An O-RAN Framework for AI/ML-Based Localization with OpenAirInterface and FlexRIC

Nada Bouknana*, Mohsen Ahadi* , Florian Kaltenberger*, Robert Schmidt†
*EURECOM, Sophia Antipolis, France
Email:{nada.bouknana,mohsen.ahadi,florian.kaltenberger}@eurecom.fr
†OpenAirInterface Software Alliance, Sophia Antipolis, France
Email: robert.schmidt@openairinterface.org

***Abstract*—Localization is increasingly becoming an integral component of wireless cellular networks. The advent of artificial intelligence (AI) and machine learning (ML) based localization algorithms presents potential for enhancing localization accuracy. Nevertheless, current standardization efforts in the third generation partnership project (3GPP) and the O-RAN Alliance do not support AI/ML-based localization. In order to close this standardization gap, this paper describes an O-RAN framework that enables the integration of AI/ML-based localization algorithms for real-time deployments and testing. Specifically, our framework includes an O-RAN E2 Service Model (E2SM) and the corresponding radio access network (RAN) function , which exposes the Uplink Sounding Reference Signal (UL-SRS) channel estimates from the E2 agent to the Near real-time RAN Intelligent Controller (Near-RT RIC). Moreover, our framework includes, as an example, a real-time localization external application (xApp), which leverages the custom E2SM-SRS in order to execute continuous inference on a trained Channel Charting (CC) model, which is an emerging self-supervised method for radio-based localization. Our framework is implemented with OpenAirInterface (OAI) and FlexRIC, democratizing access to AI-driven positioning research and fostering collaboration. Furthermore, we validate our approach with the CC xApp in real-world conditions using an O-RAN based localization testbed at EURECOM. The results demonstrate the feasibility of our framework in enabling real-time AI/ML localization and show the potential of O-RAN in empowering positioning use cases for next-generation AI-native networks.**

***Index Terms*—5G, O-RAN, Localization, xApp, Channel Charting, OAI, FlexRIC**

I. Introduction

Localization continues to evolve as a key technology for current and next-generation networks. By leveraging large bandwidths, high frequencies, and massive Multiple-Input-Multiple-Output (MIMO) capabilities, fifth-generation (5G) systems can enable precise and reliable positioning. In addition, the service-based architecture and ultra-lean design of 5G make it a cost-effective solution. Furthermore, 5G's focus on industrial verticals aligns with the requirements of location-based services (LBS).

The third generation partnership project (3GPP) positioning algorithms are geometry-based. They rely on channel parameters for position estimation, such as propagation delay, angle, or signal strength [1]. These measurements are then processed by localization algorithms to estimate the position of the user equipment (UE) [2]. However, these methods rely on line-of-sight (LOS) conditions. Consequently, their performance can degrade significantly in non-line-of-sight (NLoS) or dense multipath environments.

While conventional methods are limited in terms of accuracy and performance by the propagation conditions, emerging artificial intelligence (AI) and machine learning (ML) methods offer a promising solution to overcome these limitations, and provide reliable and accurate positioning. Channel state information (CSI) contains features that can be exploited for positioning through AI/ML data-driven tools, either to assist traditional methods or to perform direct positioning. Moreover, AI/ML models for positioning perform well in the presence of NLoS conditions, in indoor environments with dense multipath propagation, and even when the number of available antennas is limited or the user is in motion [3].

The current 3GPP 5G positioning architecture does not support the integration of AI/ML algorithms. However, recent studies in 3GPP consider the adoption of AI/ML for positioning [4]. This paves the way to adopting an AI-native architecture for the sixth-generation (6G).

Building on the principles of openness, virtualization, intelligence, and programmability. Open radio access network (O-RAN) offers promising prospects for the integration of AI/ML algorithms in the radio access network (RAN). To this end, the O-RAN standardized interfaces and components can offer a flexible and highly programmable solution. Moreover, the O-RAN Alliance defines a set of requirements for integrating AI/ML in the O-RAN architecture [5]. The Non real-time RAN Intelligent Controller (Non-RT RIC) can be used for training AI/ML models and providing updated models to the Near real-time RAN Intelligent Controller (Near-RT RIC). Furthermore, the Near-RT RIC can host external applications (xApps) to perform the AI/ML inference. This approach can also be applied to positioning, which is considered in the O-RAN Alliance Use Cases Detailed Specification [6]. Nonetheless, the O-RAN Alliance does not specify the detailed message flow, or any dedicated E2 Service Model (E2SM) for enabling AI/ML-based positioning.

In order to address this standardization gap, this paper proposes and validates an O-RAN framework which enables AI/ML-based localization for real-time deployments. Our proposed solution leverages the O-RAN E2 interface in order to transport the channel estimates of the Uplink Sounding Reference Signal (UL-SRS) to the Near-RT RIC and enable real-

ISBN: 978-3-903176-79-9

time AI/ML localization inference in an xApp. We provide a concrete open-source implementation of a new custom O-RAN compatible E2SM using FlexRIC [7] and integrate it with OpenAirInterface (OAI) [8]. Furthermore, we provide, as an example, a real-time localization xApp that performs Channel Charting (CC) inference, which is an emerging self-supervised data-driven method for UE localization in wireless networks. It is based on learning the similarities between high dimensional CSI measurements and applying dimensionality reduction techniques to the CSI to produce a lower dimensional embedding called channel chart. The produced channel chart contains information about the user location [9]. The xApp uses a pre-trained CC model based on a novel algorithm that we previously proposed in [10]. This paper also discusses experimental results obtained from real-time testing under real-world conditions with an O-RAN based localization testbed at EURECOM.

Several works have studied and evaluated wireless localization techniques in 5G using OAI. For instance, authors in [11] provide a detailed guide on the implementation of positioning features in OAI, mostly focusing on the physical (PHY) layer and on the useful tools for data extraction which is necessary for post-processing and analysis. Authors in [12] demonstrated the integration of NR positioning protocol annex (NRPPa) and Uplink time difference of arrival (TDoA) using the UL-SRS for positioning in OAI. Finally, authors in [13] presented experimental results of 5G positioning in real-conditions using experimental testbeds that are integrated with OAI, they also demonstrate a new framework suitable for AI/ML-based positioning in beyond-5G. However, the implementation does not use O-RAN standardized protocols. In summary, the primary focus of the previous works was implementing and benchmarking different positioning algorithms with OAI. However, and to the best of the authors' knowledge, our work is the first to propose a framework, with a concrete implementation, which enables AI/ML localization inference using FlexRIC and OAI.

The main contributions of this paper can be summarized as follows: we present an O-RAN-based framework, which constitutes of a custom O-RAN E2SM and an xApp, that enable real-time AI/ML inference for SRS positioning. We provide a concrete open-source implementation, of the E2SM and a specialized CC localization xApp, using OAI and FlexRIC. We validate our solution in real-world conditions using an O-RAN-based localization testbed at EURECOM.

The remainder of this paper is organized as follows. Section II describes the system model. Section III presents the deployed CC model. Section IV discusses the implementation details of the custom O-RAN service model, which transports the UL-SRS channel estimates from the E2 agent to the Near-RT RIC, it also describes the implementation of the real-time CC-based localization xApp. Section V describes the experimental validation conducted using the Firecell GEO5G testbed at EURECOM, and analyzes the obtained results. Finally, section VI concludes the paper and provides future perspectives.

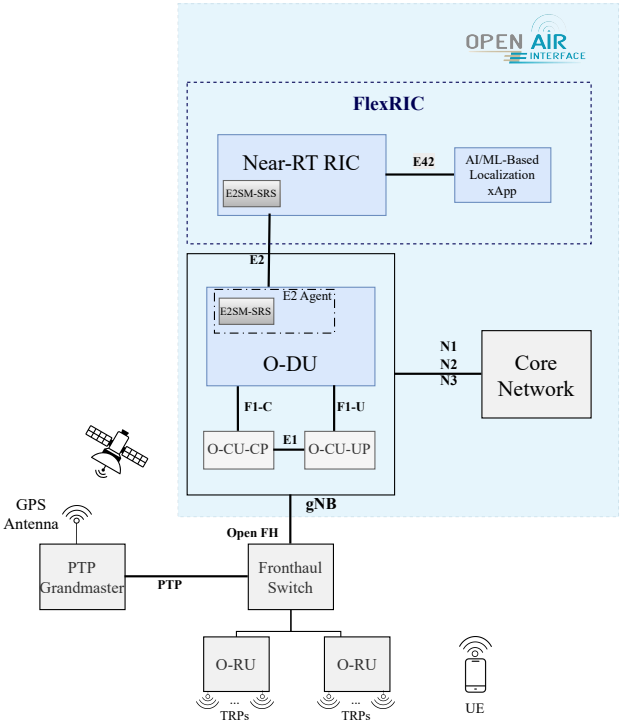

Fig. 1: AI/ML-based localization system model

II. System Model

As shown in Figure 1, our AI/ML-based 5G localization system model consists of an O-RAN compliant 5G system, with the Core Network (CN) and the RAN which includes the Next Generation Node B (gNB) and the UE. The UE transmits the UL-SRS signal to one or multiple gNBs. While we consider one gNB in this work, the approach can be extended to mutiple gNBs. The gNB is further disaggregated into centralized unit (CU), distributed unit (DU), and radio unit (RU) (O-CU, O-DU, and O-RU following the O-RAN architecture). The O-DU and O-RU communicate via the O-RAN 7.2 fronthaul interface.

There are K RUs each with known Transmission Reception Points (TRP) locations $\mathbf{x}_{m_k}$ where $m_k \in \{1, \ldots, M_k\}$. Every TRP receives the transmitted UL-SRS signal. The O-DU estimates the UL-SRS channel. The estimated channel frequency response (CFR) of the link between the m_k-th TRP of the k-th RU and the UE at time step t, spanning all orthogonal frequency division multiplexing (OFDM) N_{fft} sub-carriers, is denoted as $\mathbf{w}_{k,m_k,t} \in \mathbb{C}^{N_{\text{fft}}}$. The E2 interface transports the estimated frequency-domain MIMO channel matrix from the E2 agent (O-DU in our system model) to the Near-RT RIC. An xApp, which is part of the RIC, continuously performs the machine learning inference and updates the UE position.

We note the channel impulse response (CIR) by $\mathbf{h}_{k,m_k,t} \in \mathbb{C}^{N_{\text{fft}}}$ which is obtained by applying an inverse discrete Fourier transform (IDFT) to the CFR. In our configuration, the IDFT

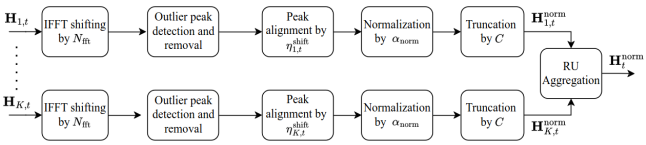

Fig. 2: CC pre-processing pipeline [10]

is performed by the xApp. Collecting the CIRs from all M_k TRPs of RU k yields the RU-level CIR matrix:

$$\mathbf{H}_{k,t} = \begin{bmatrix} \mathbf{h}_{k,1,t} \\ \mathbf{h}_{k,2,t} \\ \vdots \\ \mathbf{h}_{k,M_k,t} \end{bmatrix} \in \mathbb{C}^{M_k \times N_{\text{fft}}} \tag{1}$$

Similarly, the global CIR matrix across all K RUs is

$$\mathbf{H}_{t} = \begin{bmatrix} \mathbf{H}_{1,t} \\ \vdots \\ \mathbf{H}_{K,t} \end{bmatrix} \in \mathbb{C}^{M \times N_{\text{fft}}} \tag{2}$$

where $M = \sum_{k=1}^{K} M_k$ is the total number of TRPs across all RUs. The global CIR $\mathbf{H}_t$ is subsequently processed for ML inference.

III. Channel Charting

In this section, we provide a brief overview of the deployed CC model. The development of the CC algorithm is not part of this work, and we refer the reader to our previous work which proposed this novel CC algorithm [10].

We aim to learn a self-supervised CC function $f_{\boldsymbol{\theta}}(\cdot)$ that maps a high dimensional channel measurement into a two-dimensional embedding space:

$$f_{\boldsymbol{\theta}} : \mathbb{R}^{M \times C} \to \mathbb{R}^2.$$

The objective of this mapping is to preserve the inherent similarities of the radio channel domain within the embedded space. For example, channel similarities caused by nearby measurement locations in the physical space should also appear close to each other in the embedded space. The estimation of the CC function is carried out by training a Convolutional Neural Network (CNN) model on a dataset of channel measurements that have been cleaned and pre-processed to extract relevant features. The training yields an optimized mapping function $f^*_{\boldsymbol{\theta}}$, which is then used to predict the UE position from unseen CIR data after applying the same pre-processing steps.

A. Data Pre-Processing

Figure 2 show the pre-processing steps of the CC model. The first step is to apply inverse fast Fourier transform (IFFT) shifting to position the zero-frequency component at the center of the time-domain CIR output. Then, we remove outliers, which are the result of synchronization hardware impairments present in the testbed. All of this results in a TDoA-aligned RU-level CIR matrix $\mathbf{H}^{\text{shifted}}_{k,t}$. We normalize $\mathbf{H}^{\text{shifted}}_{k,t}$ by the normalization factor $\alpha_{\text{norm}} = \max_{k,m_k,t} (|\mathbf{h}^{\text{shifted}}_{k,m_k,t}|)$, which is computed from the training data and reused during testing to ensure consistency. Finally, we truncate the result to only contain the first C fast Fourier transform (FFT) indices. The details about each step are explained in [10]. The final result is a normalized and truncated channel matrix, expressed as:

$$\mathbf{H}^{\text{norm}}_{t} = \frac{1}{\alpha_{\text{norm}}} \begin{bmatrix} |\mathbf{h}^{\text{shifted}}_{1,t}| \\ |\mathbf{h}^{\text{shifted}}_{2,t}| \\ \vdots \\ |\mathbf{h}^{\text{shifted}}_{M,t}| \end{bmatrix} \in \mathbb{R}^{M \times C} \tag{3}$$

B. Training Phase

After collecting a substantial set of channel measurements under diverse conditions and trajectories, the data are pre-processed based on the principles from the previous section. The resulting dataset is then prepared and structured for use in training a CNN model, enabling effective feature learning and robust performance.

The model training steps are depicted in Figure 3. To train the model, we proceed as described in [10], we begin by randomly selecting pairs of CIR samples $(\mathbf{H}^{\text{norm}}_{t_i}, \mathbf{H}^{\text{norm}}_{t_j})$ as well as their corresponding TDoAs $(\Delta\hat{\tau}_{t_i}, \Delta\hat{\tau}_{t_j})$ from the dataset, with timestamps t_i and t_j if their temporal interval satisfies $|t_j - t_i| \leq \epsilon$. Otherwise, they are discarded, and new pairs are drawn.

We consider two loss functions, the first loss function $\ell^{\Delta\tau}_{t_i,t_j}$ reflects the TDoA loss, which discards the NLoS-corrupted measurements using binary masking, where a classification function, denoted as $g^*_{\boldsymbol{\phi}}$, maps normalized CIR measurements to a binary LoS/NLoS masking vector. The second loss function $\ell^{d}_{t_i,t_j}$ incorporates the displacement of the UE $\hat{d}_{i,j}$ between timestamps t_i and t_j. The displacement is assumed to be available (e.g., measured from an external sensor such as laser or inertial measurement unit (IMU)). In addition, the time interval ϵ is chosen to avoid sensor noise and bias from IMU.

While the TDoA loss function could embed the CIR into two dimensions with a global scale, the purpose of displacement loss is to smoothen the fluctuations in a moving scenario. Thus, the overall training objective integrates two terms across all T time steps, K RUs and their M_k TRPs, can be expressed as:

$$\mathcal{L} = \frac{1}{TK(M_k - 1)} \sum_{\substack{t_i,t_j=1 \\ |t_i - t_j| < \epsilon}}^{T} \sum_{k=1}^{K} \sum_{\substack{m_k=1 \\ m_k \neq m_{\text{ref}_k}}}^{M_k} \ell^{\Delta\tau}_{t_i,t_j} + \beta\, \ell^{d}_{t_i,t_j}, \tag{4}$$

where the first term $\ell^{\Delta\tau}_{t_i,t_j}$ is the pair-wise TDoA measurement loss, the second term $\ell^{d}_{t_i,t_j}$ is the displacement measurement loss, and β controls the relative weight of the displacement term.

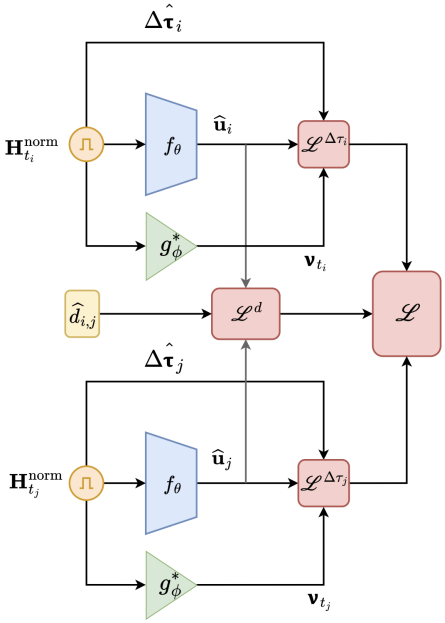

Fig. 3: CC Training with CIR and TDoA+displacement [10]

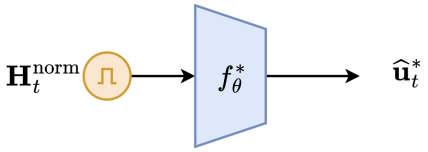

Fig. 4: CC Testing [10]

C. Testing Phase

During training, our model learns to jointly capture both TDoA and displacement information in a mixed LoS/NLoS scenario. It is important to note that the testing data are entirely unseen and have not been used during training. Consequently, in the testing phase, as shown in Figure 4, the optimized mapping function $f_{\boldsymbol{\theta}}^*(\cdot)$ operates solely on the pre-processed CIR inputs, without requiring explicit displacement, TDoA feature annotation or NLoS masking.

IV. Implementation

In this section, we introduce and detail the implementation of the custom O-RAN service model E2SM-SRS and define the "SRS Positioning" RAN function, which supports the RIC REPORT service, which is used to expose the SRS frequency-domain channel estimates inside the RIC indication message via the E2 interface. In this work, the service model will be leveraged by a localization xApp that exploits and processes the RIC indication message in order to perform continuous inference on a pre-trained CC model. The E2SM-SRS and xApps are implemented in FlexRIC and integrated with the E2 agent in OAI. All the code is available in the *srs_sm* [14] branch in FlexRIC and *srs_e2* [15] branch in OAI. [1]

A. OpenAirInterface Overview

OpenAirInterface is an open-source project that provides a software-defined implementation of 4G and 5G 3GPP systems. It is O-RAN compliant as well. OAI includes the CN, the RAN and the UE. The OAI platform consists of several coordinated projects, including the RAN, CN, Continuous Integration/Continuous Delivery (CI/CD) pipelines, and Operations, Administration and Maintenance (OAM). The platform is maintained by the OpenAirInterface Software Alliance (OSA) which defines roadmaps of each project, promotes the software and supports the community.

OAI supports different functional splits such as the F1 split between the CU and the DU and the E1 split between the CU user plane (CU-UP) and the CU control plane (CU-CP). It also supports the O-RAN 7.2 fronthaul split as well as the eCPRI split 8, in order to work with third-party software-defined radios (SDRs) or RUs.

OAI also supports the Functional Application Platform Interface (FAPI) [16], which is an interface specified by the Small Cell Forum (SCF) that shows the interplay between Layer 1 (L1) and Layer 2 (L2). It allows in-line acceleration with the integration with hardware-accelerated or Graphics Processing Unit (GPU) based physical layer implementations, such as NVIDIA's Aerial L1 stack [17] [18].

Additionally, OAI supports the E2 interface, which allows the OAI gNB, CU, or DU to act as an E2 agent for integration with a Near-RT RIC, in accordance with O-RAN architecture. This enables real-time monitoring, control, and optimization of the RAN through xApps. For more details on OAI, we refer the reader to [8].

B. FlexRIC

FlexRIC is a software development kit (SDK) that provides an O-RAN compliant Near-RT RIC and xApp SDK. It consists of server and agent library. It was designed to follow the 5G principles of ultra-lean design and it is implemented with zero-overhead principle [7]. FlexRIC supports different versions of the E2 Application Protocol (E2AP) (v1.0/2.0/3.0) and implements some O-RAN specified Service Models; the Key Performance Measurement (KPM) and Radio Control (RC) service models. It also allows the implementation of some custom service models including those for the Medium Access Control (MAC), Packet Data Convergence Protocol (PDCP), and Radio Link Control (RLC) layers. The xApps can be developed in different high-level programming languages such as C, C++, and Python, offering flexibility and rapid development.

C. E2SM-SRS

The custom E2SM-SRS service model provides the semantic description of specific fields which contain the data to be

[1] If the branches do not exist anymore, use the *dev* branch in FlexRIC and the *develop* branch in OAI

transported over the E2 interface. In our implementation, we leverage the FAPI SRS procedure from SCF [16] and extend it to the RIC, because the exchanged messages in this procedure contain the SRS channel estimates. As specified by SCF, the FAPI SRS.indication message contains fields related to the SRS including a sub-sampled version of the SRS frequency domain channel estimates. For our purposes, the E2SM-SRS exposes a modified version of the SRS.indication message, which instead contains the channel estimates with the full resolution. In total, the RIC indication message contains the SRS.indication message and a UE identifier to distinguish between multiple UEs. Figure 5 illustrates the message flow between the RIC and the other RAN entities using the E2SM-SRS. The RIC establishes connection with the E2 agent (monolithic gNB or DU in this case) via the E2 Setup procedure. To enable the FAPI SRS procedure, the L2 should include a SRS PDU in UL_TTI.request. After receiving the SRS from the UE, the L1 will process and forward the SRS response in the SRS.indication message. The reception of the SRS.indication message by the L2 constitutes the event trigger for our RIC Report Service. Whenever this trigger is satisfied, the RIC indication message gets forwarded to the Near-RT RIC via the E2 interface.

D. Localization xApp

In our framework, the Near-RT RIC hosts an xApp that performs localization inference. As an example, we implemented a specialized localization xApp in C++, that leverages our custom E2SM-SRS and performs continuous inference on a pre-trained CC model. The xApp is used to evaluate the CC model using the Firecell GEO5G testbed at EURECOM. After the xApp receives the RIC indication message, it unpacks the fields of the SRS.indication message. The next step is the pre-processing of the SRS channel in order to prepare it for ML-inference. We implemented the steps described in III using *libtorch*. Then, the xApp performs the inference using a trained model. Moreover, the CC predictions are enhanced with a moving average filter. Furthermore, the xApp integrates a demonstrator, in the form of a real-time graphical user interface(GUI) that plots the SRS channel and the position estimates in a two-dimensional representation of the testbed map.

E. E2 Agent emulator support

For prototyping and testing purposes, we extended the O-RAN compliant E2 agent emulators provided by FlexRIC to support our new E2SM-SRS for AI/ML-based localization. In our implementation, the E2 emulator reads CSI samples from the EURECOM 5G SRS CIR dataset [19] and forwards them to the Near-RT RIC. This can easily be configured to use any other dataset. This setup enables testing new algorithms within an O-RAN-compatible framework, providing a useful intermediary step before deploying a full end-to-end system. For reproducibility, we provide a detailed tutorial in [20].

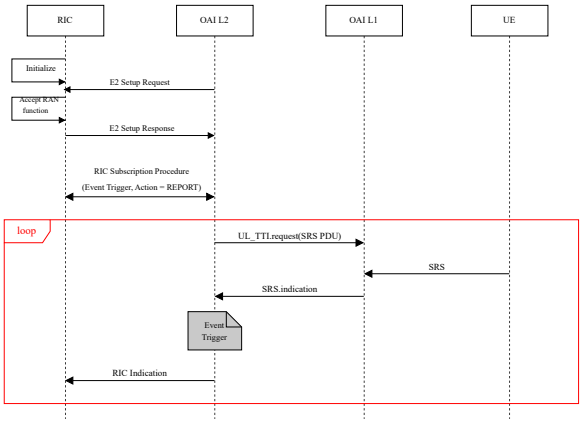

Fig. 5: Message flow between the RIC and the RAN entities

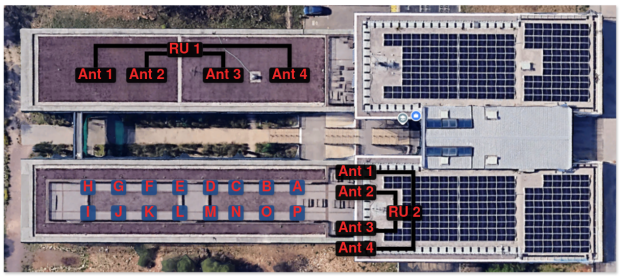

Fig. 6: Firecell GEO-5G testbed at EURECOM: Deployment of 2 O-RUs each with 4 distributed TRPs on the south terrace and the north rooftop of EURECOM, with a testing area of size 50×10 on the north terrace

V. Experimental Evaluations

A. Setup

To validate the proposed framework, we conducted experiments in real-world conditions using the Firecell GEO-5G testbed at EURECOM. The testbed is built on top of EURECOM's OpenXG platform, which provides high-speed fiber-connected computing and switching infrastructure. This environment supports virtual 5G deployments with USRPs and O-RAN radios, and integrates OAI for running virtual network functions.

Thanks to our recent contributions to OAI RAN [12], the testbed can flexibly operate under either a single-gNB with multiple RUs, or a multi-gNB with multiple RUs architecture. To evaluate the new localization features in OAI, we deployed two VVDN O-RAN RUs [21], provided by Firecell [22], configured with a single gNB (CU-DU) and integrated into the OpenXG infrastructure.

Each RU is equipped with four distributed Panorama directional antennas [23], mounted on the roof railings and connected through low-loss cables. This results in a total of eight TRPs, providing coverage over a 50×10 m testing area on the north terrace of the EURECOM building (see Figure 6). There are 16 test points (A to P) distributed in the testing area. All test points accurate locations in a local Cartesian

coordinate are calculated using true distances between them and the antennas by a laser ranging tool. The reference for our local coordinate is TRP 1 on RU1.

In our configuration, we use 5G New Radio (5G NR) band n77 with 100 Mhz maximum bandwidth, in time division duplex (TDD) mode using 30 kHz subcarrier spacing. The CU and DU run on the same server. The CN runs on a separate server in a docker environment. FlexRIC, which includes, the Near-RT RIC and the xApp (with the GUI) runs on bare metal on a separate machine connected to the network. Deploying FlexRIC in a docker environment is also possible.

B. Results

This section presents the results for the performance evaluation of our O-RAN-based localization framework. We assess the accuracy of CC, followed by an analysis of the latency. The results are compared with the 3GPP uplink time difference of arrival (UL-TDoA) positioning.

1) Accuracy: Two scenarios are considered: one for a static UE and one for a moving UE. The first experimental scenario involves placing one commercial off-the-shelf (COTS) UE on each test point with a fixed height using a tripod. The xApp is then executed to perform the CC inference. We store the CC predictions on a *csv* file in real-time. The Euclidean distance between the CC predictions and the ground truth measurements represents the error in the measurements. We consider the 90th percentile of the error and the mean absolute error (MAE) as performance metrics of the positioning accuracy.

In the moving scenario, a handheld UE moves in a random trajectory along the test points, and the CC estimations are illustrated in real-time over the GUI. This provides valuable qualitative insights into the model's behavior and allows for observation of how changes in the wireless environment affect the CC predictions.

The box plots in Figure 7 visualize the distributions of the error in all 16 test points. The lower whisker is set at the 10th percentile, and the higher whisker is set at the 90th percentile. We observe different error distributions across the test points with varying skewness. For most test points, the median error, represented by the orange line, lies in the expected range under 4 m in most of the test points except test points H,I and O.

Figure 8 shows the CC predictions in a moving scenario for a UE taking the following trajectory: $B \rightarrow C \rightarrow D \rightarrow E \rightarrow L \rightarrow M \rightarrow N \rightarrow O \rightarrow P \rightarrow A \rightarrow B$. For error analysis in this moving scenario, existing tools such as those based on real-time kinematic positioning (RTK) can deliver cm-level positioning accuracy which can be considered as ground truth for outdoor deployments. However, it remains complicated to evaluate the positioning error. This is mainly due to practical impairments, such as synchronizing the CC predictions with RTK measurements. In future developments, more reliable and robust methods will be investigated to assess the positioning accuracy for a moving UE.

We also compare the accuracy of CC with a classical 3GPP UL-TDoA protocol employing particle swarm optimization (PSO) algorithm to compute the position [12]. Table I shows

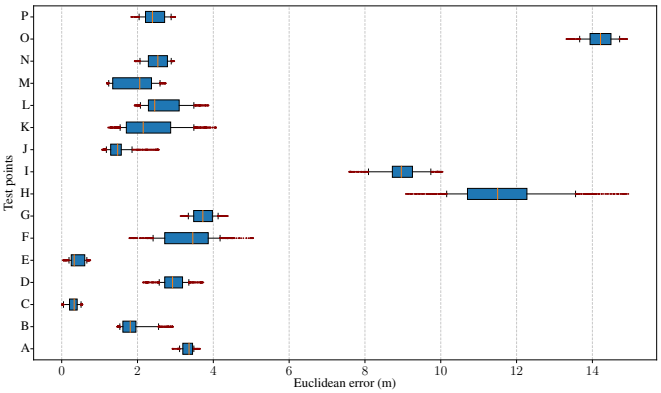

Fig. 7: Box plots of Euclidean error of all test points, where the whiskers are set to the 10th and 90th percentile, the orange line shows the median and the red dots represent the outliers.

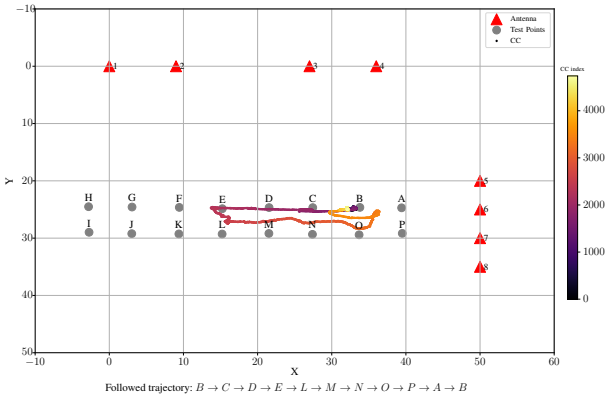

Fig. 8: CC UE tracking in a moving scenario

the comparison of the MAE for the same 16 test points. It can be observed that UL-TDoA performs better for most of the test points, but it should be noted that the measurements were done at different times and it could be that the environment has changed in between. This highlights the very important fact that neural networks for localization are very site and environment specific and should be regularly retrained.

2) Latency: Finally, we evaluate the latency of the proposed O-RAN framework and compare it with that of the 3GPP NRPPa-based positioning architecture [12]. Latency is measured over multiple trails by recording timestamps at the relevant entities. In the O-RAN framework, timestamps are recorded at the E2 agent, and xApp. For the 3GPP architecture, timestamps are recorded at the location management function (LMF) and the overall latency is measured from the initiation of location request until the reception of the positioning result. In our experimental setup, all the involved servers are synchronized using Precision Time Protocol (PTP) in order to ensure timing consistency.

As explained in Section IV, the RIC indication procedure is event-driven. Thus, the end-to-end latency is defined as the

TABLE I: Comparison of the MAE (in meters) for the test points

Test point	MAE_{CC}	$MAE_{UL\text{-}TDOA}$
A	3.33	1.88
B	1.87	1.13
C	0.30	0.82
D	2.94	0.83
E	0.40	0.95
F	3.35	0.55
G	3.73	0.92
H	11.64	0.56
I	8.95	0.60
J	1.49	0.83
K	2.33	0.82
L	2.65	1.20
M	1.94	0.64
N	2.51	0.68
O	14.19	0.97
P	2.43	1.98

elapsed time between the event trigger —corresponding to the reception of a FAPI SRS indication message— and the delivery of the CC prediction produced by the xApp.

To provide a detailed analysis, we decompose the overall latency into two components. First, the RIC indication message latency is measured as the time elapsed from when the E2 agent generates and sends the RIC indication message, until its reception by the xApp. Second, the xApp processing latency is measured separately, capturing the unpacking of the SRS.indication message, the preprocessing, and the CC inference. The end-to-end latency corresponds to the sum of these two components. In addition, the inference latency is measured in isolation to quantify its contribution to the total end-to-end latency.

Experimental results show an average end-to-end latency of 7.749 ms, out of which 4.914 ms corresponds the processing latency. The CC inference latency is 4.230 ms, accounting for around 86% of the processing latency.

For comparison, we also measured the latency of the 3GPP UL-TDoA architecture where the LMF is using PSO as positioning algorithm [12]. Unlike the O-RAN architecture, the 3GPP positioning protocol is based on request and response: the positioning process begins when an external Application Programming Interface (API) sends a location request of a UE to the LMF. In the implementation, this is issued via a single-line `curl` command, which sends an Hypertext Transfer Protocol (HTTP) POST request to the determine location API. After computing the UE location, the LMF returns a response.

The overall latency is measured by collecting timing information provided by the `curl` command. As with the RIC approach, we also separately measure the latency of the PSO algorithm. However, it is important to note that conducting a fair comparison between the two approaches remains challenging. A substantial portion of the latency in the 3GPP architecture is spent waiting for the SRS measurement response. Consequently, the measurements obtained via the `curl` command represent an upper bound on the end-to-end latency.

The results of the 3GPP NRPPa-based approach show that the average overall latency is 32.125 ms while the average latency of the PSO is 0.240 ms. While the NRPPa protocol latency is substantially higher than that observed in the RIC procedure, the PSO computation is significantly lower than the CC inference time in the xApp. However, it should be noted that we did not seek to optimize the performance of CC and there is room for improvement by using a different implementation on a more efficient inference engine, or GPU acceleration.

VI. Conclusions

This paper provided an O-RAN based framework for the integration of real-time AI/ML-based localization inference in 3GPP and O-RAN compliant 5G systems and beyond. The implementation is fully open-source, based on OAI and FlexRIC, provides a new custom E2SM that successfully transports the UL-SRS channel estimates to the NearRT-RIC via the E2 interface. Localization is performed in an xApp hosted in the Near-RT RIC. The approach was integrated with a CC xApp and validated in real-world conditions using the Firecell GEO5G testbed at EURECOM.

The purpose of the paper was to show how the O-RAN architecture can be leveraged to perform AI/ML-based positioning. The algorithms and implementations of the xApp used in this paper is just an example and not the main contribution of this work.

As future work, we plan on extending this framework to multi-cell deployments, integrating other ML methods, as well as extending to sensing. We also consider integrating the Near RT-RIC with a Non RT-RIC to perform the AI/ML model training.

References

[1] 3rd Generation Partnership Project (3GPP), "3GPP TS 38.305 V18.4.0: NR; Functionality for UE positioning," 3rd Generation Partnership Project (3GPP), Tech. Rep. TS 38.305 V18.4.0, 2025, release 18.

[2] L. Italiano, B. Camajori Tedeschini, M. Brambilla, H. Huang, M. Nicoli, and H. Wymeersch, "A tutorial on 5g positioning," *IEEE Communications Surveys & Tutorials*, vol. 27, no. 3, pp. 1488–1535, 2025.

[3] A. Nessa, B. Adhikari, F. Hussain, and X. N. Fernando, "A survey of machine learning for indoor positioning," *IEEE Access*, vol. 8, pp. 214 945–214 965, 2020.

[4] 3rd Generation Partnership Project (3GPP), "3GPP TR 38.843 V19.0.0, Study on Artificial Intelligence (AI)/Machine Learning (ML) for NR air interface," 3rd Generation Partnership Project (3GPP), Tech. Rep. TR 38.843 V19.0.0, 2025, release 18.

[5] O-RAN Work Group 2, "AI/ML workflow description and requirements," O-RAN ALLIANCE, Tech. Rep. O-RAN.WG2.AIML-v01.03.

[6] O-RAN Work Group 1 (Use Cases and Overall Architecture), "Use Cases Detailed Specification," O-RAN ALLIANCE, Tech. Rep. O-RAN.WG1.TS.Use-Cases-Detailed-Specification-R004-v16.00.

[7] R. Schmidt, M. Irazabal, and N. Nikaein, "Flexric: an sdk for next-generation sd-rans," in *Proceedings of the 17th International Conference on Emerging Networking EXperiments and Technologies*, ser. CoNEXT '21. Association for Computing Machinery, 2021, p. 411–425.

[8] F. Kaltenberger, T. Melodia, I. Ghauri, M. Polese, R. Knopp, T. T. Nguyen, S. Velumani, D. Villa, L. Bonati, R. Schmidt, S. Arora, M. Irazabal, and N. Nikaein, "Driving innovation in 6G wireless technologies: The openairinterface approach," *Computer Networks*, vol. 269, p. 111410, 2025.

[9] 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.
[10] M. Ahadi, O. Esrafilian, F. Kaltenberger, and A. Malik, "TDoA-based self-supervised channel charting with NLoS mitigation," arXiv:2510.08001, 2025. [Online]. Available: https://arxiv.org/abs/2510.08001
[11] R. Mundlamuri, R. Gangula, F. Kaltenberger, and R. Knopp, "5g nr positioning with openairinterface: Tools and methodologies," in *2025 20th Wireless On-Demand Network Systems and Services Conference (WONS)*, 2025, pp. 1–7.
[12] Malik, Adeel and Ahadi, Mohsen and Kaltenberger, Florian and Warnke, Klaus and Thinh, Nguyen Tien and Bouknana, Nada and Thienot, Cedric and Onche, Godswill and Arora, Sagar, "From Concept to Reality: 5G Positioning with UL-TDoA in OpenAirInterface," in *IEEE INFOCOM 2025 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2025, pp. 1–6.
[13] M. Ahadi, A. Malik, O. Esrafilian, F. Kaltenberger, and C. Thienot, "Experimental insights from openairinterface 5G positioning testbeds: Challenges and solutions," in *ACM Workshop on Wireless Network Testbeds, Experimental evaluation & Characterization (WiNTECH)*, Hong Kong, China, 2025, also availabe as arXiv preprint arXiv:2508.19736. [Online]. Available: https://arxiv.org/abs/2508.19736
[14] https://gitlab.eurecom.fr/mosaic5g/flexric/-/tree/srs_sm?ref_type=heads.
[15] https://gitlab.eurecom.fr/oai/openairinterface5g/-/tree/srs_e2?ref_type=heads.
[16] SCF, "5g fapi: Phy api specification," Tech. Rep.
[17] D. Villa, I. Khan, F. Kaltenberger, N. Hedberg, R. S. Da Silva, A. Kelkar, C. Dick, S. Basagni, J. M. Jornet, T. Melodia, M. Polese, and D. Koutsonikolas, "An open, programmable, multi-vendor 5g o-ran testbed with nvidia arc and openairinterface," in *IEEE INFOCOM 2024 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2024, pp. 1–6.
[18] D. Villa, I. Khan, F. Kaltenberger, N. Hedberg, R. S. da Silva, S. Maxenti, L. Bonati, A. Kelkar, C. Dick, E. Baena, J. M. Jornet, T. Melodia, M. Polese, and D. Koutsonikolas, "X5g: An open, programmable, multi-vendor, end-to-end, private 5g o-ran testbed with nvidia arc and openairinterface," *IEEE Transactions on Mobile Computing*, vol. 24, no. 11, pp. 11 305–11 322, 2025.
[19] M. Ahadi, F. Kaltenberger, O. Esrafilian, and A. Malik, "EURECOM 5G SRS Dataset," 2025. [Online]. Available: https://dx.doi.org/10.21227/t8ya-z141
[20] https://gitlab.eurecom.fr/mosaic5g/flexric/-/blob/srs_sm/examples/xApp/c/positioning/README.md?ref_type=heads.
[21] VVDN Technologies, "5G NR Enterprise Radio Units," https://www.vvdntech.com.
[22] FIRECELL, "Private 4G/5G Networks," https://www.firecell.io/.
[23] Panorama Antennas, "Directional Antennas Product Information," https://www.panorama-antennas.com/.