) model:

$$h(i_{\text{pol}}, i_{\text{tx}}, i_{\text{rx}}, i_{\text{sc}}) = \sum_{l=0}^{L-1} \alpha_{l,i_{\text{pol}}} e^{-j2\pi\tau_l(i_{\text{sc}}\Delta f)} \times e^{j2\pi f_c \frac{\hat{\mathbf{r}}_{\text{tx},l}^T \bar{\mathbf{d}}_{i_{\text{tx}}}}{c}} e^{j2\pi f_c \frac{\hat{\mathbf{r}}_{\text{rx},l}^T \bar{\mathbf{d}}_{i_{\text{rx}}}}{c}}. \quad (1)$$

Here, $h(i_{\text{pol}}, i_{\text{tx}}, i_{\text{rx}}, i_{\text{sc}})$ denotes the channel coefficient, where $i_{\text{pol}}, i_{\text{tx}}, i_{\text{rx}}$ and i_{sc} represent the polarization index, UE transmit antenna index, BS receive antenna index, and subcarrier index, respectively. The channel coefficient $h(i_{\text{pol}}, i_{\text{tx}}, i_{\text{rx}}, i_{\text{sc}})$ can be interpreted as the sum of the responses from L sub-paths.

In the l-th sub-path, $\alpha_{l,i_{\text{pol}}}$ is the complex gain corresponding to the i_{pol}-th polarization, and τ_l is the propagation delay. The constants Δf, f_c, and c denote the subcarrier spacing, carrier frequency, and speed of light, respectively.

The spherical unit vector $\hat{\mathbf{r}}_{\text{rx},l}$, defined by the azimuth angle of arrival $\phi_{l,\text{AOA}}$ and zenith angle of arrival $\theta_{l,\text{ZOA}}$, is:

$$\hat{\mathbf{r}}_{\text{rx},l} \triangleq \begin{bmatrix} \sin\theta_{l,\text{ZOA}}\cos\phi_{l,\text{AOA}} \\ \sin\theta_{l,\text{ZOA}}\sin\phi_{l,\text{AOA}} \\ \cos\theta_{l,\text{ZOA}} \end{bmatrix}. \quad (2)$$

Similarly, $\hat{\mathbf{r}}_{\text{tx},l}$ is determined by the azimuth angle of departure $\phi_{l,\text{AOD}}$ and the zenith angle of departure $\theta_{l,\text{ZOD}}$. The vectors $\bar{\mathbf{d}}_{i_{\text{tx}}}$ and $\bar{\mathbf{d}}_{i_{\text{rx}}}$ specify the three-dimensional Cartesian coordinates of the i_{tx}-th UE antenna and the i_{rx}-th BS antenna, respectively.

Note that $h(i_{\text{pol}}, i_{\text{tx}}, i_{\text{rx}}, i_{\text{sc}})$ depends on multiple indices; therefore, it can be naturally represented as a channel tensor $\boldsymbol{\mathcal{H}} \in \mathbb{C}^{N_{\text{rx}} \times N_{\text{pol}} \times N_{\text{tx}} \times N_{\text{sub}}}$, where $N_{\text{rx}}, N_{\text{pol}}, N_{\text{tx}}$, and N_{sub} denote the number of BS antennas, polarization modes, UE antennas, and subcarriers, respectively.

B. Channel Charting and Challenges Ahead

Channel charting can be expressed as a mapping function:

$$\mathbf{z}(\boldsymbol{\mathcal{H}}_i) : \mathbb{C}^{N_{\text{rx}} \times N_{\text{pol}} \times N_{\text{tx}} \times N_{\text{sub}}} \to \mathbb{R}^D \quad (3)$$

which projects the high-dimensional channel $\boldsymbol{\mathcal{H}}_i$ into a lower-dimensional pseudo-position $\mathbf{z}(\boldsymbol{\mathcal{H}}_i)$, where $D \ll N_{\text{rx}} N_{\text{pol}} N_{\text{tx}} N_{\text{sub}}$.

The goal is to preserve the relative similarity between channels, meaning that the distance between $\mathbf{z}(\boldsymbol{\mathcal{H}}_i)$ and $\mathbf{z}(\boldsymbol{\mathcal{H}}_j)$ should reflect the dissimilarity $d_{ij} \triangleq d(\boldsymbol{\mathcal{H}}_i, \boldsymbol{\mathcal{H}}_j)$ in the original space. Accordingly, channel charting can be formulated as the optimization problem:

$$\min_{\mathbf{z}(\cdot)} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \Big(d_{ij} - \|\mathbf{z}(\boldsymbol{\mathcal{H}}_i) - \mathbf{z}(\boldsymbol{\mathcal{H}}_j)\|_2 \Big)^2. \quad (4)$$

To address this problem, various charting algorithms have been proposed, broadly categorized into manifold learning-based and deep learning-based approaches [4], [5]. We focus on the latter due to their ability to learn a parametric mapping $\mathbf{z}_{\boldsymbol{\theta}}(\cdot)$, which enables efficient out-of-sample predictions, where $\boldsymbol{\theta}$ represents the neural network parameters.

A major challenge in deep learning-based methods is the lack of direct support for complex-valued inputs in standard frameworks (e.g., PyTorch). To address this, carefully designed channel features are often used to encode channel properties for training. Most existing feature representations rely on auto-correlation [5] in the time or angular domain [2], resulting in extremely high-dimensional features. Consequently, neural networks processing these flattened features—typically implemented as MLPs—incur a high parameter count and significant training overhead. Furthermore, current methods seldom account for real-world impairments such as noisy or incomplete measurements. *This limitation reduces their robustness and efficiency in practical deployments, highlighting the need for more resilient feature representations and concise neural architectures.*

III. Proposed Approach

In this section, we introduce our *deep tensor learning* (DTL) framework for reliable channel charting. Similar to existing methods, DTL begins with a feature engineering module $F(\cdot)$ that transforms raw channel data $\boldsymbol{\mathcal{H}}$ into a more robust tensor representation $F(\boldsymbol{\mathcal{H}})$, mitigating real-world impairments. A tensor Tucker decomposition-based denoising module $D(\cdot)$ then further suppress noise [19]. Unlike prior approaches relying on MLP-based dimensionality reduction, our framework employs a *tensor contraction layer* [20]–[22] operating directly on tensor-based features to produce the final channel chart. Fig. 1 illustrates the overall architecture.

A. Part 1: Tensor Feature Engineering

The feature engineering module $F(\cdot)$ is crucial yet challenging in channel charting. It must strike a balance between expressiveness and conciseness while ensuring robustness. Retaining too much raw data increases feature dimensions and network parameters, while overly compact features risk losing essential channel details, degrading chart quality.

Towards these goals, we propose computing a *spatial covariance tensor* $\boldsymbol{\mathcal{T}} \in \mathbb{C}^{N_{\text{rx}} \times N_{\text{rx}} \times \frac{N_{\text{sub}}}{h_p}}$ for each channel tensor $\boldsymbol{\mathcal{H}}$, as illustrated in Part 1 of Fig. 1. Concretely, to mitigate the impact of incomplete channel information along the subcarrier dimension due to frequency hopping, we assume a $\frac{1}{h_p}$ frequency hopping, meaning each observation captures only $\frac{1}{h_p}$ of the total subcarriers. As illustrated in the red box of Fig. 1, we select every h_p channel slice matrices to form $\frac{N_{\text{sub}}}{h_p}$ subtensors along the subcarrier mode:

$$\boldsymbol{\mathcal{H}}_i = \boldsymbol{\mathcal{H}}(:,:,:,\mathcal{J}_i) \in \mathbb{C}^{N_{\text{rx}} \times N_{\text{pol}} \times N_{\text{tx}} \times h_p}, i = 1, ..., \tfrac{N_{\text{sub}}}{h_p}, \quad (5)$$

where the index set of the i-th group is:

$$\mathcal{J}_i = \Big\{ i + \tfrac{N_{\text{sub}}}{h_p}(k-1) \mid k = 1, \dots, h_p \Big\}, i = 1, \dots, \tfrac{N_{\text{sub}}}{h_p}. \quad (6)$$

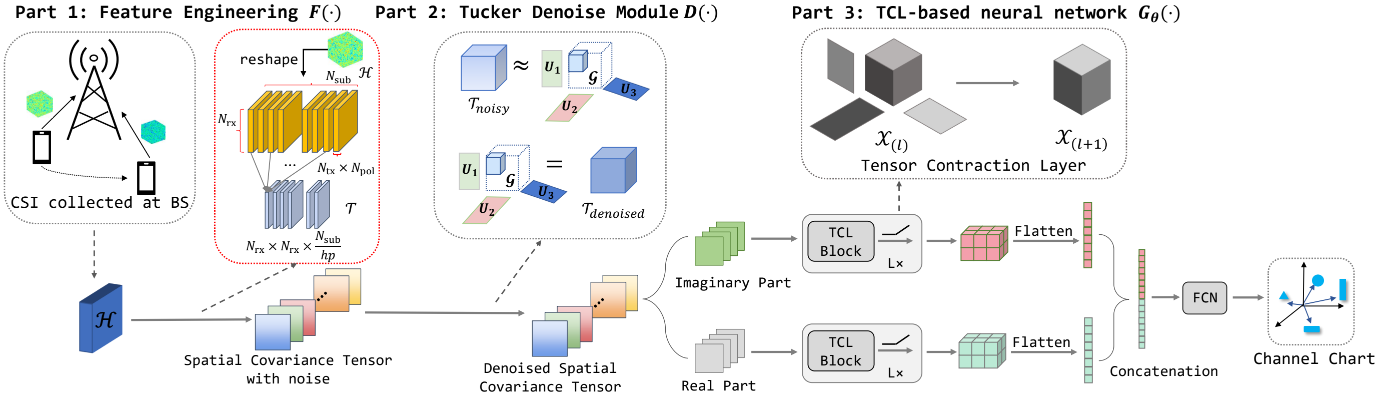
Part 1: Feature Engineering $F(\cdot)$
Part 2: Tucker Denoise Module $D(\cdot)$
Part 3: TCL-based neural network $G_{\theta}(\cdot)$
CSI collected at BS
reshape
N_{sub}
$\mathcal{H}$
N_{rx}
$N_{\text{tx}} \times N_{\text{pol}}$
$\mathcal{T}$
$N_{\text{rx}} \times N_{\text{rx}} \times \frac{N_{\text{sub}}}{hp}$
$\mathcal{T}_{noisy}$
$\approx$
U_1
$\mathcal{G}$
U_2
U_3
$=$
$\mathcal{T}_{denoised}$
$\mathcal{X}_{(l)}$
$\mathcal{X}_{(l+1)}$
Tensor Contraction Layer
Spatial Covariance Tensor with noise
Denoised Spatial Covariance Tensor
Imaginary Part
Real Part
TCL Block
L×
Flatten
Concatenation
FCN
Channel Chart

Fig. 1: The detailed architecture of our proposed deep tensor learning model for reliable channel charting.

For the i-th subtensor, we perform a *mode-1 unfolding* of $\boldsymbol{\mathcal{H}}_i$ to obtain $\mathbf{H}_{(i),1} \in \mathbb{C}^{N_{\text{rx}} \times (N_{\text{pol}}\, N_{\text{tx}}\, h_p)}$. We then compute its covariance matrix along the first mode (known as the spatial covariance matrix):

$$\mathbf{C}_i = \frac{1}{N_{\text{pol}}\, N_{\text{tx}}\, h_p} \mathbf{H}_{(i),1} \mathbf{H}_{(i),1}^H \in \mathbb{C}^{N_{\text{rx}} \times N_{\text{rx}}}. \tag{7}$$

Finally, we stack these spatial covariance matrices to construct the spatial covariance $\boldsymbol{\mathcal{T}} \in \mathbb{C}^{N_{\text{rx}} \times N_{\text{rx}} \times \frac{N_{\text{sub}}}{h_p}}$, i.e., $\boldsymbol{\mathcal{T}}(:,:,i) = \mathbf{C}_i$.

The proposed spatial covariance tensor $\boldsymbol{\mathcal{T}}$ captures informative spatial correlations among BS antennas across different subbands, *making it expressive*. Its dimensionality is significantly smaller than that of the original channel tensor or conventional auto-correlation feature vectors, *ensuring conciseness*. Furthermore, partial observations (due to frequency hopping) still allow the computation of $\boldsymbol{\mathcal{T}}$, and the averaging operation in (7) helps mitigate noise to some extent. Therefore, $\boldsymbol{\mathcal{T}}$ is also *robust* against noise and missing channel data.

B. Part 2: Tucker Denoise Module

We begin by recalling tensor Tucker decomposition [19]. For a third-order tensor $\boldsymbol{\mathcal{Y}} \in \mathbb{C}^{I_1 \times I_2 \times I_3}$, its Tucker representation is

$$\boldsymbol{\mathcal{Y}} \approx \boldsymbol{\mathcal{G}} \times_1 \mathbf{U}_1 \times_2 \mathbf{U}_2 \times_3 \mathbf{U}_3, \tag{8}$$

where $\mathbf{U}_n \in \mathbb{C}^{I_n \times R_n}$ $(n = 1,2,3)$ are orthonormal factor matrices $(\mathbf{U}_n^H \mathbf{U}_n = \mathbf{I}_n)$, $R_n \leq I_n$, and $\boldsymbol{\mathcal{G}} \in \mathbb{C}^{R_1 \times R_2 \times R_3}$ is the core tensor. Here, $\times_l$ denotes a mode-l product: for an L-th order tensor $\boldsymbol{\mathcal{A}} \in \mathbb{C}^{I_1 \times \cdots \times I_L}$ and a matrix $\mathbf{B} \in \mathbb{C}^{J_l \times I_l}$, the result $\boldsymbol{\mathcal{C}} = \boldsymbol{\mathcal{A}} \times_l \mathbf{B}$ satisfies

$$\boldsymbol{\mathcal{C}}_{i_1,\ldots,j_l,\ldots,i_L} = \sum_{k=1}^{I_l} \boldsymbol{\mathcal{A}}_{i_1,\ldots,k,\ldots,i_L} \mathbf{B}_{j_l,k}. \tag{9}$$

Similar to SVD, choosing relatively small $\{R_n\}_{n=1}^3$ in Tucker decomposition helps filter out noise:

$$\boldsymbol{\mathcal{Y}}_{\text{recons}} = \boldsymbol{\mathcal{G}} \times_1 \mathbf{U}_1 \times_2 \mathbf{U}_2 \times_3 \mathbf{U}_3. \tag{10}$$

Motivated by this principle, we apply (8) and (10) to $\boldsymbol{\mathcal{T}}$ to obtain $\boldsymbol{\mathcal{T}}_{\text{denoised}}$, as illustrated in Part 2 of Fig. 1. For neural network training, we then separate its real and imaginary parts, $\Re(\boldsymbol{\mathcal{T}}_{\text{denoised}})$ and $\Im(\boldsymbol{\mathcal{T}}_{\text{denoised}})$, which serve as inputs to subsequent layers. We refer to this entire process as the *Tucker denoise module*, denoted by $D(\cdot)$.

C. Part 3: Integrating Tensor Computations in the Neural Network

Our framework employs a *tensor contraction layer* (TCL) [20], [21] to process the feature tensors, as illustrated in Part 3 of Fig. 1. Let $\boldsymbol{\mathcal{X}}_{(l)} \in \mathbb{R}^{I_1^l \times I_2^l \times I_3^l}$ be the input to the l-th TCL. Its output is

$$\boldsymbol{\mathcal{X}}_{(l+1)} = \zeta\Big(\boldsymbol{\mathcal{X}}_{(l)} \times_1 \mathbf{V}_{(l)}^{(1)} \times_2 \mathbf{V}_{(l)}^{(2)} \times_3 \mathbf{V}_{(l)}^{(3)}\Big), \tag{11}$$

where $\{\mathbf{V}_{(l)}^{(i)} \in \mathbb{R}^{I_i^{l+1} \times I_i^l}\}_{i=1}^3$ are learnable factor matrices, $\boldsymbol{\mathcal{X}}_{(l+1)} \in \mathbb{R}^{I_1^{(l+1)} \times I_2^{(l+1)} \times I_3^{(l+1)}}$, and $\zeta(\cdot)$ is activation function.

Because TCL builds upon Tucker decomposition, it effectively captures multi-dimensional structures with relatively few parameters. To further increase model capacity, we stack multiple TCLs to form an L-layer block:

$$\begin{aligned}\boldsymbol{\mathcal{X}}_{(L)} = \zeta\Big(\ldots\zeta\big(\boldsymbol{\mathcal{X}}_{(0)} \times_1 \mathbf{V}_{(0)}^{(1)} \times_2 \mathbf{V}_{(0)}^{(2)} \times_3 \mathbf{V}_{(0)}^{(3)}\big)\ldots \\ \times_1 \mathbf{V}_{(L-1)}^{(1)} \times_2 \mathbf{V}_{(L-1)}^{(2)} \times_3 \mathbf{V}_{(L-1)}^{(3)}\Big),\end{aligned} \tag{12}$$

where $\boldsymbol{\mathcal{X}}_{(0)}$ is the initial tensor input and $\{\mathbf{V}_{(i)}^{(j)} | i = 0, \ldots, L-1;\ j = 1,2,3\}$ are learnable factors. Let $\boldsymbol{\mathcal{X}}_{\text{Re},(L)}$ and $\boldsymbol{\mathcal{X}}_{\text{Im},(L)}$ denote outputs of these two L-layer TCL blocks with inputs $\Re(\boldsymbol{\mathcal{T}}_{\text{denoised}})$ and $\Im(\boldsymbol{\mathcal{T}}_{\text{denoised}})$. We vectorize and concatenate them as

$$\text{Concat}(\text{Vec}(\boldsymbol{\mathcal{X}}_{\text{Re},(L)}), \text{Vec}(\boldsymbol{\mathcal{X}}_{\text{Im, (L)}})) \in \mathbb{R}^{2(I_1^L \times I_2^L \times I_3^L)}, \tag{13}$$

which is finally fed into a fully connected network (FCN) to produce the channel chart. We denote the overall parametric mapping as $G_{\boldsymbol{\theta}}(\cdot)$, where $\boldsymbol{\theta}$ denotes the set of parameters in TCL layers and FCN.

D. Training and Inference

In the channel charting optimization problem (4), defining appropriate dissimilarity metrics d_{ij} is crucial. Various metrics have been proposed to quantify distances between channel data, but many fail to account for real-world impairments. As outlined in [5], these metrics typically fall into two categories: side information-based and channel data-based. In this work, we focus on the latter. Motivated by the importance of channel covariance matrices in MIMO-OFDM systems [23] and their robustness against noise and missing channel data, we adopt a spatial covariance matrix-based dissimilarity:

$$d_{\text{SCM},ij} = 1 - \frac{\text{trace}\big(\mathbf{C}_i^H \mathbf{C}_j\big)}{||\mathbf{C}_i||_F ||\mathbf{C}_j||_F}, \tag{14}$$

where $\mathbf{C}_i$ and $\mathbf{C}_j$ are the spatial covariance matrices of the i-th and j-th channel observations, respectively. As illustrated in [5], [24], geodesic dissimilarity provides a more rational representation of the distance between channel data and is therefore applied in our work. We adopt a similar procedure as Sec.III.H in [5] to get $d_{\text{G-SCM},ij}$ for our network training.

As illustrated in Fig. 1 and according to Sec. III A-C, our proposed channel charting function is expressed as:

$$\mathbf{z}(\cdot) = G_{\boldsymbol{\theta}}(D(F(\cdot))). \tag{15}$$

Consequently, the training objective of our *DTL-based channel charting* is:

$$\begin{aligned}\mathcal{L}_{\text{TDL-charting}} = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \Big(& d_{\text{G-SCM},ij} \\ & - \|G_{\boldsymbol{\theta}}(D(F(\boldsymbol{\mathcal{H}}_i))) - G_{\boldsymbol{\theta}}(D(F(\boldsymbol{\mathcal{H}}_j)))\|_2 \Big)^2, \end{aligned} \tag{16}$$

where the network parameters can be learned via Adam optimizer [25].

During inference, let $\tilde{\boldsymbol{\mathcal{H}}}$ be an out-of-sample channel data to be charted, and $\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathcal{L}_{\text{TDL-charting}}$. Then, its pseudo-position is given by:

$$\tilde{\mathbf{z}} = G_{\boldsymbol{\theta}^*}(D(F(\tilde{\boldsymbol{\mathcal{H}}}))). \tag{17}$$

IV. Experimental Results

We evaluate our method using Quadriga-simulated channel data consisting 1000 samples. The BS is equipped with a 4×8 dual-polarized rectangular antenna array, yielding $N_{\text{rx}} = 32, N_{\text{pol}} = 2$. The UE has two transmit antennas ($N_{\text{tx}} = 2$), and each channel data includes 408 subcarriers ($N_{\text{sub}} = 408$) spaced over a 240kHz bandwidth with a 3.5GHz carrier frequency. For incomplete observations, we consider $1/17$ frequency hopping, i.e., only consecutive $1/17$ subcarrier data can be observed at a time. Channel data is mapped into two-dimensional pseudo-positions, i.e., in (3), $D = 2$.

Baselines: We compare 5 channel charting algorithms: two deep learning-based methods—Angle-Delay Profile-based (ADP) [5] and Raw 2$^{\text{nd}}$-moment-based (R2M) [2], and three manifold learning-based methods—UMAP [26], t-SNE [9] and Isomap [24]. This paper primarily focuses on deep learning-based methods, as their parametric nature enables the rapid generation of pseudo-positions for new test channels, which is crucial for real-world channel charting applications. In contrast, manifold learning-based methods are non-parametric, making them difficult to extend for out-of-sample charting. These methods are included for comparison to assess the effectiveness of our proposed approach. For deep learning approaches, the batch size, learning rate and training epochs are set to 32, 1×10^{-3} and 300, respectively.

TABLE I: Channel Charting Results For Ideal Channel Data

	Method	CT↑	TW↑	KS↓	Parameters	Out-of-sample
Deep learning-based	**DTL**	**0.9922**	**0.9732**	**0.1304**	**11730**	**Easy**
	ADP	0.9604	0.9264	0.2027	1.5B	Easy
	R2M	0.9152	0.9475	0.2474	985M	Easy
Manifold learning-based	UMAP	0.9929	0.9866	0.3846	/	Hard
	t-SNE	0.9960	0.9989	0.4180	/	Hard
	Isomap	0.9829	0.9750	0.1375	/	Hard

TABLE II: Channel Charting Results Under Noisy And Incomplete Observations

	Method	SNR: 0dB			1/17 Frequency Hopping			Out-of-sample
		CT↑	TW↑	KS↓	CT↑	TW↑	KS↓	
Deep learning-based	**DTL**	**0.9875**	**0.9727**	**0.1669**	**0.9866**	**0.9665**	**0.1781**	**Easy**
	ADP	0.8991	0.8794	0.3595	/	/	/	Easy
	R2M	0.9087	0.8797	0.5829	/	/	/	Easy
Manifold learning-based	UMAP	0.9940	0.9857	0.3595	0.9913	0.9848	0.4042	Hard
	t-SNE	0.9952	0.9977	0.4008	0.9930	0.9967	0.3883	Hard
	Isomap	0.9776	0.9727	0.1672	0.9757	0.9677	0.1795	Hard

Performance metrics: We adopt three widely used metrics in channel charting: *continuity* (CT), *trustworthiness* (TW), and *Kruskal's stress* (KS). All three range from 0 to 1. Specifically, CT and TW are optimal at 1, indicating local-neighborhood preservation. KS is optimal at 0 to capture global-structure fidelity. For details of these definitions, see [5]. We also compare the number of trainable parameters for each deep learning method.

Results: Tab. I summarizes the performance of various charting methods using ideal channel data. Among the deep learning approaches, our proposed DTL achieves the highest CT and TW while yielding the lowest KS, which is attributed to our tensor modeling for feature extraction. Also, our tensor-based neural network leads to significantly fewer parameters, confirming that the tensor computations are effective for network-size reduction. In contrast, UMAP and t-SNE perform the best in terms of CT and TW, but exhibit poor performance in KS. This is because these algorithms prioritize preserving local structure instead of global structure.

Tab. II shows charting results under noisy and incomplete scenarios (SNR=10dB, $1/17$ frequency hopping). Our proposed DTL model still outperforms other deep learning approaches with minimal performance degradation. The large channel features and network parameters in ADP and R2M make them highly susceptible to noise, and they struggle to compute physically meaningful channel features and dissimilarity distances under incomplete observations. In contrast, our DTL model fully leverages low-rank tensor structure in feature design, making it robust and reliable to noisy and incomplete measurements. Additionally, manifold learning algorithms, which only require dissimilarity distance matrix for training, demonstrate stable performance, indicating the inherent robustness of our designed metric (14) in practical settings.

Fig. 2 shows our charting results. Following [5], [27], we adopt an optimal affine transformation to the channel chart obtained under noisy and incomplete measurements. Clearly, both local and global structures are preserved well under real-world impairments compared to ideal chart, which indicates the reliability of our proposed DTL for channel charting.

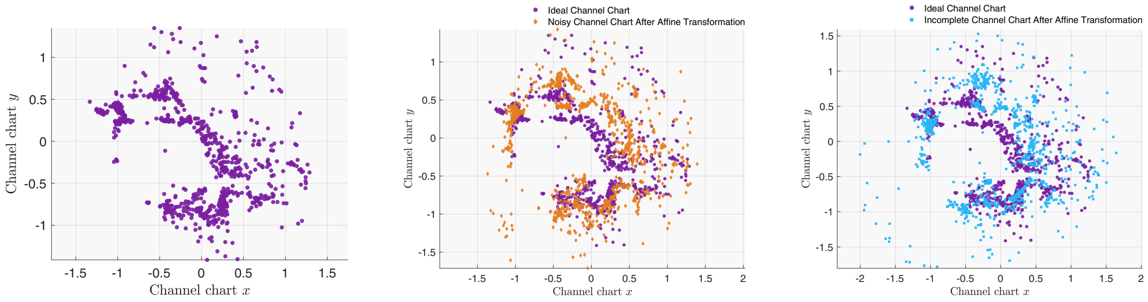

Fig. 2: (Left) Channel chart with ideal data. (Middle) Channel charts using noisy data (SNR = 0 dB, orange points) and ideal data (purple points). (Right) Channel charts via incomplete data (1/17 frequency hopping, blue points) and ideal data (purple points). Despite real-world impairments, both the local and global structures in our DTL-based charting results are well preserved compared to the ideal chart, demonstrating the reliability of our proposed DTL for channel charting.

V. Conclusion

In this paper, we introduce a deep tensor learning method for reliable channel charting under noisy and incomplete measurements. By exploiting the inherent low-rank structure of MIMO-OFDM channels, we effectively extract features while suppressing noise and handling missing entries. Additionally, our work highlights the importance of integrating domain-specific multi-dimensional data priors into neural networks for processing wireless channel data. Experimental results demonstrate that DTL outperforms state-of-the-art deep learning-based methods with significantly reduced model complexity and is therefore well-suited for practical deployment.

References

[1] Y. Zeng, J. Chen, J. Xu, D. Wu, X. Xu, S. Jin, X. Gao, D. Gesbert, S. Cui, and R. Zhang, "A tutorial on environment-aware communications via channel knowledge map for 6g," *IEEE communications surveys & tutorials*, vol. 26, no. 3, pp. 1478–1519, 2024.

[2] C. Studer, S. Medjkouh, E. Gonultaş, T. Goldstein, and O. Tirkkonen, "Channel charting: Locating users within the radio environment using channel state information," *IEEE Access*, vol. 6, pp. 47 682–47 698, 2018.

[3] P. Ferrand, A. Decurninge, L. G. Ordonez, and M. Guillaud, "Triplet-based wireless channel charting: Architecture and experiments," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 8, pp. 2361–2373, 2021.

[4] P. Ferrand, M. Guillaud, C. Studer, and O. Tirkkonen, "Wireless channel charting: Theory, practice, and applications," *IEEE Communications Magazine*, vol. 61, no. 6, pp. 124–130, 2023.

[5] P. Stephan, F. Euchner, and S. Ten Brink, "Angle-delay profile-based and timestamp-aided dissimilarity metrics for channel charting," *IEEE Transactions on Communications*, 2024.

[6] L. Le Magoarou, T. Yassine, S. Paquelet, and M. Crussière, "Channel charting based beamforming," in *2022 56th Asilomar Conference on Signals, Systems, and Computers*. IEEE, 2022, pp. 1185–1189.

[7] P. Kazemi, T. Ponnada, H. Al-Tous, Y.-C. Liang, and O. Tirkkonen, "Channel charting based beam snr prediction," in *2021 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit)*. IEEE, 2021, pp. 72–77.

[8] J. B. Kruskal, "Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis," *Psychometrika*, vol. 29, no. 1, pp. 1–27, 1964.

[9] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.

[10] Y. Bengio, J.-f. Paiement, P. Vincent, O. Delalleau, N. Roux, and M. Ouimet, "Out-of-sample extensions for lle, isomap, mds, eigenmaps, and spectral clustering," *Advances in neural information processing systems*, vol. 16, 2003.

[11] E. Lei, O. Castañeda, O. Tirkkonen, T. Goldstein, and C. Studer, "Siamese neural networks for wireless positioning and channel charting," in *2019 57th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, 2019, pp. 200–207.

[12] Y. Liu, "Frequency hopping spread spectrum: An effective way to improve wireless communication performance," *Advanced Trends in Wireless Communications*, pp. 187–202, 2011.

[13] K. Wu, J. A. Zhang, X. Huang, and Y. J. Guo, "Frequency-hopping mimo radar-based communications: An overview," *IEEE Aerospace and Electronic Systems Magazine*, vol. 37, no. 4, pp. 42–54, 2021.

[14] D. C. Araújo, A. L. De Almeida, J. P. Da Costa, and R. T. de Sousa, "Tensor-based channel estimation for massive mimo-ofdm systems," *IEEE Access*, vol. 7, pp. 42 133–42 147, 2019.

[15] Z. Zhou, J. Fang, L. Yang, H. Li, Z. Chen, and R. S. Blum, "Low-rank tensor decomposition-aided channel estimation for millimeter wave mimo-ofdm systems," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 7, pp. 1524–1538, 2017.

[16] R. Zhang, L. Cheng, S. Wang, Y. Lou, W. Wu, and D. W. K. Ng, "Tensor decomposition-based channel estimation for hybrid mmwave massive mimo in high-mobility scenarios," *IEEE Transactions on Communications*, vol. 70, no. 9, pp. 6325–6340, 2022.

[17] R. Zhang, L. Cheng, W. Zhang, X. Guan, Y. Cai, W. Wu, and R. Zhang, "Channel estimation for movable-antenna mimo systems via tensor decomposition," *IEEE Wireless Communications Letters*, 2024.

[18] "Study on channel model for frequencies from 0.5 to 100 GHz," 3GPP, Sophia Antipolis, France, Tech. Rep. TR 38.901 V16.1.0, 2019, available: https://www.3gpp.org.

[19] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM review*, vol. 51, no. 3, pp. 455–500, 2009.

[20] J. Kossaifi, A. Khanna, Z. Lipton, T. Furlanello, and A. Anandkumar, "Tensor contraction layers for parsimonious deep nets," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2017, pp. 26–32.

[21] J. Kossaifi, Z. C. Lipton, A. Kolbeinsson, A. Khanna, T. Furlanello, and A. Anandkumar, "Tensor regression networks," *Journal of Machine Learning Research*, vol. 21, no. 123, pp. 1–21, 2020.

[22] S. Li, L. Cheng, T. Zhang, H. Zhao, and J. Li, "Striking the right balance: Three-dimensional ocean sound speed field reconstruction using tensor neural networks," *The Journal of the Acoustical Society of America*, vol. 154, no. 2, pp. 1106–1123, 2023.

[23] K. Li, Y. Li, L. Cheng, Q. Shi, and Z.-Q. Luo, "Downlink channel covariance matrix reconstruction for fdd massive mimo systems with limited feedback," *IEEE Transactions on Signal Processing*, vol. 72, pp. 1032–1048, 2024.

[24] J. B. Tenenbaum, V. d. Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

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

[26] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," *arXiv preprint arXiv:1802.03426*, 2018.

[27] M. Stahlke, G. Yammine, T. Feigl, B. M. Eskofier, and C. Mutschler, "Indoor localization with robust global channel charting: A time-distance-based approach," *IEEE Transactions on Machine Learning in Communications and Networking*, vol. 1, pp. 3–17, 2023.