SMART: Sim2Real Meta-Learning-Based Training for mmWave Beam Selection in V2X Networks

Divyadharshini Muruganandham, Suyash Pradhan, *Graduate Student Member, IEEE*, Jerry Gu, Torsten Braun, *Senior Member, IEEE*, Debashri Roy, *Senior Member, IEEE*, and Kaushik Chowdhury, *Fellow, IEEE*

***Abstract*—Digital twins (DT) offer a low-overhead evaluation platform and the ability to generate rich datasets for training machine learning (ML) models before actual deployment. Specifically, for the scenario of ML-aided millimeter wave (mmWave) links between moving vehicles to roadside units, we show how DT can create an accurate replica of the real world for model training and testing. The contributions of this paper are twofold: First, we propose a framework to create a multimodal Digital Twin (DT), where synthetic images and LiDAR data for the deployment location are generated along with RF propagation measurements obtained via ray-tracing. Second, to ensure effective domain adaptation, we leverage *meta-learning*, specifically *Model-Agnostic Meta-Learning* (MAML), with *transfer learning* (TL) serving as a baseline validation approach. The proposed framework is validated using a comprehensive dataset containing both real and synthetic LiDAR and image data for mmWave V2X beam selection. It also enables the investigation of how each sensor modality impacts domain adaptation, taking into account the unique requirements of mmWave beam selection. Experimental results show that models trained on synthetic data using transfer learning and meta-learning, followed by minimal fine-tuning with real-world data, achieve up to 4.09× and 14.04× improvements in accuracy, respectively. These findings highlight the potential of synthetic data and meta-learning to bridge the domain gap and adapt rapidly to real-world beamforming challenges.**

***Index Terms*—Digital twin (DT), transfer learning, meta-learning, multimodal data, mmWave, beam selection.**

I. Introduction

VEHICLE-TO-EVERYTHING (V2X) networks, in which a vehicle may communicate with or respond to stimuli from other vehicles, infrastructure, pedestrians, or networks, are shaping not only autonomous driving [1] but also our social and entertainment experiences [2], [3]. Central to this capability is creating high-bandwidth communication links, such as in the millimeter wave (mmWave) band, which can relay data at Gbps rates [4]. To overcome the lengthy beam sweeping process defined by the standard for such links [5], [6], recent works have proposed leveraging machine learning (ML) [7], [8], [9]. However, training such ML models requires difficult-to-obtain large datasets collected in the real world, along with data from risky, outlier conditions to stress-test the model, which may even be infeasible. In this paper, we address these challenges for ML-aided V2X connectivity by designing digital twins (DT) that create rich replicas of the real world for model training and testing.

A. DT for Mitigating Scarcity of Real-World Data

Given several different environmental conditions that impact mmWave links, such as blockage due to buildings, pedestrians, and other cars, and the difficulty of setting up a persistent experimental testbed in mobile, congested urban neighborhoods, we need to consider alternate sources of V2X data collection. One promising direction for V2X data collection is using software that emulates the practical deployment conditions. The resulting high-fidelity DT can construct virtual environments and run experiments spanning lengthy intervals of time, thus solving the challenge of missing data. Moreover, it can emulate adversarial situations that may rarely occur or have dangerous consequences (e.g., a pedestrian ahead of an oncoming vehicle).

B. Non-RF Modalities in Mmwave V2X Beam Selection

Despite the many benefits of mmWave bands in establishing high bandwidth links, they are highly directional in nature and thus, suffer from misalignment, blockages, and atmospheric absorption, to name a few. We have had previous success with using non-RF modalities for the purpose of selecting the best beam in V2X scenarios, either individually or in conjunction with RF data, to provide a richer representation of the environment [10], [11]. Specifically, we have used synchronized image, LiDAR, and Global Positioning System (GPS) data from sensors installed in an actual autonomous car, in addition to mmWave RF data, to train ML models for beam selection. While intuitive, again, collecting such multimodal data is time and cost-restrictive from using actual testbeds. Hence, we would like to design a DT that also allows virtual captures of similar camera

Received 4 April 2025; accepted 21 May 2025. Date of publication 3 June 2025; date of current version 3 September 2025. This work was supported by the US National Science Foundation under Grant 2516080 and Grant 2120447. Recommended for acceptance by D. Xu. *(Corresponding author: Divyadharshini Muruganandham.)*

Divyadharshini Muruganandham, Suyash Pradhan, and Kaushik Chowdhury are with the Wireless Networking, Communications Group, University of Texas at Austin, Austin, TX 78712 USA (e-mail: muruganandham.d@utexas.edu; suyash.p@utexas.edu; kaushik@utexas.edu).

Jerry Gu is with Northeastern University, Boston, MA 02115 USA (e-mail: gu.je@northeastern.edu).

Torsten Braun is with the Institute of Computer Science, University of Bern, 3012 Bern, Switzerland (e-mail: torsten.braun@unibe.ch).

Debashri Roy is with the Department of Computer Science, Engineering, University of Texas at Arlington, Arlington, TX 76019 USA (e-mail: debashri.roy@uta.edu).

Digital Object Identifier 10.1109/TMC.2025.3576203

images and LiDAR pointclouds, generating valuable synthetic datasets for training ML models.

C. Challenges in DT Design

While DT hold great promise in training and testing models when real-world data is difficult to obtain, there are many open challenges that need to be addressed.

• *Challenge 1- Creation of DT for multimodal synthetic data generation*: There are practical considerations that should be scrutinized at every stage of DT implementation. To begin, the choice of software or data generation application has a large impact on every parameter of the dataset, from the degree of flexibility of open-sourced tools or the licensing cost and duration of proprietary applications to the compatibility of the different data types that will be generated within a multimodal dataset. There will be some trade-offs between data resolution, computation, and automation that should be addressed to create the best-suited recreation of the real-world analog. Finally, data organization and descriptive metadata, particularly when multiple variations of similar scenarios are generated, need to be considered, especially if the DT suite or resultant dataset will be made publicly available.

• *Challenge 2- Bridging the Simulation-to-Reality Gap in Digital Twin-Based mmWave V2X Communication:*

a) *Meta-Learning Driven Domain Adaptation for Enhanced Sim2Real Transfer in mmWave V2X Communication:* While Challenge 1 addresses the creation of a high-fidelity DT, there remains a simulation-to-reality gap that must be addressed for effective deployment. Our prior works have focused on applying advanced machine learning techniques—including transfer learning (TL) [12] and model-agnostic meta-learning (MAML) [13], [14] to enable robust adaptation to unseen scenarios. We have demonstrated that selecting an appropriate starting scenario for TL initiation and intelligently freezing selected layers during fine-tuning can significantly improve performance. Similarly, we leveraged MAML in [15] to utilize limited data across a number of tasks *seen* during training (e.g., pedestrian blockage) to find an adaptive model that, when fine-tuned during testing time, performs well on new or *unseen* scenarios (e.g., moving vehicular blockage), given that these tasks still fall under a common distribution. However, in both these works, we trained and tested on data collected in the real world only. How such models perform when transferring knowledge from a multimodal DT to the real world remains unknown.

b) *Domain adaptation from DT to real world:* A key challenge in this transfer process is that real-world data often contains noise, artifacts, and imperfections not present in the synthetic dataset. Despite employing advanced sensor simulation tools for DT creation, this gap will exist due to differences in illumination, texture, and optical distortions in camera images, while LiDAR point clouds experience variations in interference intensities and occlusions. It would be particularly beneficial to examine how the performance of each sensor modality influences domain adaptation, especially when considering an unconventional ML task of mmWave beam selection.

D. MAML for Mmwave V2X Beam Selection: Motivation for Designing SMART

In mmWave V2X communication, rapid environmental changes due to vehicular obstacles and mobility create significant challenges for beam selection using conventional supervised learning, primarily due to the lack of large-scale real-world datasets. As an alternative, this paper explores the potential of Digital Twins (DT) to generate synthetic datasets that replicate real-world environments, providing an initial dataset for training machine learning models where real-world data is unavailable. However, the limited availability of synthetic data and constrained training time in V2X setups make traditional supervised learning approaches impractical. To address this, we propose SMART, an adaptive meta-learning-based beam selection framework optimized for domain adaptation, bridging the Sim2Real gap specifically for beam selection in V2X communication.

E. Our Contributions:

In this paper, we demonstrate how a combination of approaches, such as non-RF modality usage, DT-generated data, and meta-learning can address the challenges of long communication overheads and training time with limited data for mmWave V2X communication. Specifically, we propose the SMART framework, which makes the following contributions:

• Using a software suite comprised of Blender [16], Blender Sensor Simulation (BlenSor) [17], and Wireless InSite (WI) [18], we create a multimodal DT setup where synthetic data can be generated to effectively train models for beam selection in mmWave V2X environments. This DT consists of high-fidelity reproductions of image and LiDAR samples that closely resemble analogous samples collected in the real world (addresses *Challenge 1*).

• We implement an adapted version of the MAML framework, a ML approach known for its robust generalization across diverse tasks, using simulated images and LiDAR samples created within the DT for ML training. We demonstrate the effectiveness of the models derived from meta-learning by validating them on real-world data, analyzing their cross-domain and inter-task performance across various scenarios in mmWave V2X communication. (addresses *Challenge 2a*).

• We independently conduct meta-learning based domain adaptation on image and LiDAR data samples to assist mmWave beam selection. We provide insights into the unique domain shifts experienced by each modality and their distinct impacts on Sim2Real adaptation. Furthermore, we highlight the importance of structured data representation and domain-informed DT modeling in fully harnessing MAML's generalization capabilities to tackle a challenging problem of beam selection (addresses *Challenge 2b*).

• For evaluation purposes, we show baseline beam selection results with our non-RF DT samples with empirical risk minimization (ERM), a non-meta-learning-based approach, then compare the proposed MAML performance with a TL-based domain adaptation method. Our results demonstrate that MAML-trained models, with minimal fine-tuning, achieve 14.04× accuracy improvements when transitioning from synthetic to real-world data, surpassing TL-based methods, which achieve an accuracy of 4.09×, highlighting the robustness of meta-learning to quickly adapt to unseen environments with minimal data and compute.

• We will release the data and software code to generate such multimodal DT and the ML codes for TL and MAML implementations that will allow the community to independently validate results and design their own DT for accelerating research in this emerging area.

Organization: The rest of this paper is structured as follows: Section II discusses the background and related works. Section III presents the SMART problem statement and its framework. Section IV describes the design of the SMART framework along with DT creation. Section V provides an overview of the datasets used for evaluation. Section VI details the experimental setup, evaluation process, and comparisons with competing methods. Finally, Section VII concludes the paper with a discussion, and Section VIII outlines potential future work.

II. Preliminaries and Related Work

As this work touches on a number of topics, we select a few relevant areas of discussion to sample the current state of the research landscape.

A. Mmwave V2X Beam Selection Overhead Reduction With Non-RF-Based Techniques

ML-based techniques leverage non-RF modalities to reduce overhead by enabling classification, object detection, or semantics extraction for faster beam selection. Sensors like LiDAR facilitate 3D environmental mapping to detect motion [19] or refine beam management using past trajectories [20]. Some approaches adopt a multimodal strategy, integrating LiDAR with GPS [19], [21], radar [22], or all three [23] to enhance user detection and processing efficiency.

B. DT Applications in Vehicular Networks

Digital Twins (DT) are gaining traction across various industries, including 5G networks [24], wireless communications [25], and autonomous driving [26], offering scalable synthetic data generation compared to real-world data collection. In vehicular networks, DT applications extend beyond wireless communications to physical routing [27], intersection management [28], environment monitoring and vehicle management [29]. They also aid in resource management [30] and communication reliability [31] in V2X networks. While DT-based beam management for V2X is still emerging [32], [33], this paper presents the first real-world application of a DT-enabled ML framework for mmWave beam selection.

C. Domain Transfer for Synthetic-to-Real Environments

Transferring models from synthetic to real-world applications remains a key challenge in ML, as even slight feature variations, such as image resolution differences between training and testing, can significantly affect performance [34]. Various techniques address this issue, but knowledge transfer between synthetic and real environments typically falls into two main categories—a) decreasing the size of the shift such that the source and target domains more closely resemble one another, or b) learning adaptive policies from synthetic data for real-world deployment. We further label these two approaches under TL and meta-learning, respectively.

Transfer learning (TL) leverages knowledge from models trained on a source domain to enhance performance on a target domain with differing data distributions. A specific application, domain adaptation, enables models to adjust to new contexts by retraining or fine-tuning on target data. While effective, domain adaptation methods often involve computational overhead and multi-step process [35], [36]. In this paper, TL serves as a state-of-the-art (SOTA) baseline for comparison.

Meta-learning, or the mechanism of learning to learn, focuses on developing models that generalize across tasks by optimizing their learning processes, enabling rapid adaptation with limited data. MAML [14], a gradient-based approach, exemplifies this by initializing models capable of fine-tuning effectively for new tasks. This few-shot approach utilizes limited data across a number of tasks *seen* during training to find an adaptive model that, when fine-tuned during testing time, performs well on new or *unseen* tasks, given that these tasks still fall under a common distribution.

Comparison of TL and Meta-learning for Domain Adaptation: Between these two, the former approach aligns more closely with traditional domain adaptation methods. Feature extraction methods using encoders [37] are common [32], [38], as are loss-centric methods, particularly using adversarial networks [39]. Our previous work on TL [35] that focuses on retraining weights also falls within this category. Though using these methods may provide high performance, there are oftentimes more computational requirements to consider, with some methods being multi-step processes that are invariably more time-consuming.

The latter approach is more closely associated with meta-learning, particularly gradient-based meta-learning policies, such as those shown in [40], [41]. These approaches tend to be more task-oriented, reducing either time complexity or end-to-end computation time. Though the base framework that these papers construct their approaches around, MAML [14], has been optimized in areas such as convergence [42] and task distribution [43], the fundamental area of meta-learning applied to DT is still relatively unexplored, especially when LiDAR data is considered.

Limitations in the State-of-the-art: Though DT applications in V2X scenarios are promising, DT utilization with domain

application is still early in development, let alone DT usage itself. To our knowledge, there are no prior works based on applying meta-learning on synthetic-to-real-world domain adaptation for mmWave beam selection, and our previous works focused only on domain adaptation within the real world (i.e., training and testing a model on real-world data). We propose the SMART framework as an initial solution to this deployment idea.

III. SMART Problem Statement

In this section, we detail the two fundamental problems that we seek to address, namely, beam selection via ML and DT-based beam selection, and list the aspects of the SMART framework that address these challenges. We summarize the notations in Table I.

A. Beam Selection With an ML-Based Approach

To avoid the costly exhaustive search overhead in traditional beam selection, one possible solution uses deep learning (DL) models that predict the best beam using non-RF sensor data captured from active sensors integrated in the vehicle [10]. In this method, a percentage (denoted by $\mathtt{p}$) of the dataset is available as a training set prior to deployment to $\mathtt{L}$ different scenarios $\{\mathtt{E}_l\}_{l=1}^{\mathtt{L}}$. Suppose the overall dataset of non-RF samples along with the RF ground-truth is represented as: $\mathbb{D}_{\mathtt{total}} = \{\{X_{l,j}^{\mathtt{I}}, Y_{l,j}\}_{j=1}^{|\mathtt{E}_l|}\}_{l=1}^{\mathtt{L}}$, where $X_{l,j}^{\mathtt{I}}$ is the j^{th} sample and $Y_{l,j} \in \{0,1\}^{\mathcal{B}}$ is the corresponding label for the l^{th} scenario, $\mathcal{B}$ is the total number of possible beams, and $|\mathtt{E}_l|$ is the total of samples for the l^{th} scenario $\mathtt{E}_l$. Now, the training dataset $\mathbb{D}_{\mathtt{train}}$ is generated by randomly selecting $\mathtt{p}\%$ of samples and labels from the $\mathbb{D}_{\mathtt{total}}$ in such a way that at least one sample from each $\mathtt{L}$ scenario is present, both within $\mathbb{D}_{\mathtt{train}}$. The test dataset is then denoted as the $\mathbb{D}_{\mathtt{test}}$ dataset. Hence, $\{\mathtt{E}_l\}_{l=1}^{\mathtt{L}} \in \mathbb{D}_{\mathtt{train}}$, $\{\mathtt{E}_l\}_{l=1}^{\mathtt{L}} \in \mathbb{D}_{\mathtt{test}}$, $\mathbb{D}_{\mathtt{train}} = \mathtt{p} \times \mathbb{D}_{\mathtt{total}}$, and $\mathbb{D}_{\mathtt{test}} = \mathbb{D}_{\mathtt{total}} - \mathbb{D}_{\mathtt{train}}$.

The learning model $f_{\theta^{\mathtt{n}}}(.)$ is trained on $\mathbb{D}_{\mathtt{train}}$, where $f_{\theta^{\mathtt{n}}}(.)$ is a function parameterized by $\theta^{\mathtt{n}}$, i.e., a neural network with weights $\theta^{\mathtt{n}}$. The empirical loss of the model parameters $\theta^{\mathtt{n}}$ on the training dataset $\mathbb{D}_{\mathtt{train}}$ is generated following [15]. The DL training approach finds a model that minimizes the loss across all training samples by solving: $\min_{\theta^{\mathtt{n}}} \ell(\theta^{\mathtt{n}})$ over multiple training epochs. After the model training, the best beam is predicted as:

$$\hat{\mathbb{Y}}_{l,j} = f_{\theta^{\mathtt{n}}}(\mathbb{X}_{l,j}^{\mathtt{I}}), \tag{1}$$

where $\mathbb{X}_{l,j}^{\mathtt{I}}$ is j^{th} sample from l^{th} scenario and $\mathbb{X}_{l,j}^{\mathtt{I}} \in \mathbb{D}_{\mathtt{test}}$, respectively.

Challenges: In practical situations, it is not guaranteed that the training data will come from all possible scenarios, thus, introducing some "unseen scenarios" [14] during test time. Therefore, the trained model $f_{\theta^{\mathtt{n}}}(.)$ will generate unreliable predictions when presented with samples from these scenarios. Moreover, the challenges mentioned in Section I-C motivate us to consider the scenarios generated from a synthetic environment as a *training scenario* or *seen scenario* and the scenarios collected from the real world as *test scenario* or *unseen scenario*.

TABLE I
Notation Summary

Notation	Description
$(X_{i,j}^{I}, Y_{i,j})$	Input sample and corresponding beam selection label
B	Number of beam selection classes (one-hot labels)
Tx_{loc}	Transmitter location
Rx_{loc}	Receiver trajectory locations
δ_x, δ_y	Step increments for trajectory sampling
$S = \{C_{\text{front}}, C_{\text{side}}, L_{\text{top}}\}$	Multimodal sensor set (cameras and LiDAR)
$\mathcal{T}_{i,k}^{S,\text{sup}}, \mathcal{T}_{i,k}^{S,\text{tar}}$	Support and target sets for training tasks
$\mathcal{T}_{i,k}^{U,\text{sup}}, \mathcal{T}_{i,k}^{U,\text{tar}}$	Support and target sets for unseen testing tasks
M_i	Number of tasks in scenario i
N	Total number of training samples across all scenarios
$\mathcal{L}(\theta, \mathcal{S}_i)$	Empirical loss function for training scenario S_i
$\hat{\theta}_{i,k}$	Adapted model parameters after meta-learning
$L_{\text{test}}^{\text{adapt}}$	Loss function evaluated on unseen real-world tasks
B_{opt}	Optimal beam selection for maximum received power P_{R_x}
$\mathcal{M}$	Metadata dictionary storing key information about obstacles, sensors, and trajectory
$Rx = \{Rx_1, Rx_2, ..., Rx_N\}$	Set of receiver positions along the vehicle's trajectory
$\hat{y}_{D_{\text{test}}} = f_{\theta_{\text{DT}}}(x_{D_{\text{test}}})$	Prediction from the trained model on real-world test data

Synthetic and Real Scenario Definition: In this case, we consider the $\mathtt{Q}$ different scenarios $\{\mathtt{S}_l\}_{l=1}^{\mathtt{Q}}$ collected from the synthetic environment as the training set. However, during testing time, the $\mathtt{R}$ different scenarios $\{\mathtt{U}_l\}_{l=1}^{\mathtt{R}}$ are from the real world. Hence, the characteristics of the training and test datasets are represented as: $\{\mathtt{S}_l\}_{l=1}^{\mathtt{Q}} \in \mathbb{D}_{\mathtt{train}}$, $\{\mathtt{S}_l\}_{l=1}^{\mathtt{Q}} \notin \mathbb{D}_{\mathtt{test}}$, $\{\mathtt{U}_l\}_{l=1}^{\mathtt{R}} \notin \mathbb{D}_{\mathtt{train}}$, and $\{\mathtt{U}_l\}_{l=1}^{\mathtt{R}} \in \mathbb{D}_{\mathtt{test}}$.

B. Digital Twin-Based Beam Selection

Suppose we have a real-world environment $\mathcal{R}$ where we need to deploy a beam selection module that is pre-trained on the data collected from a DT. We generate a representation $\mathcal{DT}$ of $\mathcal{R}$ and collect $\mathbb{D}_{\mathtt{train}}$ scenarios from $\mathcal{DT}$. The objective is to generate

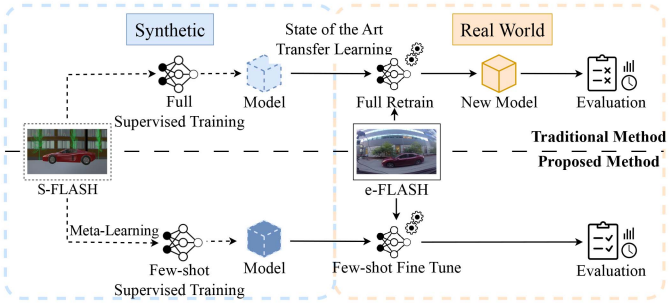


Fig. 1. The proposed methodology based on meta-learning outperforms the traditional transfer-learning-based method in terms of accuracy, the amount of data used for training, and end-to-end computation time.

a neural network model in which $f_{\theta^{\mathcal{DT}}}(.)$ will be trained on the collected training data from $\{\mathtt{S}_l\}_{l=1}^{\mathtt{Q}}$ seen scenarios. We also have a test dataset $\mathbb{D}_{\mathtt{test}}$ consisting of $\{\mathtt{U}_l\}_{l=1}^{\mathtt{R}}$ unseen scenarios. Overall, we want to generate a trained model $f_{\theta^{\mathcal{DT}}}(.)$ which will be trained on the training data $\mathbb{D}_{\mathtt{train}}$, where $x_{\mathbb{D}_{\mathtt{train}}}$ and $y_{\mathbb{D}_{\mathtt{train}}}$ represent the data and ground truth of each sample within the dataset. The same notation applies for the $\mathbb{D}_{\mathtt{test}}$. Overall, the problem is formulated as:

$$\begin{aligned} &\min\ (\hat{y}_{\mathbb{D}_{\mathtt{test}}} - y_{\mathbb{D}_{\mathtt{test}}}) \\ &\text{where } \hat{y}_{\mathbb{D}_{\mathtt{test}}} = f_{\theta^{\mathcal{DT}}}(x_{\mathbb{D}_{\mathtt{test}}}) \\ &\qquad f_{\theta^{\mathcal{DT}}}(.) \text{ is trained on } \mathbb{D}_{\mathtt{train}} \end{aligned} \tag{2}$$

C. SMART Framework

SMART solves the above problems by proposing a learning paradigm that generalizes on the synthetic dataset by creating a model that is adapted with a few real-world samples during real-world deployment by using a few fine-tuning steps. The overall system function is given in Fig. 1.

- *DT Creation:* First, we create a DT environment that replicates the real-world experiment setup (discussed in Section IV-A).
- *Synthetic Data Collection in DT:* We collect a comprehensive synthetic dataset in the generated DT setup (see Section IV-B).
- *Meta-learning-based Training on the Synthetic Dataset:* We generate trained models that are generalized over different data categories within the synthetic dataset.
- *Domain Adaptation using Meta-learning:* We evaluate the trained model on the real-world data after performing a few fine-tuning steps on a few samples. Algorithm 1 outlines the data collection and synchronization process used to prepare the dataset for training. Further the results of the evaluation is compared directly to the performance of TL and ERM with the same datasets.

IV. SMART Framework Design

In this section, we elaborate on the individual solutions proposed in Section III-C; namely, the creation of the DT, the data collection process while using the DT, and the domain adaptation methods used to adapt the ML models trained on the synthetic data from the DT to the real world in the SMART framework.

A. DT Creation

The creation of a DT involves an accurate replication of the real-world environment in simulation software. This process begins with the generation of an experimentation scenario by using a detailed, precise map, and defining the area of interest using appropriate geographical coordinates. We initialize our DT as $\mathcal{DT} : f_{twin}(map, O) \rightarrow [0, 1]^{\mathcal{B}}$, which signifies our twin is a function of two components, map representing the coarse terrain of the experiment area (in our case, imported from OpenStreetMaps (OSM) [44]) and O incorporates the finer details like trees, vehicles, sensors, and radio devices. To create the twin world, the map is a function of the form $map = f_{box}(R_{tl}, R_{br})$ that requires the user to specify a rectangular region covering the experiment area by specifying the geographical coordinates (latitude and longitude) for the top left and bottom right corner of the bounding box, R_{tl} and R_{br} respectively. Additionally, the surrounding objects are characterized by bounding boxes in the $\mathcal{DT}$, such that $O_{box} = \{(x_k, y_k, z_k, l, w, h)\}_{k=1}^{K}$, where K signifies the number of objects that are present in the $\mathcal{DT}$. Furthermore, (x_k, y_k, z_k) are the centroid coordinates of the k^{th} object with dimensions (l, w, h) corresponding to the length, width, and height respectively. To further elucidate the development of the $\mathcal{DT}$, we explain it in two components - a multimodal sensor component to generate the synthetic input data, and a wireless component to collect the corresponding ground truth labels.

Multimodal Component: The multimodal sensor component of $\mathcal{DT}$ focuses on replicating the position and visual sensors integrated into an autonomous car. These include coordinates, cameras, and LiDAR, which are integral sensors used in self-driving cars, to enable perception and navigation. We leverage these sensors to obtain contextual information about the environment to understand the relative positions and orientations of the radio devices, the transmitter (Tx) and receiver (Rx), along with the presence of any obstacles in the line of sight (LOS) path between the devices. We implement this by placing virtual camera objects C_{front}, to capture the front view, and C_{side}, to focus on the right side view of the Rx, which is on top of the car. We appropriately configure the focal length and field of view of these virtual cameras according to the real-world camera configurations. Correspondingly, we place a virtual LiDAR object L_{top} at the same locations as the Rx to capture a 360-degree view of the vehicular environment. We place these virtual camera and LiDAR sensor objects at a height h_{car} to emulate these sensors being placed on top of an autonomous car and collectively address the multi-modal sensor objects as S such that $\{C_{front}, C_{side}, L_{top}\} \in S$.

Wireless Component: We import the same coarse environment landscape from the animation software to the wireless simulation tool to collect corresponding optimal beam labels for training our ML-based decision framework. Here, we utilize a ray-tracing tool (in our case, WI) to accurately model the propagation of electromagnetic waves through the V2X environment. We

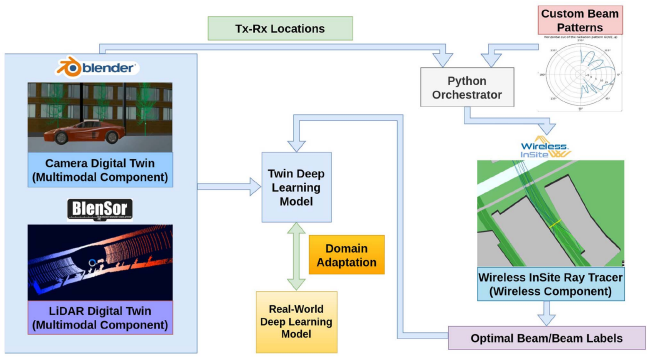


Fig. 2. Orchestration process between the DT components and the DL models within the SMART framework to collect and process the multimodal sensor data and ground truth beam labels.

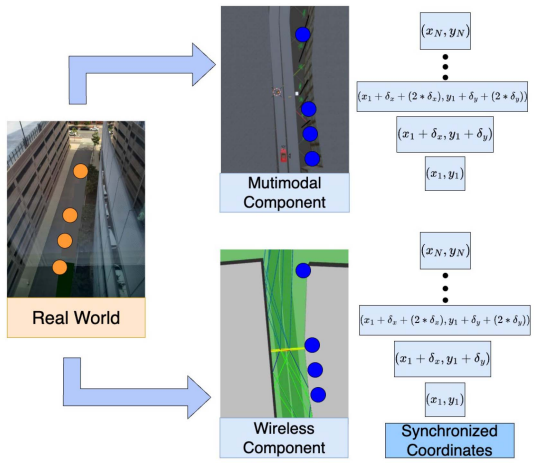


Fig. 3. Corresponding sampling locations (blue dots) and their coordinates within the multimodal and wireless components, reflective of the real-world GPS coordinates (orange dots). The sampling frequency is not to scale.

follow Extensible 3D [45] for modeling the mmWave propagation path that considers the phase information of the rays. It accurately models reflections, transmissions and diffractions along with frequency dependent atmospheric absorption for conducting realistic simulations. The finer environment details include the obstacles O, along with the placements of the radio devices at appropriate locations obtained from the multimodal component.

In our use case, there is one static transmitter Tx, whereas the receiver is integrated into the autonomous car that is moving along a linear trajectory. To emulate this, we consider multiple receiver instances $R = \{Rx_n\}_{n=1}^{N}$ in both components along the vehicle's trajectory that are linked between the two components by coordinates, emulating a sampling scheme similar to the synchronized samples in [10] and [11].

B. Synthetic Data Collection in DT

Since our DT consists of two components, we need to develop a software orchestration module to coordinate the input data generation and ground truth collection between these two components. This section describes the overall orchestrator design, as shown in Fig. 2 and the structured workflow of orchestration process is detailed in Algorithm 1, followed by the data collection process in the two components.

1) Orchestration: Our data collection scenario involves a static roadside transmitter placed at Tx_{loc} in both components and a moving vehicle, accompanied by a receiver radio device. As outlined in Algorithm 1, we model the moving Rx along the vehicle trajectory by uniformly sampling it at a particular frequency. We consider a trajectory of length d with N sample points. Consequently, we obtain the sampling locations, as shown in Fig. 3, as $Rx_{loc} = \{(x_1, y_1), (x_1 + \delta_x, y_1 + \delta_y), (x_1 + \delta_x + (2 * \delta_x), y_1 + (2 * \delta_y)) \ldots (x_N, y_N)\}_{loc=1}^{N}$. Here, δ_x and δ_y represent the step increment obtained by dividing the lengths along each axis by the sampling period. The incremental step size can be computed as $\delta_x = (x_N - x_1) / N$ and $\delta_y = (y_N - y_1) / N$. We place our sensor objects S and receivers R along these sampled locations in their respective twin components to collect the data. Static obstacles are placed using the centroid points and dimensions of the objects described in O, whereas the motion of dynamic obstacles additionally requires emulating the motion in a similar way as the moving receiver devices. The static objects simply use the bounding box information for placement, represented by $O_S = \{O_{box}\}$. On the other hand, dynamic objects require some additional information to recreate the motion, which can be represented as $O_M = \{O_{box}, O_{start}, O_{end}, M\}$, where O_{start} is the starting frame number, O_{end} is the ending frame number and M is the number of samples collected along this path. To exchange the metadata between the multimodal and wireless components, we create a metadata dictionary with the structure $M = \{O_S, O_M, Tx_{loc}, Rx_1, Rx_N, N\}$, where the latter three components are used to emulate the moving receiver device.

2) Multimodal Data Collection in DT: We collect the visual data in terms of image samples and LiDAR pointclouds by moving the corresponding sensor objects C_{front}, C_{side}, and L_{top} along the trajectory of the car. We synchronize the frame rate of the cameras C_{front}, C_{side} with the sampling frequency of the LiDAR L_{top} to ensure uniform data collection. The animation software provides a key-frame functionality to define the starting and ending points for the motion of the receiver car and surrounding objects, as described in the previous section. This facilitates smooth motion and continuous rendering of data samples, which are saved in an automated fashion. To systematically capture the multimodal sensor data, we define the following functions to extract camera images and LiDAR point clouds at each receiver position along the trajectory. The function `CAPTURE_FRAME(camera_object, location)` is used to collect image frames from the front and side cameras, where:

$$X^I[i][j][\text{front}] \leftarrow \text{CAPTURE\_FRAME}(C_{\text{front}}, Rx_{[t]})$$

$$X^I[i][j][\text{side}] \leftarrow \text{CAPTURE\_FRAME}(C_{\text{side}}, Rx_{[t]})$$

Similarly, the function `CAPTURE_POINTS(lidar_object, location)` records LiDAR point clouds from the

Algorithm 1: SMART Orchestration Algorithm.

Input: Transmitter location Tx_{loc},
Receiver trajectory $Rx = \{Rx_1, Rx_2, \ldots, Rx_N\}$,
Static obstacles O_S,
Dynamic obstacles O_M,
Sensor set $S = \{C_{\text{front}}, C_{\text{side}}, L_{\text{top}}\}$,
Step increments δ_x, δ_y, Sampling frequency F_s,
Total sample points N

Output: Metadata dictionary $\mathcal{M}$ storing key sampling data

Compute step increments:
$\delta_x \leftarrow \frac{Rx_N - Rx_1}{N}, \quad \delta_y \leftarrow \frac{Rx_N - Rx_1}{N}$

Initialize metadata dictionary:
$\mathcal{M} \leftarrow \{O_S, O_M, Tx_{loc}, Rx_1, Rx_N, N, \text{samples} \leftarrow [\,]\}$

where [ ] represents sequentially growing list storing structured sensor data at each receiver location.

Position static objects: $O_S \leftarrow \{O_{\text{box}}\}$

for $i \leftarrow 1$ ***to*** N **do**

  Compute receiver location:
  $Rx_i \leftarrow (Rx_1 + i \cdot \delta_x, Rx_1 + i \cdot \delta_y)$

  Position dynamic objects:
  $O_M \leftarrow \{O_{\text{box}}, O_{\text{start}}, O_{\text{end}}, M_i\}$

  Position sensors at Rx_i: $S \leftarrow \{C_{\text{front}}, C_{\text{side}}, L_{\text{top}}\}$

  Update metadata dictionary:
  $\mathcal{M}[\text{samples}].\text{append}(\{Rx_i, O_S, O_M, S\})$

end

Synchronize sensor frame rates:SYNC_FRAME_RATE(S)

for $t \leftarrow 1$ ***to*** N **do**

  $X^I[i][j][\text{front}] \leftarrow \text{CAPTURE\_FRAME}(C_{\text{front}}, Rx_{[t]})$

  $X^I[i][j][\text{side}] \leftarrow \text{CAPTURE\_FRAME}(C_{\text{side}}, Rx_{[t]})$

  $X^I[i][j][\text{lidar}] \leftarrow \text{CAPTURE\_POINTS}(L_{\text{top}}, Rx_{[t]})$

  Scanning all beam patterns:
  $Y[i][j] \leftarrow \text{SCAN\_BEAM\_PATTERNS}(Rx_{[t]})$

  Optimal beam selection:
  $B_{\text{opt}} \leftarrow \text{OPT\_BEAM\_SELECTION}(X^I[i][j], Y[i][j])$

  $\text{data}_t \leftarrow \{\text{“image\_front”} : X^I[i][j][\text{front}], \text{“image\_side”} : X^I[i][j][\text{side}], \text{“lidar”} : X^I[i][j][\text{lidar}], \text{“beam\_selection”} : B_{\text{opt}}\}$

  $\mathcal{M}[\text{samples}][t][\text{“}data\text{''}] \leftarrow \text{data}_t$

  SAVE_DATA($\mathcal{M}[\text{samples}][t]$)

end

return $\mathcal{M}$

top-mounted LiDAR sensor:

$$X^I[i][j][\text{lidar}] \leftarrow \text{CAPTURE\_POINTS}(L_{\text{top}}, Rx_{[t]})$$

These functions ensure synchronized data collection by aligning the camera frame rate with the LiDAR sampling frequency, facilitating structured dataset generation for downstream model training and analysis.

3) Optimal Beam Label in DT: To automate the placement of radio devices at the sampled locations and perform exhaustive beam search scans along the defined trajectory, we use a Python orchestrator. Our framework sequentially activates each Rx_n along the vehicle's trajectory utilizing the positioning information from the metadata dictionary $\mathcal{M}$ and performs a multi-path ray-tracing scan for all available beam patterns using the function

$$Y[i][j] \leftarrow \text{SCAN\_BEAM\_PATTERNS}(Rx_{[t]})$$

For each of the beam patterns, we record the received power P_{Rx}. To identify the optimal beam for each frame, we introduce the $\texttt{OPT\_BEAM\_SELECTION}(X^I[i][j], Y[i][j])$ function, which identifies the beam pattern with maximum P_{Rx} beam pattern and assigns it as the ground truth for that frame.

4) Data Preprocessing: The images can be readily supplied to the convolutional neural network (CNN)-based DL models such as the ones described in [10], [35]. Conversely, the LiDAR pointclouds require some additional preprocessing to make them compatible with the SMART framework. LiDAR pointclouds consist of an unstructured set of points in 3D space. However, the permutation invariance of these points poses a challenge when leveraging CNN architectures. Unlike CNNs, which process ordered grid structures like image pixels, rearranging the order of LiDAR points does not alter the represented scene. Therefore, we need to convert LiDAR pointclouds into structured, ordered grid representations of the 3D space through a 3D quantized cuboid structure. Each unit within this structure is called a *voxel*, which stores the occupancy information of point clouds. Voxel values are set to 1 if they contain at least one point, indicating the presence of obstacles in that specific region. Conversely, unoccupied voxels are assigned a value of 0, while the voxels at the current Rx and Tx positions are labeled -2 and -1, respectively. Thereafter, we select a 20 m radius around each the Rx car and quantize each axis to a (20, 20, 4) grid, with each voxel set to size (2, 2, 1).

C. Domain Adaptation Using MAML

As noted in Section I, we use two tools in the SMART framework for domain adaptation: one which is based on TL, and one which is based on meta-learning, namely, an adaptation of the MAML algorithm [14]. We explain how we implement TL in [35] and meta-learning in [15], but restate core meta-learning concepts here in this section for framework completeness. Implementation changes to our TL framework are explained briefly in Section VI.

Recall that we have $\mathtt{Q}$ different scenarios $\{\mathtt{S}_l\}_{l=1}^{\mathtt{Q}}$ collected from the synthetic environment that serve as the training set. However, at test time, the $\mathtt{R}$ different scenarios $\{\mathtt{U}_l\}_{l=1}^{\mathtt{R}}$ come from the real-world. We can represent the characteristics of the training and test datasets as $\{\mathtt{S}_l\}_{l=1}^{\mathtt{Q}} \in \mathbb{D}_{\mathtt{train}}$, $\{\mathtt{S}_l\}_{l=1}^{\mathtt{Q}} \notin \mathbb{D}_{\mathtt{test}}$, $\{\mathtt{U}_l\}_{l=1}^{\mathtt{R}} \notin \mathbb{D}_{\mathtt{train}}$, and $\{\mathtt{U}_l\}_{l=1}^{\mathtt{R}} \in \mathbb{D}_{\mathtt{test}}$.

Prior to model deployment, we access the training set of labeled samples from $\mathtt{Q}$ different scenarios from the synthetic environment. Training data corresponding to the i^{th} scenario is given by $\mathtt{S}_i := \{(X_{i,j}^{\mathtt{I}}, Y_{i,j})\}_{j \in n_i}$, where each $X_{i,j}^{\mathtt{I}} \in \mathbb{R}^d$ is a sample, $Y_{i,j} \in \{0,1\}^{\mathcal{B}}$ is its corresponding label, and n_i is the number of samples from the scenario $\mathtt{S}_i$. The learning model is a function $f_\theta : \mathbb{R}^d \mapsto \mathbb{R}^{\mathcal{B}}$ parameterized by $\theta \in \mathbb{R}^D$, e.g., f_θ may be a neural network with weights θ. The empirical loss of the model parameters θ on a dataset $\mathcal{S}_i$ is defined as $\mathcal{L}(\theta; \mathcal{S}_i) := \frac{1}{n_i} \sum_{j=1}^{n_i} [\ell(f_\theta(X_{i,j}^{\mathtt{I}}), Y_{i,j})]$, where $\ell : \mathbb{R}^{\mathcal{B}} \times \{0,1\}^{\mathcal{B}} \to \mathbb{R}^+$ is a

cross-entropy loss function measuring the discrepancy between predicted and true labels.

The standard ML training approach is to find a model that minimizes the average loss across all of the training samples, namely, Empirical Risk Minimization (ERM). Specifically, ERM solves $\min_{\theta\in\mathbb{R}^D} L(\theta) := \frac{1}{N}\sum_{i=1}^{\mathtt{Q}} n_i \mathcal{L}(\theta, \mathcal{S}_i)$, where $N = \sum_{i=1}^{\mathtt{Q}} n_i$. One can run a variety of easy-to-implement gradient-based algorithms to optimize this objective—for instance, stochastic gradient descent (SGD). While this approach is natural for finding high-performing models during training, it is not well-suited to find models that can adapt samples from unseen scenarios when deployed, as we show in [15] and Section VI.

Beam Selection in Real-world Unseen Scenarios: In practical applications such as V2X networks, an ML framework does not typically have enough data or computational budget to perform full supervised learning in the new scenario; rather, it is only provided with a few labeled samples and must yield a prediction within a time frame on the scale of seconds, as a vehicle may only be within communication range of a base station for a few seconds.

We consider each of these adaptation opportunities as a "task". That is, a task consists of a small number of support samples that can be used for adapting the model, along with target samples for evaluating the adapted model. Each task has data that is a subset of the dataset for a particular scenario. Specifically, the k^{th} task from scenario $\mathtt{S}_i$ is defined by the pair of datasets $(\mathcal{T}_{i,k}^{\mathtt{S},sup}, \mathcal{T}_{i,k}^{\mathtt{S},tar})$, where $\mathcal{T}_{i,k}^{\mathtt{S},sup}$ contains the support samples, $\mathcal{T}_{i,k}^{tar}$ contains the target samples, $\mathcal{T}_{i,k}^{\mathtt{S},sup} \cup \mathcal{T}_{i,k}^{\mathtt{S},tar} \subseteq \mathcal{S}_i$ and $\mathcal{T}_{i,k}^{\mathtt{S},sup} \cap \mathcal{T}_{i,k}^{\mathtt{S},tar} = \emptyset$. We let $m_1 := |\mathcal{T}_{i,k}^{\mathtt{S},sup}|$ and $m_2 := |\mathcal{T}_{i,k}^{\mathtt{S},tar}|$, for all tasks i, k, and let M_i denote the number of tasks for scenario i.

Next, we denote the tasks for $\mathtt{R}$ real-world unseen scenarios as as: $(\mathcal{T}_{i,k}^{\mathtt{U},sup}, \mathcal{T}_{i,k}^{\mathtt{U},tar})$, following the same notation. We suppose that the task-specific adaptation procedure is τ steps of gradient descent (GD) with step size α using the support samples in the task's support set, where τ is small. Let $\hat{\theta}_{i,k} := \text{GD}(\hat{\theta}, \mathcal{T}_{i,k}^{\mathtt{U},sup}, \alpha, \tau)$ denote the result of this adaptation procedure starting from $\hat{\theta}$. Ultimately, we aim to find a $\hat{\theta}$ such that the loss of $\hat{\theta}_{i,k}$ is small on average across tasks from unseen scenarios, i.e., our performance metric is: $L_{\text{adapt}}^{\text{test}}(\hat{\theta}) := \frac{1}{\mathtt{R}}\sum_{i=1}^{\mathtt{R}}\sum_{k=1}^{M_i} \mathcal{L}(\hat{\theta}_{i,k}; \mathcal{T}_{i,k}^{\mathtt{U},tar})$. As mentioned previously, models found by standard ERM are not well-suited to perform well on the adaptive metric $L_{\text{adapt}}^{\text{test}}(\cdot)$. Thus, we leverage the unseen scenarios in our approach, described next.

D. MAML for Beam Selection in Unseen Scenarios

In order to find models that perform well after adaptation, i.e., achieve a small $L^{adapt}(\cdot)$, we utilize a MAML-based approach. MAML aims to find an adaptable initialization for task-specific SGD in multi-task settings. To do so, MAML executes an episodic training procedure, referred to as meta-training, in which each episode consists of first adapting the current initialization to the corresponding task, then improving the initialization based on the performance of the adapted model on the same task. In particular, MAML aims to solve the following objective in our setting:

$$\min_{\theta\in\mathbb{R}^D} L_{\text{adapt}}^{\text{train}} := \frac{1}{\mathtt{Q}}\sum_{i=1}^{\mathtt{Q}}\sum_{k=1}^{M_i} \mathcal{L}(\theta_{i,k}; \mathcal{T}_{i,k}^{\mathtt{S},tar}) \tag{3}$$

where $\theta_{i,k} := \text{GD}(\theta, \mathcal{T}_{i,k}^{\mathtt{S},sup}, \alpha, \tau)$. This process is depicted in Fig. 4. In words, we aim to find an initial model $\hat{\theta}$ that performs well after τ GD steps using the samples $\mathcal{T}_i^{sup}$, on average across all scenarios indexed by i. To solve (3), we execute the MAML algorithm, which is equivalent to performing SGD on (3). This framework is displayed in Fig. 4. Note that the MAML training objective $L_{\text{adapt}}^{\text{train}}$ is the analogue of $L_{\text{adapt}}^{\text{test}}$ on the training data. Indeed, our evaluation procedure on tasks from unseen scenarios exactly corresponds to what MAML refers to as the meta-testing phase. Overall, the training objectives over $L_{\text{adapt}}^{\text{train}}$ and $L_{\text{adapt}}^{\text{test}}$ ensures a solution for (2).

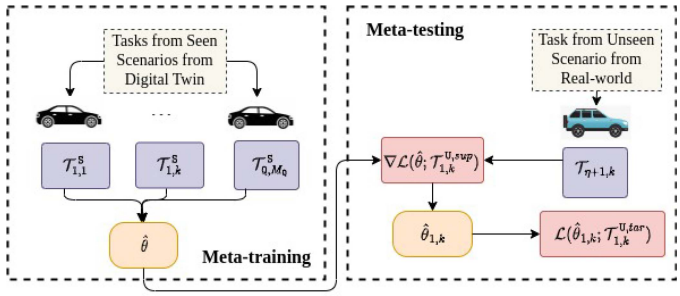
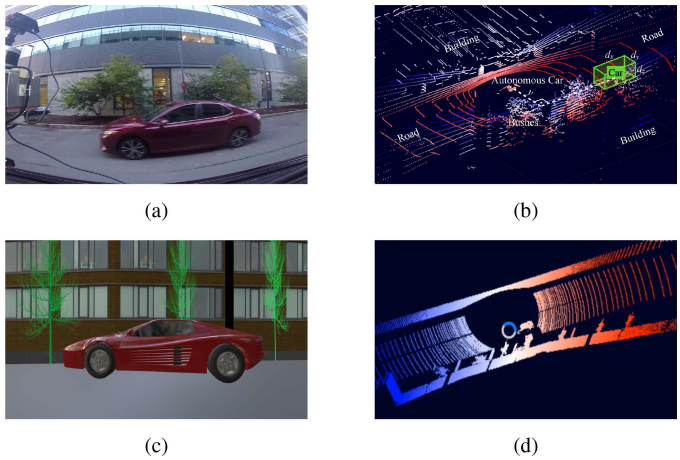


Fig. 4. Proposed MAML-based framework for adapting to unseen scenarios for beam selection. $\hat{\theta}$ is the model after meta-training, and $\hat{\theta}_{1,k}$ is generated after fine-tuning, during meta-testing.



(a) (b) (c) (d)

Fig. 5. Select samples taken from the e-FLASH and S-FLASH dataset with a camera image and LiDAR pointcloud side-view pairing from the real world (a), (b) and a camera image and LiDAR pointcloud front-view pairing from the synthetic environment (c), (d), respectively. The images for the LiDAR captures were visualized in MATLAB LiDAR Toolbox [46].

V. Datasets

In this section, we detail the two datasets that we use for our evaluation, the real-world e-FLASH dataset, and its DT counterpart, S-FLASH.

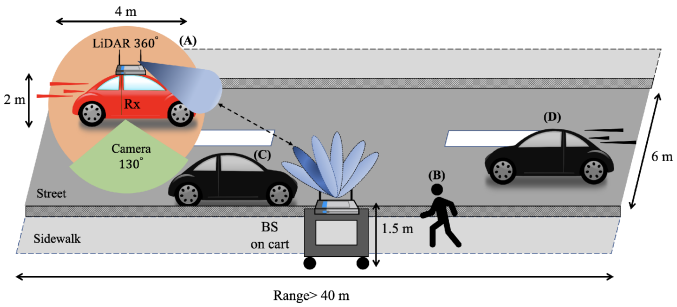


Fig. 6. Schematics of data collection environment for: A) Category 1: LOS passing, B) Category 2: NLOS pedestrian, C) Category 3: NLOS static car, D) Category 4: NLOS moving car.

TABLE II
SUMMARY OF DIFFERENT CATEGORIES OF DATA GENERATION

Cat.	Lane	Featuring	Obstacle Scenarios	# Eps.	# Smpl.
1	Same Opposite	-	-	10	1900
2	Opposite	Pedestrian	Standing Walking right to left Walking left to right	30	5700
3	Opposite	Static car	In front	10	1900
4	Opposite	Moving car	Same lane Opposite lane	20	3800

A. Real-World Dataset: E-FLASH

The extended Federated Learning for Automated Selection of High-band mmWave Sectors (e-FLASH) dataset is a real-world multimodal dataset comprised of synchronized LiDAR, camera, and GPS data with ML applications that are primarily used to speed up beam selection in mmWave V2X networks. Though the finer details of the dataset can be found in [11], we briefly describe the dataset here: The e-FLASH dataset is an extension of the Federated Learning for Automated Selection of High-band mmWave Sectors (FLASH) dataset [10], and thus, is similarly structured in terms of having four categories with MAML-specific task quantities as follows: **1)***LOS with no obstacles (1 task),* **2)***Non-LOS (NLOS) with a pedestrian obstacle (3 tasks),* **3)***NLOS with a static car obstacle (1 task),* and **4)***NLOS with a moving car obstacle (2 tasks).* (see Fig. 6) with additional variations as shown in Table II. These categories consist of unique scenarios and multiple episodes per scenario with synchronized 16-channel LiDAR, 64-channel LiDAR, side-facing camera images (to the right with respect to the moving Rx vehicle), front-facing camera images (in the front of the Rx), and GPS samples intended to comprehensively represent commonly encountered LOS and NLOS V2X scenarios. Overall, e-FLASH contains 10853 samples ($\sim$ 22GB processed data) that are ready for use in mmWave V2X beam selection applications (see sample camera and LiDAR pairings in Fig. 5(a) and (b)).

B. Synthetic Dataset: S-FLASH

1) Tool Selection for DT Creation: For creating a high-fidelity DT, we selected Blender [16], Blender Sensor Simulation (BlenSor) [17], and Wireless InSite (WI) [18] over other alternatives like Sionna [49] due to their accuracy and robustness to create realistic emulation environments.

- Blender was selected to replicate the physical world due to its open-source nature and widespread use. It offers add-ons that facilitate seamless integration of geospatial data from OpenStreetMap and enables exporting scenes to Wireless InSite in various compatible formats. Moreover, Blender's advanced 3D modeling capabilities allow for the creation of synthetic images, making it suitable for simulating realistic environments.
- Blensor provides an open-source sensor simulation toolkit to obtain high-precision LiDAR point clouds, making it well-suited for accurately replicating real-world V2X scenarios.
- WI was chosen for conducting ray-tracing due to its ability to provide detailed environment modeling including high-fidelity terrain, material, and foliage effects, including in environments with dense vegetation. It accurately models frequency-dependent interactions and terrain impacts. While SionnaRT offers a GPU-accelerated differentiable ray tracer with smoother AI integration, it does not yet match WI's precision in modeling the detailed physics of the environment, making WI our preferred choice for creating a high-fidelity wireless DT for V2X beam selection.

By integrating Blender, BlenSor, and WI, our SMART framework ensures realistic synthetic data generation, supporting robust beam selection modeling across diverse real-world conditions.(addresses *Contribution 1*).

2) Components of S-FLASH: The Synthetic FLASH (S-FLASH) dataset is a virtual recreation of the e-FLASH dataset that captures a high-fidelity V2X scenario along a 2-lane urban road.

Multimodal Twin: We implement this following Section IV-A, creating a replica of the e-FLASH environment with the use of OSM and Blender, a 3D computer graphics software toolkit. We use an add-on service named Blender-OSM that allows users to select a rectangular region of interest based on geographical coordinates and import the associated landscape in Blender. The imported terrain includes coarse details, such as the surrounding buildings, roads, and walkways. Additionally, we insert finer details like surrounding bushes, trees, and suitable building textures and materials in the approximate location of their real-world counterparts, then utilize Blender functionalities to modify the visual properties for a quality representation of the virtual world and production of camera images that closely resemble their real-world counterparts. We place a box to play the role of the roadside Tx while capturing the visual data. Furthermore, we collect LiDAR data using a complementary open-source sensor simulation tool, BlenSor [50]. It implements a 64-channel Velodyne-64 LiDAR object to collect point cloud samples.

Wireless Twin: We leverage Wireless Insite (WI) to perform ray-tracing for collecting signal strengths and determining optimal beam patterns. The wireless system parameters are highlighted in Table III. We place a Tx at a height of 1.5 m to simulate the roadside BS and place the Rx at a height of 3.5 m (the height of the real-world car) at specific points along the vehicle's trajectory, with the Tx and Rx configured with compatible mmWave antenna patterns. On the Tx side, we scan

TABLE III
WIRELESS DT SYSTEM PARAMETERS

Materials	**Buildings: Concrete** **Road: Asphalt** **Foliage: Grass**
Antenna orientation	T_x and R_x facing opposite directions
Waveform	$f_c = 60\,GHz$, BW = $2.16\,GHz$
Tx power	$24\,dBm$ [47]
Tx and Rx pattern source	Talon antenna measurements [48]
Tx height	On a cart, $1.5\,m$
Rx height	On a car, $3.5\,m$
Noise power ($\mathcal{N}$)	$-100.99\,dBm$
Antenna sensitivity	$-250\,dBm$
Ray spacing ($\Delta\omega$)	$0.25°$
Number of allowed diffraction	1
Number of reflections	3
Scenarios	LOS and NLOS

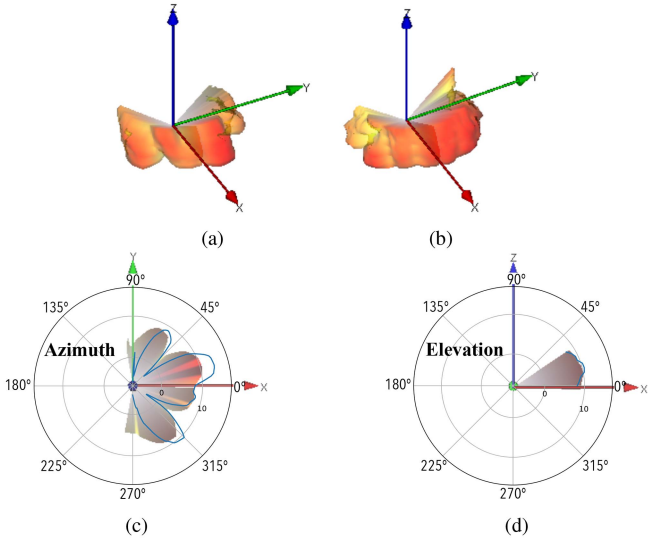
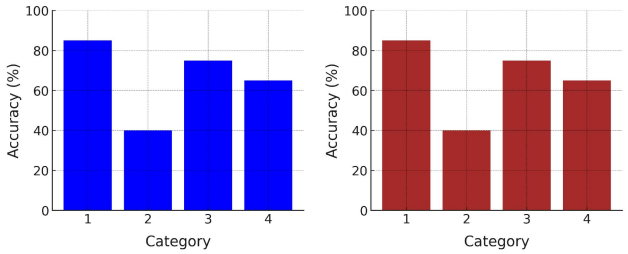


Fig. 7. Re-created antenna pattern examples in Wireless InSite: (a) Tx Antenna-24 (t_{24}) and (b) the Rx antenna. Antenna pattern comparisons between the Talon antenna patterns (solid blue line) and the re-created beams in WI (shaded area): (c) Azimuth, (d) Elevation.

the 34 available patterns [6], collecting the received powers for each scan. The beam patterns which comprise the ground truth for our problem come from a pre-defined codebook of 34 beams of the Talon AD7200 mmWave radio [26] used while collecting the real-world FLASH dataset. The beam pattern with maximum power is assigned as the optimal beam label for a particular sample captured. One example of the custom antenna patterns is shown in Fig. 7. The Talon manufacturers release the MATLAB files of the precise measurements of these 34 antenna patterns, with RSSI values for the entire range of azimuth and elevation values, which are pre-processed and imported in WI. In Fig. 7(c) and (d), we provide an example comparison between the Talon antenna patterns (solid blue line) and the re-created beams in WI (shaded area) for the 24^{th} beam (element t_{24}) in 2D azimuth ($\theta = 0°$) and elevation ($\phi = 0°$), respectively. The slight discrepancy comes from the fact that WI's user defined antenna patterns only accept sample points with integer increments (ϕ_δ, θ_δ). To maintain the full SNR values, we allocate the

TABLE IV
COMPARING TALON AND WI METRICS

	ϕ	ϕ_δ	θ	θ_δ	SNR
Talon [6]	$[-90°, 90°]$	$1.8°$	$[0°, 32.4°]$	$3.6°$	same
WI	$[-100°, 100°]$	$2°$	$[0°, 36°]$	$4°$	same



(a) Images (b) LiDAR

Fig. 8. Testing accuracies for TL models when trained and tested on only synthetic data, i.e., keeping training and testing within $\mathcal{DT}$.

experimental SNR values across the nearest azimuth and elevation regions, keeping $\phi = 0$ as the reference point. As a result, ϕ ranges from $-100°$ to $100°$ and θ spans from $0°$ to $36°$. The complete metrics for antenna pattern sample comparison is given in Table IV. Ultimately, S-FLASH contains 26,600 samples ($\sim$90GB processed data) across four categories analogous to the categories in e-FLASH that can support DT-based mmWave V2X beam selection.

VI. RESULTS

In this section, we evaluate the performance of our proposed SMART framework, with the help of mmWave beam selection task described in Section III-A. The model is trained to predict the optimal sector from a pre-defined codebook based on multimodal sensor inputs (i.e.,camera images and LiDAR point clouds). This prediction is critical for reducing the exhaustive beam search overhead in mmWave V2X communications. The accuracy results presented in Figs. 9 to 11 quantify the performance improvements achieved by our adapted meta-learning (MAML) approach compared to traditional methods, thereby demonstrating the effectiveness of our domain adaptation strategy in bridging the Sim2Real gap. We also validate our proposed SMART framework using the two datasets described in Section V by analyzing algorithm performance in a few select experiments. We note that we only use the side-view camera images and LiDAR samples from S-FLASH, as they provide the most amount of information [11]. We perform all experiments on an NVIDIA Tesla A100 GPU with PyTorch v1.10.0.

A. Competing Methods

As outlined in Section IV, we evaluate the SMART framework using the following approaches:

ERM: We use an Empirical Risk Minimization (ERM) algorithm with objectives presented in Sections IV-C and IV-D.

TL: The TL framework is adapted from the state-of-the-art domain adaptation technique presented in [35]. We make two

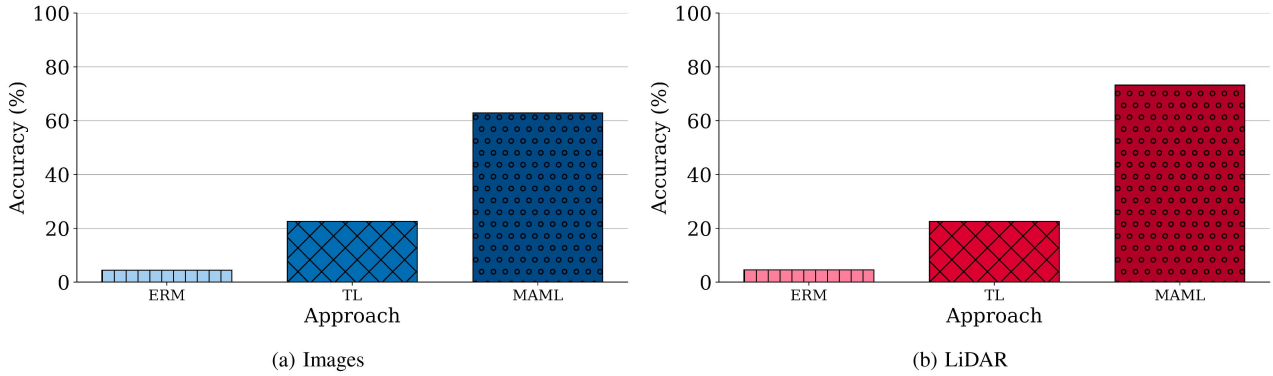


Fig. 9. Accuracy when training on all data from $\mathcal{DT}$ and testing on all data from $\mathcal{R}$ for each approach.

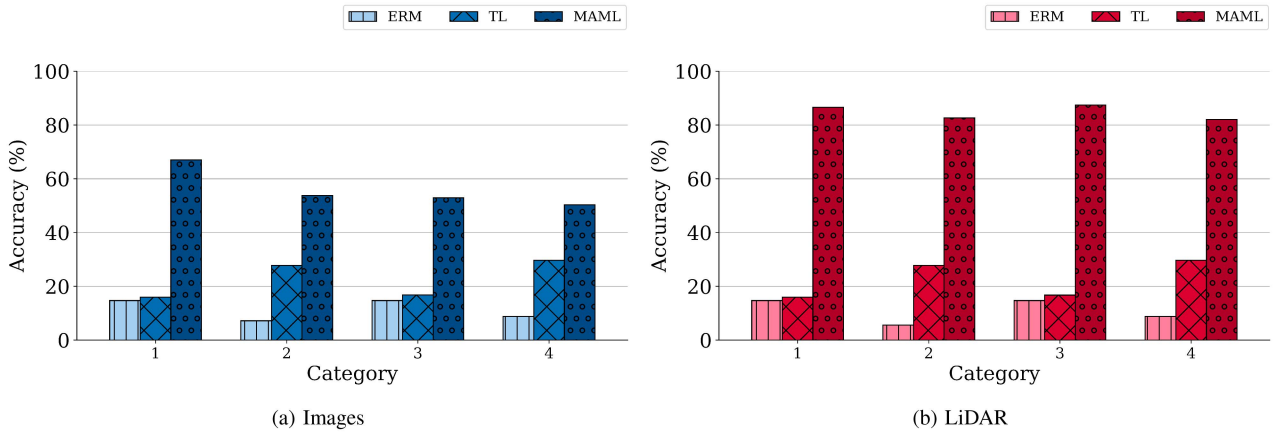


Fig. 10. Results for training on individual category data from $\mathcal{DT}$ and testing on analogous individual category data from $\mathcal{R}$.

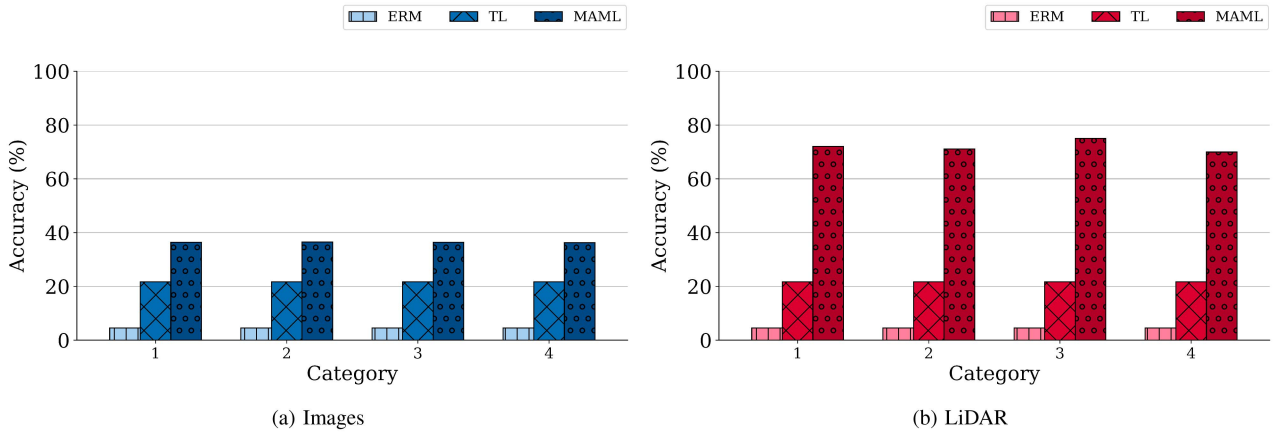


Fig. 11. Results for training on single-category data from $\mathcal{DT}$ and testing on *all* data from $\mathcal{R}$.

distinct changes such that (1) the framework is able to train on, create models from, and test on the generated synthetic data from the DT and (2) we change the model architecture such that it no longer uses *CameraNet* or *LidarNet*, instead using the *VGGNet-16* (VGG) architecture [51] for our beam selection problem, given that VGG is implemented in MAML, and we keep the model architecture between approaches as consistent as possible. Notably, the VGG architecture for both LiDAR and image modalities is the same, with changes made only to the input layer to accommodate the different input sizes. This is done to preserve the effect of the VGG architecture in our evaluation and to fairly assess performance, regardless of the data modality.

MAML: We use the MAML algorithm as presented in Section IV-C and IV-D. As MAML in its original implementation was designed for image classification tasks, we take advantage of the unstructured nature of LiDAR pointclouds, noting that simply reshaping each pointcloud into a (length, width, channel) structure as used to traditionally represent images in three dimensions is enough while keeping the voxel values as defined in Section IV-B4 the same. Subsequent results we present in this section support this notion.

B. Experiment Setup

For all experiments, we set a maximum of 100 epochs for training and use the Adam optimizer [52] with a learning rate of 0.001 for all algorithms, with early stopping implemented if training accuracy does not increase in 10 epochs. Slightly modified from [15], we use 70 samples per class per task for a total of $m := m_1 + m_2 = 2380$ samples per task in ERM while using 20 support samples and 50 target samples from each class per task in MAML. Since each task has 34 classes, this means that $m_1 = 680$ and $m_2 = 1700$ for all tasks. For MAML and ERM, the batch size is set to 5 tasks sampled per epoch, and for MAML specifically, we use five fine-tuning steps and set $\tau = 5$.

For TL, we note that the experiment setup process is slightly different as defined in [35], but the intended end result is comparable to the results produced by MAML and ERM. Since TL is inherently a two-step process, we set a batch size of 32 samples for all experiments with differently-sized partitions used at each step. When creating the initial model, we perform no TL at all and use training, validation, and testing partition sizes of 80%, 10%, and 10% of the input synthetic dataset, respectively. When performing TL by using the previously generated on the real-world dataset, we fully retrain the existing weights with a respective 5% and 2% of the real-world data for training and validation partition in order to keep training sets similar in total samples across each step. The rest of the real-world data is used for testing. This approach ensures a fair comparison between TL and MAML by maintaining consistency in dataset proportions while highlighting the effectiveness of minimal real-world fine-tuning.

C. Evaluation

To evaluate model performance across various experiments, we compute accuracy using the following formula:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \times 100 \tag{4}$$

where TP, TN, FP, and FN denote the number of true positives, true negatives, false positives, and false negatives, respectively. This metric enables us to assess the effectiveness of beam selection predictions across synthetic and real-world datasets.

1) Domain Adaptation Using All Categories: To begin, we simply train models on synthetic data generated in the DT from all categories, then test on the real-world data from the real-world category counterparts. In short, we train on all data from $\mathcal{DT}$ and test on all data from $\mathcal{R}$ and show the results in Fig. 9. This initial experiment demonstrates MAML's superior performance over the other approaches and establishes a benchmark for the subsequent experiments.

Observation 1: We note that MAML outperforms ERM and TL by 14.04× and 2.79×, respectively, when using images, and 13.12× and 3.24×, respectively, when using LiDAR pointclouds (see Fig. 9, validates Contribution 4).

Additionally, we note that the accuracy computed using (4) is 10.30% higher for LiDAR with MAML compared to images with MAML. Our results show that LiDAR data adapts more efficiently than camera images due to its ability to capture spatial structure consistently across synthetic and real-world domains. LiDAR point clouds, whether generated in the DT or collected from real-world environments, primarily encode geometric information such as object positioning and obstacle presence. Since our beam selection task relies on coarse spatial features rather than fine-grained information, this structural alignment ensures a minimal domain gap. Furthermore, to effectively utilize LiDAR data, we had transformed the point clouds into structured, ordered grid representations of the 3D space using a quantized voxel-based encoding. This structured encoding reduces sensitivity to domain-specific noise while preserving key spatial features relevant to beam selection. MAML-based adaptation learns an optimal initialization that quickly refines these voxelized representations with minimal adjustments, primarily addressing minor density variations and noise. In contrast, camera images exhibit significant variations in illumination, color distortions, and textures, making direct adaptation more challenging. These discrepancies necessitate additional domain adaptation techniques beyond MAML to achieve robust real-world alignment. Our findings highlight that leveraging LiDAR's stable depth-based features, combined with structured voxel encoding, enables more efficient Sim2Real transfer for beam selection, reducing the need for extensive fine-tuning and making it a more reliable modality for adaptation.

2) Domain Adaptation Using Individual Categories: We then analyze the effects of each approach in a category-wise manner, i.e., training on a single category with synthetic data from $\mathcal{DT}$ and testing on the analogous category with real-world data from $\mathcal{R}$. Fig. 10 shows the results across all categories for each approach for image (Fig. 10(a)) and LiDAR (Fig. 10(b)) modalities. Using (4) the accuracy results presented in Fig. 10 are based on multi-class classification accuracy, where the model predicts the optimal beam among multiple beam patterns. This metric allowed us to evaluate the model's ability to predict beam selection outcomes when trained on DT data and tested on analogous real-world data from $\mathcal{R}$. We include Fig. 8 for completeness, showing the accuracy of the models used in TL when tested on synthetic data. Note that this data is not available in MAML or ERM as the pre-trained model is fine-tuned and tested directly on real-world data instead of being retrained.

Across all experiments, we note that MAML outperforms ERM and TL by an average of 5.34× and 1.49×, respectively, for image samples. MAML performs at least ∼15% better in *Category 1* conditions versus the other NLOS environments, which may be attributed to a low number of sector variability due to the lack of obstacles present. MAML's ability to generalize across tasks helps it adapt more effectively to unseen NLOS

conditions, where TL struggles with domain shifts. The model benefits from meta-training across diverse synthetic NLOS scenarios, which improves its ability to fine-tune on real-world NLOS cases with minimal data. General MAML performance is increased when using LiDAR samples, with an average 1.51× accuracy increase over using images and boosting the average accuracy up by 8.98× over the ERM performance and 4.09× over the TL performance for each individual category. These boosts in accuracy illustrate how LiDAR, as a 3D representation of the environment, is able to capture more information from the environment, specifically in the presence of obstacles and thus has more leverageable knowledge when performing domain adaptation from a synthetic to real environment. (addresses Contribution 3). Noticeably, ERM and TL performance remained consistent across modalities with both approaches having nearly equal performance in *Category 1* and *Category 3* and TL having much higher accuracies in *Category 2* and *Category 4*, regardless of modality. This may speak to the importance of modeling dynamic conditions within a DT; even with a stationary Tx and a mobile Rx, the ray-tracer may not be able to mimic real-world changes to the optimal sector without the presence of moving obstacles. With regards to the TL performance, we note that a simple retraining step, even when using a larger portion of the real-world dataset in comparison to the few-shot fine-tuning process of MAML, may not be adequate in the presence of such a large domain shift, i.e., from $\mathcal{DT}$ to $\mathcal{R}$.

Observation 2: Throughout all the experiments, MAML outperforms ERM and TL by 5.34× and 1.49× respectively when using images, and 8.98× and 4.09× respectively when using LiDAR samples. Using MAML, the LiDAR data outperforms the image experiments by an average of 1.51× (refer to Fig. 10, validates Contribution 3).

The experimental findings in Fig. 10 shows that MAML outperforms ERM and TL significantly, with LiDAR achieving an 8.98× improvement over ERM compared to 5.34× for images. Additionally, MAML performance for LiDAR remains relatively consistent across categories along with a 1.51× higher performance over images, aligning with our insight that LiDAR requires fewer adjustments, making it inherently more adaptable in meta-learning-based domain transfer.

3) All Data Domain Adaptation Using Individual Categories: In this set of experiments, we use single-category synthetic data for training, but test instead on real-world data from every category, keeping testing partition sizes and content the same. This is done to analyze the practical effectiveness of using single-category synthetic data in domain adaptation to any given data from a real-world environment. Again, we use the TL models with accuracies shown in Fig. 8 and display the experiment results computed using (4) in Fig. 11.

As the test set is the same across all experiments, we note that image sample performance is consistent (within ∼1%) across all categories, with Category 2 having the highest MAML performance at 38.48%. There is more of a discrepancy when using LiDAR samples, with Category 3 having the highest MAML-based accuracy at 75.01%, and LiDAR samples yielding accuracies about 1.98× that of images on average. ERM and TL performances remain similar across categories and modalities with MAML outperforming, on average, 8.01× and 1.68× ERM and TL with images and a remarkable 15.87× and 3.32× ERM and TL, respectively. These results suggest the environmental similarities across categories within $\mathcal{DT}$ while highlighting the larger discrepancies between categories in their real-world counterparts, suggesting the need for greater resolution and variability when creating the $\mathcal{DT}$.

TABLE V
TRAINING COMPUTATION TIME ACROSS ALL CATEGORIES FOR SINGLE CATEGORY-WISE DOMAIN ADAPTATION PERFORMED WITH TL AND MAML

Approach	Images Avg. ± Std. Dev. (s)	LiDAR Avg. ± Std. Dev. (s)
TL	12.61±2.68	7.51±1.09
MAML	5.20±0.27	5.00±0.30

The times given for TL are the sum of the computation times for pre-training the model on synthetic data and re-training the model on real-world data.

Observation 3: We observe similar outcomes for MAML, as it outperforms ERM and TL by 8.01× and 1.68× for images, and 15.87× and 3.32× for LiDAR samples, respectively. With MAML, LiDAR samples outperform the image experiments by 1.98× (refer to Fig. 11, validates Contribution 2).

4) Computation Time: Finally, we provide a brief synopsis of computation time for the TL and MAML algorithms. Table V shows the average computation time *per epoch* during training in single-category-wise domain adaptation (Section VI-C2), as the number of training epochs per category may vary due to early stopping. We do not evaluate ERM, as it has the lowest performance out of the three approaches, and do not account for fine-tuning time, inference time, and the time it takes to load the pre-trained model for TL specifically, as these are negligible.

The larger computation time for each training epoch when performing TL can be accounted for by two sources: 1) as TL is inherently a two-step process, meaning that a model has to be pre-trained with the synthetic data before performing TL with the real-world data, *two* runs need to be executed in comparison to the singular execution of MAML, which has fine-tuning built into the meta-testing phase, and 2) When retraining the pre-trained model in TL, a significantly greater amount of data is required compared to the few-shot nature of MAML, leading to higher memory usage and increased computational overhead. In contrast, MAML remains a lightweight adaptation mechanism that drastically reduces training overhead, making it far more practical for real-time, low-latency V2X deployments, where rapid beam selection is critical.

Observation 4: As model creation and training take up the majority of the computation time, MAML deployment provides faster domain adaptation than the SOTA TL technique (refer to Table V, validates Contribution 4).

VII. DISCUSSIONS

The proposed SMART framework demonstrates the effectiveness of DT-based synthetic data and meta-learning in addressing domain adaptation challenges for mmWave beam selection in V2X networks. By creating multimodal DT environments

that generate high-fidelity LiDAR and camera data alongside ray-tracing-based RF measurements, the framework enables robust model training without the constraints of real-world data collection.

• *Broader Observations:* Our experimental results show that meta-learning, particularly MAML, significantly outperforms traditional TL by achieving up to 14.04× higher accuracy with minimal real-world fine-tuning. However, challenges remain in improving DT fidelity, optimizing computational efficiency, and expanding the framework's applicability.

• *Future Directions:* Future work will focus on enhancing the robustness and scalability of the SMART framework by expanding the dataset to include higher-resolution data, performing data augmentation for camera-based images, incorporating additional modalities such as radar, and introducing greater variations to better capture complex real-world scenarios. Our results indicate that camera-based images exhibit significant variability due to illumination changes and texture inconsistencies, limiting their direct Sim2Real transfer. Addressing this requires higher-quality image data and improved domain adaptation techniques before effectively integrating them into a multimodal fusion approach. We also aim to enable real-time deployment of the framework for V2X mmWave beam selection by optimizing computational efficiency through methods like quantization and model pruning, ensuring compatibility with edge devices.

Additionally, to improve accessibility and reproducibility, we plan to migrate the data collection process from the commercial Wireless InSite (WI) [18] framework to the open-source Sionna [49] framework. While WI offers high-fidelity ray-tracing with detailed material modeling, its computational cost and licensing constraints limit large-scale dataset generation. Sionna, with its GPU-accelerated architecture, provides a more efficient alternative for generating extensive and diverse datasets, making it a viable option for future DT-based simulations. To address challenging beam selection scenarios, we will explore task-robust MAML [43] to ensure equal importance is given to rare or difficult tasks, thus improving adaptability across dynamic conditions. Finally, we intend to expand the S-FLASH dataset, improve open-source accessibility, and evaluate the framework under adversarial attacks and dynamic spectrum allocation, ensuring its robustness and reliability for future V2X applications.

VIII. Conclusion

We make a case for leveraging DT-generated multimodal sensor data to enhance mmWave beamforming in V2X networks, addressing the limitations of solely RF-based approaches. The SMART framework integrates deep learning-driven synthetic data generation with advanced domain adaptation techniques, demonstrating how multimodal data fusion—incorporating LiDAR, camera images, and ray-traced RF signals—can improve beam selection efficiency while reducing the dependency on extensive real-world data collection. Our results show that models trained on synthetic DT-generated data, when fine-tuned with minimal real-world samples using meta-learning, significantly outperform traditional transfer learning approaches. Specifically, our MAML-based domain adaptation method achieves up to 14.04× accuracy improvements over models trained without adaptation and surpasses transfer learning approaches by up to 4.09×. Additionally, LiDAR-based learning demonstrates superior adaptation efficiency compared to image-based models, emphasizing the robustness of structured 3D representations in overcoming domain shift challenges. Our study highlights the feasibility of DT-based synthetic-to-real adaptation for real-world V2X deployments, paving the way for practical, scalable, and computationally efficient beam selection solutions. The dataset and code for the proposed SMART framework, including multimodal data generation and meta-learning-based beam selection models, will be released online [53] for independent validation and further research on Sim2Real adaptation in V2X communication upon the acceptance of this article.

Acknowledgment

The authors gratefully thank Liam Collins, Dr. Aryan Mokhtari, and Dr. Sanjay Shakkottai from The University of Texas at Austin for their feedback on meta-learning implementation.

References

[1] L. Hobert, A. Festag, I. Llatser, L. Altomare, F. Visintainer, and A. Kovacs, "Enhancements of V2X communication in support of cooperative autonomous driving," *IEEE Commun. Mag.*, vol. 53, no. 12, pp. 64–70, Dec. 2015.

[2] C. Gschwendtner, S. R. Sinsel, and A. Stephan, "Vehicle-to-X (V2X) implementation: An overview of predominate trial configurations and technical, social and regulatory challenges," *Renewable Sustain. Energy Rev.*, vol. 145, 2021, Art. no. 110977. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1364032121002690

[3] C. R. Storck and F. Duarte-Figueiredo, "5G V2X ecosystem providing entertainment on board using mmWave communications," in *Proc. IEEE 10th Latin- Amer. Conf. Commun.*, 2018, pp. 1–6.

[4] M. H. C. Garcia et al., "A tutorial on 5G NR V2X communications," *IEEE Commun. Surveys Tut.*, vol. 23, no. 3, pp. 1972–2026, Third Quarter, 2021.

[5] C. N. Barati et al., "Initial access in millimeter wave cellular systems," *IEEE Trans. Wireless Commun.*, vol. 15, no. 12, pp. 7926–7940, Dec. 2016.

[6] D. Steinmetzer, D. Wegemer, M. Schulz, J. Widmer, and M. Hollick, "Compressive millimeter-wave sector selection in off-the-shelf IEEE 802.11ad devices," in *Proc. Int. Conf. Emerg. Netw. EXperiments Technol.*, 2017, pp. 414–425.

[7] M. Alrabeiah, A. Hredzak, Z. Liu, and A. Alkhateeb, "ViWi: A deep learning dataset framework for vision-aided wireless communications," in *Proc. IEEE 91st Veh. Technol. Conf.*, 2019, pp. 1–5.

[8] A. Klautau, P. Batista, N. Gonzalez-Prelcic, Y. Wang, and R. W. Heath, "5G MIMO data for machine learning: Application to beam-selection using deep learning," in *Proc. Inf. Theory Appl. Workshop*, 2018, pp. 1–9.

[9] M. B. Mashhadi, M. Jankowski, T.-Y. Tung, S. Kobus, and D. Gündüz, "Federated mmwave beam selection utilizing LiDAR data," *IEEE Wireless Commun. Lett.*, vol. 10, no. 10, pp. 2269–2273, Oct. 2021.

[10] B. Salehi, J. Gu, D. Roy, and K. Chowdhury, "FLASH: Federated learning for automated selection of high-band mmwave sectors," in *Proc. IEEE Conf. Comput. Commun.*, 2022, pp. 1719–1728.

[11] J. Gu, B. Salehi, D. Roy, and K. R. Chowdhury, "Multimodality in mmWave MIMO beam selection using deep learning: Datasets and challenges," *IEEE Commun. Mag.*, vol. 60, no. 11, pp. 36–41, Nov. 2022.

[12] Q. Dai, X.-M. Wu, J. Xiao, X. Shen, and D. Wang, "Graph transfer learning via adversarial domain adaptation with graph convolution," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 5, pp. 4908–4922, May 2023.

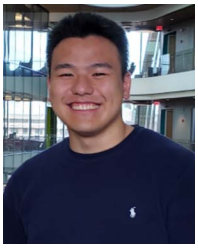
[13] W. Yang, C. Yang, S. Huang, L. Wang, and M. Yang, "Few-shot unsupervised domain adaptation via meta learning," in *Proc. IEEE Int. Conf. Multimedia Expo*, 2022, pp. 1–6.

[14] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 1126–1135.

[15] J. Gu, L. Collins, D. Roy, A. Mokhtari, S. Shakkottai, and K. R. Chowdhury, "Meta-learning for image-guided millimeter-wave beam selection in unseen environments," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2023, pp. 1–5.
[16] B. Foundation, "Blender," 2023. [Online]. Available: https://www.blender.org/
[17] M. Gschwandtner, R. Kwitt, A. Uhl, and W. Pree, "Blensor: Blender sensor simulation toolbox," in *Proc. Adv. 7th Int. Symp. Vis. Comput.*, 2011, pp. 199–208.
[18] Remcom, "Wireless InSite," 2023. [Online]. Available: https://www.remcom.com/wireless-insite-em-propagation-software
[19] S. Jiang, G. Charan, and A. Alkhateeb, "LiDAR aided future beam prediction in real-world millimeter wave V2I communications," *IEEE Wireless Commun. Lett.*, vol. 12, no. 2, pp. 212–216, Feb. 2023.
[20] D. Marasinghe et al., "LiDAR aided wireless networks - beam prediction for 5G," in *Proc. IEEE 96th Veh. Technol. Conf.*, 2022, pp. 1–7.
[21] G. Charan, T. Osman, A. Hredzak, N. Thawdar, and A. Alkhateeb, "Vision-position multi-modal beam prediction using real millimeter wave datasets," in *Proc. IEEE Wireless Commun. Netw. Conf.*, 2022, pp. 2727–2731.
[22] J. Nie, Q. Zhou, J. Mu, and X. Jing, "Vision and radar multimodal aided beam prediction: Facilitating metaverse development," in *Proc. 2nd Workshop Integr. Sens. Commun. Metaverse*, New York, NY, USA, 2023, pp. 13–18, doi: 10.1145/3597065.3597449.
[23] Q. Zhou, Y. Lai, H. Yu, R. Zhang, X. Jing, and L. Luo, "Multi-modal fusion for millimeter-wave communication systems: A spatio-temporal enabled approach," *Neurocomputing*, vol. 555, 2023, Art. no. 126604.
[24] H. X. Nguyen, R. Trestian, D. To, and M. Tatipamula, "Digital twin for 5G and beyond," *IEEE Commun. Mag.*, vol. 59, no. 2, pp. 10–15, Feb. 2021.
[25] L. U. Khan, Z. Han, W. Saad, E. Hossain, M. Guizani, and C. S. Hong, "Digital twin of wireless systems: Overview, taxonomy, challenges, and opportunities," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 4, pp. 2230–2254, Fourth Quarter, 2022.
[26] S. Almeaibed, S. Al-Rubaye, A. Tsourdos, and N. P. Avdelidis, "Digital twin analysis to promote safety and security in autonomous vehicles," *IEEE Commun. Standards Mag.*, vol. 5, no. 1, pp. 40–46, Mar. 2021.
[27] L. Zhao, Z. Bi, A. Hawbani, K. Yu, Y. Zhang, and M. Guizani, "Elite: An intelligent digital twin-based hierarchical routing scheme for softwarized vehicular networks," *IEEE Trans. Mobile Comput.*, vol. 22, no. 9, pp. 5231–5247, Sep. 2023.
[28] T. Wágner, T. Ormándi, T. Tettamanti, and I. Varga, "Spat/map V2X communication between traffic light and vehicles and a realization with digital twin," *Comput. Elect. Eng.*, vol. 106, 2023, Art. no. 108560.
[29] C. M. Ezhilarasu, Z. Skaf, and I. K. Jennions, "Understanding the role of a digital twin in integrated vehicle health management (IVHM)," in *Proc. IEEE Int. Conf. Syst., Man Cybern.*, 2019, pp. 1484–1491.
[30] W. Sun, P. Wang, N. Xu, G. Wang, and Y. Zhang, "Dynamic digital twin and distributed incentives for resource allocation in aerial-assisted internet of vehicles," *IEEE Internet Things J.*, vol. 9, no. 8, pp. 5839–5852, Apr. 2022.
[31] S. Zelenbaba, B. Rainer, M. Hofer, and T. Zemen, "Wireless digital twin for assessing the reliability of vehicular communication links," in *Proc. IEEE Globecom Workshops*, 2022, pp. 1034–1039.
[32] U. Demir, S. Pradhan, R. Kumahia, D. Roy, S. Ioannidis, and K. Chowdhury, "Digital twins for maintaining QoS in programmable vehicular networks," *IEEE Netw. Mag.*, vol. 37, no. 4, pp. 208–214, Jul./Aug. 2023.
[33] L. Cazzella, F. Linsalata, M. Magarini, M. Matteucci, and U. Spagnolini, "A multi-modal simulation framework to enable digital twin-based V2X communications in dynamic environments," Mar. 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2303.06947
[34] S. Dodge and L. Karam, "Understanding how image quality affects deep neural networks," in *Proc. 8th Int. Conf. Qual. Multimedia Experience*, 2016, pp. 1–6.
[35] J. Gu, B. Salehi, S. Pimple, D. Roy, and K. R. Chowdhury, "Tune: Transfer learning in unseen environments for V2X mmWave beam selection," in *Proc. IEEE Int. Conf. Commun.*, 2023, pp. 1658–1663.
[36] P. Singhal, R. Walambe, S. Ramanna, and K. Kotecha, "Domain adaptation: Challenges, methods, datasets, and applications," *IEEE Access*, vol. 11, pp. 6973–7020, 2023.
[37] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "PointPillars: Fast encoders for object detection from point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 12697–12705.
[38] S. Huch, L. Scalerandi, E. Rivera, and M. Lienkamp, "Quantifying the LiDAR sim-to-real domain shift: A detailed investigation using object detectors and analyzing point clouds at target-level," *IEEE Trans. Intell. Veh.*, vol. 8, no. 4, pp. 2970–2982, Apr. 2023.
[39] Y. Luo, L. Zheng, T. Guan, J. Yu, and Y. Yang, "Taking a closer look at domain shift: Category-level adversaries for semantics consistent domain adaptation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 2507–2516.
[40] A. Nagabandi et al., "Learning to adapt in dynamic, real-world environments through meta-reinforcement learning," 2019.
[41] K. Arndt, M. Hazara, A. Ghadirzadeh, and V. Kyrki, "Meta reinforcement learning for sim-to-real domain adaptation," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2020, pp. 2725–2731.
[42] A. Antoniou, H. Edwards, and A. Storkey, "How to train your MAML," in *Proc. Int. Conf. Learn. Representations*, 2019. [Online]. Available: https://arxiv.org/abs/1810.09502
[43] L. Col lins, A. Mokhtari, and S. Shakkottai, "Task-robust model-agnostic meta-learning," in *Proc. Adv. Neural Inf. Process. Syst.*, Curran Associates, Inc., 2020, pp. 18860–18871.
[44] "OpenStreetMaps," 2023. [Online]. Available: https://www.openstreetmap.org/
[45] T. W. Consortium, "Extensible 3D," 2023. [Online]. Available: https://www.web3d.org/x3d/what-x3d
[46] MATLAB, "LiDAR toolbox," 2023. [Online]. Available: https://www.mathworks.com/products/lidar.html
[47] "WiFi routers| AD7200," 2025. Accessed: Nov. 11, 2022. [Online]. Available: https://www.tp-link.com/us/home-networking/wifi-router/ad7200/#specifications
[48] "Measurements legacy," 2018. Accessed: Nov. 11, 2022. [Online]. Available: https://github.com/seemoo-lab/talon-sector-patterns/tree/master/legacymeasurements
[49] J. Hoydis et al., "Sionna: An open-source library for next-generation physical layer research," 2023. [Online]. Available: https://arxiv.org/abs/2203.11854
[50] M. Gschwandtner, "Blender sensor simulation," 2023, Accessed: Feb. 14, 2025. [Online]. Available: https://www.blensor.org/
[51] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Representations*, 2015.
[52] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014. [Online]. Available: https://arxiv.org/abs/1412.6980
[53] "SMARTdataset," 2025. [Online]. Available: https://genesys-lab.org/smart

Divyadharshini Muruganandham is currently working toward the PhD degree in computer engineering with the University of Texas at Austin under the supervision of Prof. K. Chowdhury. Her research lies at the intersection of applied machine learning and next-generation wireless systems, with a particular focus on O-RAN architectures, digital twin, mmWave beamforming, and AI-driven network optimization. She is actively involved in experimental deployments of emerging wireless technologies, aimed at shaping the future of next-generation communication systems.

Suyash Pradhan (Graduate Student Member, IEEE) received the BTech degree in electronics engineering from Veermata Jijabai Technological Institute, Mumbai, in May 2021. He is currently working toward the PhD degree with The University of Texas at Austin, where he serves as a graduate research assistant under the mentorship of professor Kaushik Chowdhury. His research centers on integrating machine learning with wireless communication systems and developing experimental frameworks for deploying distributed intelligence. Specifically, he works on Over-the-Air Federated Learning, bridging the gap between simulation-based approaches and real-world deployment while exploring the use of Digital Twins for wireless networks.



Jerry Gu received the MS degree in electrical and computer engineering from Northeastern University, in 2021. His current research focuses on the use of machine learning in wireless communications, including multimodal data fusion and RF fingerprinting.

Torsten Braun (Senior Member, IEEE) received the PhD degree from the University of Karlsruhe (Germany), in 1993. He is a head of the Communication and Distributed Systems (CDS) research group with the Institute of Computer Science, University of Bern, where he has been a full professor since 1998. From 1994 to 1995, he was a guest scientist with INRIA Sophia-Antipolis (France). From 1995 to 1997, he worked with the IBM European Networking Centre Heidelberg (Germany) as a project leader and senior consultant. He has been a vice president of the SWITCH (Swiss Research and Education Network Provider) Foundation from 2011 to 2019. He has been a deirector with the Institute of Computer Science and Applied Mathematics, University of Bern between 2007 and 2011, and from 2019 to 2021. His research interests are networking support for virtual reality, distributed machine learning for future wireless and mobile networks as well as energy-efficient machine learning for the Internet of Things.

Debashri Roy (Senior Member, IEEE) received the PhD degree in computer science from the University of Central Florida. She is an assistant professor with the Department of Computer Science and Engineering, The University of Texas at Arlington. Her research interests involve machine learning based applications in wireless communication domain, targeted to the areas of deep spectrum learning, millimeter wave beamforming, multimodal fusion, Open-RAN, networked robotics for next-generation communication.

Kaushik Chowdhury (Fellow, IEEE) received the PhD degree from the Georgia Institute of Technology, in 2009. He is a Chandra Family Endowed distinguished professor in electrical and computer engineering with The University of Texas at Austin. His current research interests involve systems aspects of machine learning for agile spectrum sensing/access, unmanned autonomous systems, programmable and open cellular networks, and largescale experimental deployment of emerging wireless technologies.