MIMO-RIC: RAN Intelligent Controller for MIMO xApps

Sesha Sai Rakesh Jonnavithula[1] , Ish Kumar Jain[1,2], and Dinesh Bharadia[1]
[1]University of California San Diego, CA, USA, [2] Rensselaer Polytechnic Institute, Troy, NY, USA

Abstract

The adoption of MIMO technology in wireless networks enhances spectral efficiency and enables novel functionalities such as wireless sensing and localization. These functionalities can be enabled by Open-RAN architecture to provide high computation, memory, and data-driven inference through a RAN Intelligent Controller (RIC). However, existing RICs focus mainly on higher network layers and lack essential PHY layer functionalities for MIMO. We present MIMO-RIC, an open-source RAN intelligent controller tailored for MIMO applications. We enable the streaming of extensive 3D wireless channel measurements across antennas, subcarriers, and time, from RAN to MIMO-RIC to develop various MIMO apps like beamforming and localization. We implemented MIMO-RIC on the srsRAN open-source platform using ZeroMQ messaging system for efficient, low-latency communication. Our over-the-air experiment setup consisting of USRP radios and commercial user equipment demonstrates effective jammer monitoring, nulling, and user localization applications.

CCS Concepts

• **Hardware** → **Wireless devices**; • **Networks** → **Physical links**; **Wireless access points, base stations and infrastructure**.

Keywords

MIMO, O-RAN, RIC, Cellular Stack, srsRAN, Beam-Nulling, Anti-Jamming, Software-Defined Radios, Localization

ACM Reference Format:
Sesha Sai Rakesh Jonnavithula[1] , Ish Kumar Jain[1,2], and Dinesh Bharadia[1], [1]University of California San Diego, CA, USA, [2] Rensselaer Polytechnic Institute, Troy, NY, USA . 2024. MIMO-RIC: RAN Intelligent Controller for MIMO xApps . In *The 30th Annual International Conference on Mobile Computing and Networking (ACM MobiCom '24), November 18–22, 2024, Washington D.C., DC, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3636534.3701548

ACM MobiCom '24, November 18–22, 2024, Washington D.C., DC, USA

ACM ISBN 979-8-4007-0489-5/24/11.
https://doi.org/10.1145/3636534.3701548

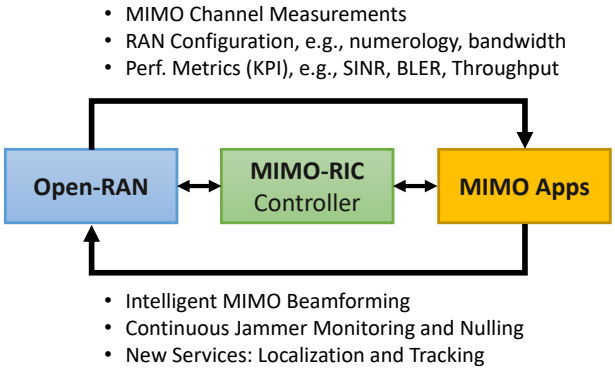


Figure 1: MIMO-RIC is a controller platform that can simplify MIMO App development in Open-RAN.

1 Introduction

Today, the use of multiple antennas in Radio Access Networks (RAN), enabled by MIMO technology, not only enhances spectral efficiency through multiple parallel streams but also enables new possibilities in wireless sensing, such as user localization and tracking, multi-user beamforming, interference/jammer monitoring and nulling. These functionalities can be further enhanced by data-driven techniques, but are often secluded from RAN due to limitations in RAN's computation or data storage capabilities [1–5]. RAN must handle PHY layer tasks efficiently, often within less than one Transmission Time Interval (TTI) or 1 ms, to avoid packet drops, leaving little room for compute-intensive applications.

Open RAN offers an avenue for application development within RAN through RAN intelligent controllers (RICs). Deployed outside the RAN environment, RICs possess considerable compute and storage capabilities and operate on more relaxed time-scales, ranging from 10 ms to 1 sec for near-real-time RICs [6]. This setup allows for the implementation of novel data-driven, compute-intensive techniques directly into operational RAN. However, most industry-developed RICs are proprietary and inaccessible to academic research [7, 8]. Among open-source RICs available to the research community, such as FlexRIC [9], EdgeRIC [10], and CloudRIC [11], they primarily focus on higher layers of the networking stack and generally lack essential PHY layer functionalities needed for MIMO operations.

To facilitate compute-efficient data-driven MIMO techniques, Open RAN presents an opportunity to facilitate application development using RAN intelligent controllers (RICs). RICs are deployed outside of RAN environment with considerable compute and storage capabilities and operate on more relaxed time-scales, ranging from 10 ms to 1 sec for near-real-time RICs [6]. This setup allows for the implementation of novel data-driven, compute-intensive techniques directly into operational RAN. However, most industry-developed RICs are proprietary and inaccessible to academic research [7, 8, 12]. Among open-source RICs available to the research community [9, 10, 10], they primarily focus on higher layers of the networking stack and generally lack essential PHY layer functionalities needed for MIMO operations.

In this paper, we present MIMO-RIC, that addresses this gap by developing an intelligent controller for MIMO applications as shown in Figure 1. The key requirement for MIMO-RIC is the ability to stream wireless channel measurements from RAN to the controller, as channel information is crucial for MIMO applications such as beamforming and localization. Further, the channel should be accessible across multiple antennas, frequency subcarriers, and time slots to enable advanced algorithm development. The second requirement for MIMO-RIC is to monitor various network performance indicators such as SINR, BLER, and Throughput to continuously detect interferers or jammers and to evaluate the performance of new beamforming techniques. Finally, the controller should be able to create policies that dictate how the RAN should be configured or updated to enhance MIMO performance. These policies could involve beamforming weights that RAN applies directly to multi-antenna streams or higher-level decisions that prioritize among various beamforming or beam-nulling algorithms at any given time.

With these requirements in mind, we developed and demonstrated MIMO-RIC using the srsRAN open-source platform [13] and modified the RAN stack for MIMO applications. To facilitate bi-directional communication between RAN and the controller, we adopted a ZeroMQ (ZMQ) based message passing library similar to EdgeRIC [10] and BeamArmor [14]. However, ZMQ interfaces built for one application are not easily transferable to others due to differences in data types, structures, and access points within the RAN stack. For example, BeamArmor [14] developed a ZMQ interface to stream raw IQ data from each antenna in RAN to the controller, but it is limited by low-frequency data transfer (on the order of seconds) and high transfer delays, which could even crash the RAN stack. Instead of raw IQ data, MIMO-RIC streams processed uplink channel estimates to reduce streaming latency. This implementation involved isolating the PUSCH processing in the srsRAN stack, identifying specific DMRS signals that provide the channel data, and inserting the ZMQ interface at an appropriate location to ensure seamless channel data transfer to the controller. We share our experiences with various design choices and insights. For instance, implementing a ZMQ interface to stream per-antenna, per-subcarrier (or resource element) data incurred high latency and overhead due to excessive ZMQ flows. To mitigate this, we combined the channel data across all four antennas and all subcarriers, and stream the aggregated channel data to the controller to achieve low latency and overhead. Additionally, we deployed ZMQ in "conflate" mode, which reduces the space in shared memory, particularly when the controller operates on a slower time scale than RAN.

We built a testbed for MIMO-RIC with software-defined radios such as USRP N310, B210, and PlutoSDR, srsRAN gNB software stack, and a OnePlus 8 and Google Pixel phone as user-equipment. We particularly demonstrated three use cases for jammer monitoring, jammer nulling and user angular localization using MIMO antennas. We monitor for jammer uisng variation in performance metrics such as SINR, BLER, and throughput. We then estimate the jammer channel and apply per-subcarrier nulling to mitigate the impact of wideband interference. We finally show accurate angle estimation using FFT-based techniques for angle estimation. Various indoor and outdoor experiments demonstrate the effectiveness of these use cases. We believe our open-source platform will be a valuable tool for the research community to develop more advanced data-driven use cases for these and other MIMO applications.

In summary, we make the following contributions by building MIMO-RIC:

- A platform to stream MIMO channel across antenna, time, and frequency to the controller
- Supports up to 4 antennas in srsRAN 5G.
- Monitors SINR, Throughput, and BLER in real-time to detect jammers.
- Demonstrates MIMO applications such as beam-nulling and direction of arrival estimation.

2 Motivation and Related Work

2.1 Why O-RAN for MIMO Applications?

We provide a brief background on O-RAN and motivate for why it will enhance the development of MIMO applications. The architecture of Open-RAN is divided into three main components, the Radio Unit (RRU), the Distributed Unit (DU) and the Centralized Unit (CU). The RRU handles the RF processing and is located at the cell site. The DU takes care of the real-time processing tasks such as scheduling and beamforming, while the CU, which is further split into control plane (CU-CP) and User plane (CU-UP), manages the higher-layer protocols and non-real-time processes. The RAN Intelligent Controller (RIC) in O-RAN is a component

that enhances network flexibility and performance through constant optimization. It is typically deployed on general-purpose, commercial off-the-shelf (COTS) servers within the network's cloud infrastructure and has interfaces to communicate with the various network layers in O-RAN (eg: PHY, MAC, RLC etc.). The RIC architecture offer several significant advantages for modern network management that are typically not available in RAN, such as–.

- **Near-Real-Time optimization Capabilites:** The RIC enables near-real-time operations around the PHY layer and MIMO applications, crucial for maintaining high performance and efficiency in complex radio environments.
- **Computational Capabilites:** RIC supports the execution of compute-heavy algorithms involving AI and machine learning tasks, which require dedicated hardware to run effectively. This allows for more sophisticated and adaptive network optimizations.
- **Data Storage Capabilites:** Monitoring applications benefit from the RIC's ability to maintain a history of data and track changes within the network. For instance, interference monitoring relies on this historical data to detect and mitigate issues that could degrade network performance.
- **Interfacing External Hardware/Information:** External information that needs to be integrated into the ORAN system can be efficiently managed and acted upon through the RIC, making it a central hub for data-driven decision-making.

These capabilities are realized through xApps which are applications that the network operators can deploy onto the RIC to interface with O-RAN and collect data for various applications broadly classified as monitoring and control applications. Monitoring Apps are used for inference tasks such as estimating the presence of a jammer or predicting user location using multiple antennas. On the other hand, control Apps provide policies that can change certain network states such as RAN configuration of adoption of new beamforming/nulling weights. This way, novel data-driven and compute-extensive MIMO algorithms can be adopted in RAN using the proposed MIMO-RIC architecture.

2.2 Related Work

RIC Platforms: With the development of O-RAN, many industries race to build their own RIC, such as VMware RIC [12], Microsoft [7], Nokia Bell Labs [8], Mavenir [15], Juniper Networks [16], etc. However, these RICs are proprietary or are built on non-open-source RAN platforms and therefore not available to the research community. FlexRIC [9] is an open-source standard-compliant RIC solution that works with open-source RAN stacks such as srsRAN [13] and OpenAir Interface [17]. However, FlexRIC only offer Apps development at higher layers of networking stack, but not for the PHY layer. Similarly, CloudRIC [11] develops hardware accelerators for multi-DU settings. There are other platforms that focuses on mmWave networks [18–23], while we focus on sub-6 MIMO setting.

EdgeRIC [10] and BeamArmor [14, 24] are closest to MIMO-RIC as they are built on srsRAN with similar ZMQ based interface, but they lack MIMO capabilities in many ways. EdgeRIC [10] only supports SISO links; BeamArmor [14, 24] supports MIMO, but it can only stream raw IQ data instead of wireless channel data to the RIC that limits its usage and suffer from high latency overhead. Moreover, BeamArmor is built on srsRAN 4G stack with support form only two antennas in uplink. In contrast, MIMO-RIC provides DMRS channel streaming across all subcarriers and up to 4 antennas and is built on the 5G srsRAN software stack.

App Development: Most xApps are built for higher layers in network stack such as network slicing [25, 26], Network automation [27, 28], and multi-user resource scheduling [10, 29, 30]. In contrast, we demonstrate PHY-layer MIMO-based apps such as user localization, jammer monitoring and nulling.

Other MIMO platforms: Bigstation [31], Agora [32], and Hydra [33] provide extremely large scale Massive MIMO testbed with parallel threading and multi server implementation—e.g. Hydra [33] supports 150x32 antennas—however, they only implement PHY layer of network stack and does not address compliance with end-end O-RAN stack.

3 MIMO-RIC Design

MIMO application primarily depends on channel estimates provided by the RAN. We developed MIMO-RIC to facilitate MIMO applications by granting access to channel estimates derived from uplink channel estimates in 5G NR. By providing access to these estimates across all receive antennas, we enable the development of various applications focused on advanced beamforming techniques, jammer detection and nulling, and localization. In this section, we discuss various design choices, 5G NR signaling, and integration with srsRAN. We will also discuss the jammer nulling application in detail.

3.1 Near-Real Time Design with O-RAN

The overall architecture for MIMO-RIC is shown in Figure 2. MIMO-RIC controller is based on srsRAN, an open-source RAN solution that adheres to 3GPP and O-RAN alliance standards. Communication between the RAN and the controller is managed using the ZMQ publisher-subscriber architecture, facilitated by the ZMQ library. In this setup, a ZMQ publisher socket can transmit messages to multiple subscriber sockets,

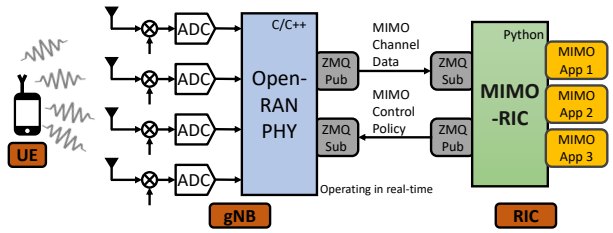


Figure 2: MIMO-RIC Architecture: Consists of ZMQ publisher/subscriber messaging between RAN and RIC to stream MIMO channel data for MIMO Apps development.

while subscriber sockets receive messages from specific publishers. This architecture allows for flexible data streaming, enabling new publishers and subscribers to be added without disrupting existing components. Additionally, it decouples the controller from the RAN, as publishers and subscribers do not need to be aware of each other's presence, thus supporting modular development.

In our implementation, we adhere to the O-RAN policy of independent controller and RAN operation by employing the ZMQ Pub and Sub architecture and interfacing them with the upper layers of the RAN stack such as the PUSCH processor and equalization. This architecture helps us meet the stringent timing requirements of the lower layers of the cellular stack. To minimize disruption to these timing requirements, we place the ZMQ sockets in the upper layers of the srsRAN stack. This approach avoids the complexities associated with timing in the lower layers, such as the PHY layer. It also provides the flexibility to incorporate additional features, such as conditional streaming, where logical operations are implemented in the upper layers to manage the frequency of data transfer into the ZMQ buffers. Such features would be challenging to implement if we were streaming data directly from the PHY layer due to the stringent timing constraints and the computational load of these operations.

3.2 Seamless integration into 5G NR Stack and Access to Channel Parameters

The localization and jammer nulling applications developed on the MIMO-RIC platform required the latest channel estimates from all four receive antennas to be streamed to the controller with minimal delay. To seamlessly integrate into the 5G NR cellular stack, we opted to access these channel values at the Physical Uplink Shared Channel (PUSCH) processor within the upper layer of the srsRAN stack. In 5G NR, the PUSCH is utilized by User Equipment (UE) to transmit user data to the base station and includes Demodulation Reference Signals (DMRS). DMRS are known reference signals that assist the base station in estimating channel characteristics, thus correcting signal distortions caused by fading, interference, and noise. These DMRS transmissions, which occur more frequently than other 5G reference signals during active data transmission, provide a detailed view of channel characteristics over time. The channel estimates derived from DMRS are interpolated across all frequency subcarriers (or resource elements) allocated to the UE, offering a comprehensive picture of channel properties in both time and frequency domains.

For our applications, it was crucial to access the most recent channel estimates at the RIC in near-real-time rather than relying on historical data. This requirement led us to use ZMQ sockets in "Conflate" mode, where the ZMQ buffers retain only the most recent channel estimates and stream them to subscribers on-demand. This approach significantly reduces timing and memory overhead while ensuring the controller receives the necessary channel data. We optimized data structures to enable the channel estimates from all active Rx antennas to be written to the ZMQ buffer in a single operation per 5G NR slot. On the controller side, we read these values in a single operation per slot and forward them for further processing.

3.3 MIMO Nulling Matrix Calculation for Jammer Mitigation

We present a methodology to mitigate the effects of a wideband jammer. that creates significant interference, introducing noise into the User Equipment (UE) signals. This interference reduces the signal-to-interference-plus-noise ratio (SINR) of the UL traffic, resulting in a higher block error rate (BLER) and a significant drop in the cellular network's throughput. Our proposed anti-jamming technique involves creating a beam-nulling at the receiver antennas that effectively nullifies the signal received from the jammer. This is achieved by calculating a precoding matrix, also known as a nulling matrix, which applies specific weights to the incoming UE data. The nulling matrix is derived from channel estimates obtained by the Physical Uplink Shared Channel (PUSCH) processor within the MIMO-RIC.

To illustrate, let's denote $\mathbf{h}_u$ as the channel estimate at the gNB when the UE is connected without the jammer, and $\mathbf{h}_t$ as the interference observed when the UE is connected with the jammer active. The channel for the jammer can then be estimated as $\mathbf{h}_j = \mathbf{h}_t - \mathbf{h}_u$. In our scenario, let $\mathbf{h}$ represent the nulling vector for a single subcarrier. Given that we have four receiver antennas on our gNB, $\mathbf{h}$ would be a 4x1 vector. To calculate $\mathbf{h}$, we formulate an optimization function that maximizes the product $\mathbf{h} * \mathbf{h}_u$ while minimizing $\mathbf{h} * \mathbf{h}_j$. Listing 1 below shows the pseudocode for this optimization in MATLAB as follows:

Listing 1: CVX Optimization Block

```
% Optimize using CVX
cvx_begin quiet
    variable h(4) complex
    minimize(norm(hu' * h - 1))
    subject to
        hj' * h == 0.1;
cvx_end
```

Once $\mathbf{h}$ vectors are computed across all frequency subcarriers, they are applied to the incoming UE signals. This accumulation results in a matrix $\mathbf{h}_n$ that acts as an equalization matrix, minimizing the received jammer signal while maximizing the desired UE signal. The nulling matrix is calculated at the controller once channel estimates are available from the RIC, and it is then sent back to the RAN for application in the equalization process at the upper layers.

However, in scenarios where the channel conditions change rapidly, such as in high-mobility environments, the accuracy of our anti-jamming technique may decrease because the nulling matrix would be based on outdated channel estimates. Nonetheless, this method is more precise than using a single nulling vector for the entire wideband, as it accounts for frequency-selective fading across subcarriers. Channel prediction and extrapolation techniques can further enhance the channel quality under high mobility.

4 Implementation with srsRAN 5G

We implemented MIMO-RIC with srsRAN 5G software stack and address various challenges associated with srsRAN. We also provide details on our over-the-air experimental testbed.

4.1 Challenges: srsRAN Implementation

One main challenge in implementing ZMQ with srsRAN is the ZMQ initialization, which has to be done only once. To avoid multiple initializations of ZMQ publishers, they have to be initialized in the header file. A naive way is binding these ZMQ sockets to TCP ports in the class constructor, but the problem is it may be called multiple times due to the creation of multiple objects from the same class. This could potentially lead to the same ZMQ socket being bound repeatedly to the same TCP port, resulting in errors. To prevent this, we created a global boolean variable in the header file that tracks whether the ZMQ socket has been bound. This ensures that each socket is bound to a TCP port only once, skipping the binding operation if it has already been performed.

The next challenge is to obtain channel across all subcarriers and antennas. In the pusch_processor_impl.cpp file of the srsRAN stack [13], we use the PDU object to retrieve the number of OFDM symbols, RX ports, and layers for the current session. This information is utilized to collect channel estimates for each symbol across all RX ports and layers. The get_symbol_estimates function is called iteratively to obtain the channel estimate arrays for each symbol. This function is invoked every time a PUSCH is processed. The estimates, interpolated from DMRS values, cover all PRBs (Physical Resource Blocks) and symbols within a slot. For example, a UE with an allocation of 106 PRBs at a 30 kHz subcarrier spacing has 1272 channel estimate values per RX antenna. These estimates from all antennas are serialized into a single vector and written to the ZMQ buffer in one operation.

The final channelge is to setup a ZMQ subscriber in the srsRAN stack to receive null-steering coefficients from the RIC. This subscriber socket is bound to a TCP socket in the channel_equalizer_zf_ impl.cpp file and is used in the equalize_1xn.h file to receive coefficients from the RIC and apply them to all resource elements (REs). Similar to the publisher, socket binding is performed in the constructor of channel_equalizer_zf_impl .cpp, with a global variable ensuring that multiple TCP port bindings are avoided. It is noted that equalize_1xn.h in the srsRAN implementation does not process all REs simultaneously but instead divides them to perform equalization across multiple function calls. Therefore, the null-steering values sent by the RIC are received, split, and applied to the incoming data according to the same RE index split.

4.2 Over-the-air Experiment Setup

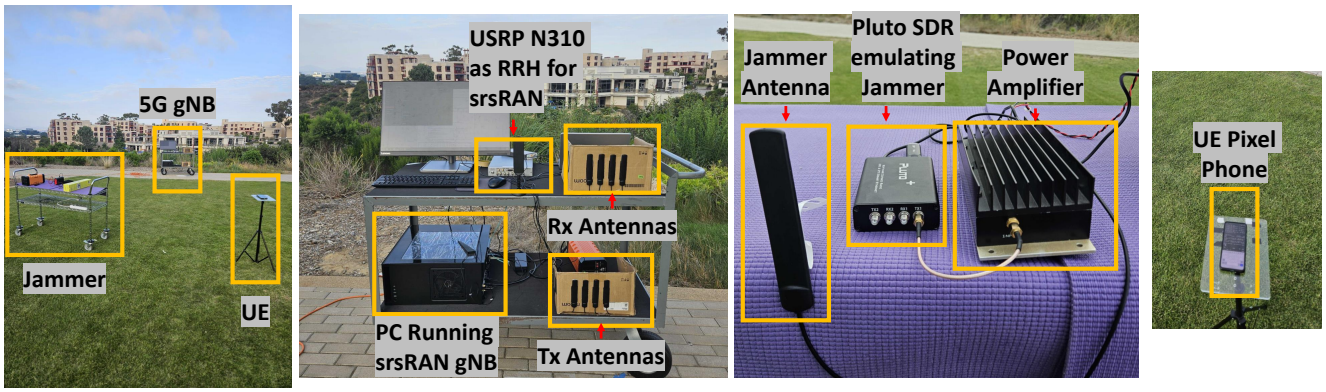
We created an over-the-air setup consisting of a Jammer, a UE, and a gNB. The gNB was a PC running srsRAN 5G with USRP N310 as the RF front-end for 5G uplink and downlink transmissions. We used the OnePlus 8 and Google Pixel phone as UE and a Pluto SDR as jammer that creates a wideband (40 MHz) OFDM signal in the UE's operating band, hence acting as an interference to the UE uplink transmissions. MIMO-RIC, our RAN controller was running on the same PC as the srsgnb. We ran our setup in various indoor and outdoor environments and it involved placing the UE at different angular locations wrt the gNB and collecting the channel estimate data using the MIMO-RIC framework. For emulating a scenario where the UE is experiencing interference from an external jammer, we placed the jammer (pluto SDR) close to the UE and collected the channel estimate data with and without the jammer.

5 Use Cases and Platform Evaluation

We first present three use cases for Jammer Monitoring, Jammer Nulling, and Localization and then show benchmark for platform evaluation.

5.1 Jammer Monitoring

We took multiple measurements with the UE and Jammer present at different locations. This was done to evaluate both



(a) Experiment Scenario (b) 5G gNB with 4 antenna MIMO Setup (c) Jammer PlutoSDR Setup (d) COTS UE Setup

Figure 3: Experiment Setup: the srsGNB PC running the RAN and the controller, PlutoSDR jammer, and Pixel phone UE.

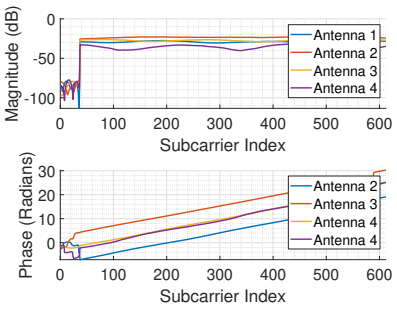


Figure 4: Wideband channel without jammer

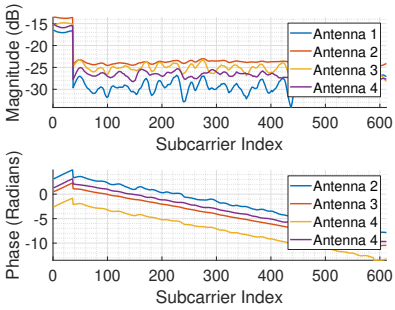


Figure 5: Wideband channel with jammer

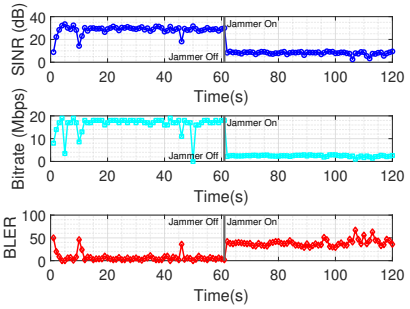


Figure 6: Jammer monitoring: SINR, BLER, and Bitrate

the accuracy and stability of the channel received. We captured multiple data points at each of these locations which included the SNR, throughput, and BLER measurements along with the channel estimates.

Figure 4 and 5 show the magnitude and phase plots of channel response across the subcarriers in the presence and absence of our wideband jammer. The effect of the jammer interference varied across the subcarriers in both magnitude and phase. This highlights, the need for having the channel estimate data per resource element (RE) since every RE would need a different nulling (or a precoding) matrix when the interference is wideband and has varying effects across subcarriers.

A rudimentary way of detecting a jammer would be to constantly monitor the SNR, BLER, and throughput metrics of the UE. In our implementation, we have placed hooks to monitor these values. Figure 6 show the SINR BLER and Bitrate plots with and without a jammer. As observed, there is a drop in the observed values of Bitrate and throughput as the jammer is introduced. As expected, the BLER value is higher in the presence of jammer.

5.2 Jammer Nulling

We use our MIMO-RIC platform to address the use case of jammer nulling. In contrast to previous work which only demonstrate LOS jammer nulling using average channel across all subcarriers [14], we demonstrate per-subcarrier jammer nulling using our unique wideband uplink channel data, which is desired for multi-path environments. Using the approach detailed in the design section, we calculated the per-subcarrier nulling matrix with the jammer at different angular locations. This nulling matrix is then used to equalize the received data. We compared the performance of using a per-subcarrier nulling matrix against that of using a constant nulling vector across all frequency subcarriers in Figure 7. Specifically, we evaluated the norm magnitude $\|\mathbf{h}_n * \mathbf{h}_t\|$, where $\mathbf{h}_n$ is the calculated nulling matrix with the dimension given by: *no. of subcarriers* $\times$ *no. of Rx ports* and $\mathbf{h}_t$ is the channel matrix with both the user and jammer operational, i.e., $\mathbf{h}_t = \mathbf{h}_u + \mathbf{h}_j$. The norm magnitude of this term is directly proportional to the SNR improvement. It is observed that the nulling matrix created using the per-subcarrier nulling coefficients provides us 2-5 dB

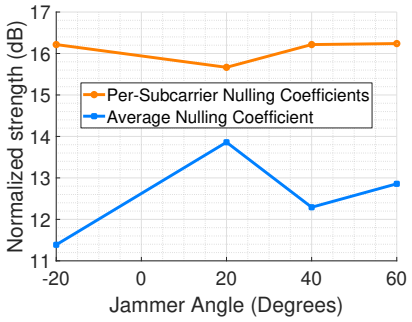
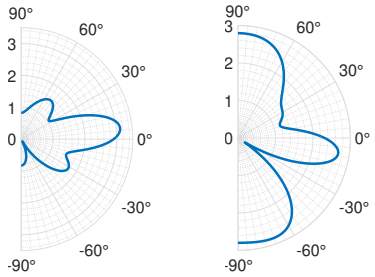


Figure 7: Normalized Signal Strength comparing wideband vs narrowband channel nulling performance.



(a) UE at 10^o **(b) UE at -20^o**

Figure 8: Beam pattern plotted using channel measurements for UE at different angles

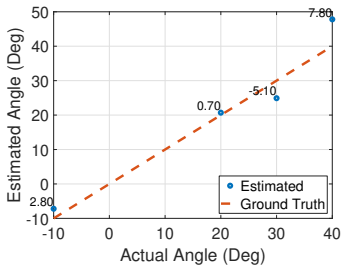


Figure 9: Plot with annotations showing error in the predicted angles at different UE angular locations

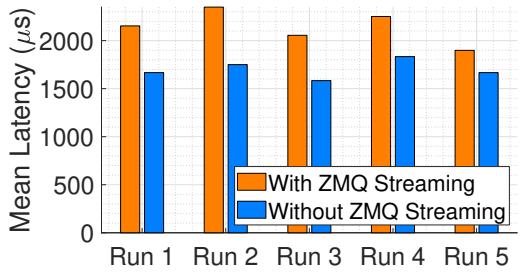


Figure 10: Plot Showing Latency Comparison between cases where MIMO-RIC ZMQ streaming is enabled and disabled

better performance when compared to using average nulling coefficient across all the frequency.

5.3 Localization

We demonstrate MIMO-RIC platform can be used for localization application. Using the gathered channel estimates with the UE positioned at various angles relative to the gNB, we plotted the FFT-based beam pattern in Figure 8 for the UE positioned at -10 degree and 20 degree with respect to the gNB. Figure 9 shows localization error at various other angles and demonstrates high accuracy of the detected angle even with simple FFT-based techniques. More advanced data-driven localization and tracking techniques such as DLoc [34] or mDTrack [35] can be included as an xApp in our MIMO-RIC platform in the future.

5.4 Platform Evaluation

To evaluate platform latency, we integrated a high-resolution timer into the srsRAN codebase to precisely measure the time intervals between consecutive calls of the PUSCH processing function. This approach allowed us to assess the impact of enabling and disabling ZMQ streaming to MIMO-RIC on latency. We conducted five test runs, each comprising 8,000 PUSCH processing function calls. The latency for each run was averaged across these calls, providing a comprehensive comparison of the system's performance with and without ZMQ streaming. Figure 10 shows that ZMQ introduces latency less than 500μs which is tolerable in real-time RAN stacks.

6 Conclusion

MIMO-RIC provides an open-source srsRAN-based controller platform for developing novel MIMO applications in real-time such as beamforming, beam-nulling, anti-jamming, and localization, demonstrating the significant potential for future O-RAN innovations.

7 Acknowledgement

This research was supported in part by the NSF grant 2030245 and the U.S. Department of Defense N66001-23-F-0484 P00001. We thank the anonymous reviewers and Shephard for providing insightful feedback and the members of WCSNG lab, UC San Diego, for group discussions.

References

[1] Iain Morris. Open RAN and the mission to crack massive MIMO. https://www.lightreading.com/open-ran/open-ran-and-the-mission-to-crack-massive-mimo, 2023.

[2] Xilinx. The Open RAN System Architecture and mMIMO. https://my.avnet.com/silica/resources/article/open-ran-system-architecture-and-mmimo/, 2021.

[3] Mavenir. Clarity on O-RAN Specification Updates for Massive MIMO Radios. https://www.mavenir.com/blog/clarity-on-o-ran-specification-updates-for-massive-mimo-radios/, 2023.

[4] Keysight. Keysight Enables Open RAN Massive MIMO Innovation. https://www.keysight.com/us/en/about/newsroom/news-releases/2024/0226-pr24-035-keysight-enables-open-ran-massive-mimo-innovation.html, 2023.

[5] Iain Morris. Verizon tech boss says open RAN still fails at massive MIMO. https://www.lightreading.com/open-ran/verizon-tech-boss-says-open-ran-still-fails-at-massive-mimo, 2023.

[6] Michele Polese, Leonardo Bonati, Salvatore D'Oro, Stefano Basagni, and Tommaso Melodia. Understanding o-ran: Architecture, interfaces, algorithms, security, and research challenges. *IEEE Communications Surveys & Tutorials*, 2023.

[7] Xenofon Foukas, Bozidar Radunovic, Matthew Balkwill, and Zhihua Lai. Taking 5G RAN analytics and control to a new level. In *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking*, pages 1–16, 2023.

[8] Chang Liu, Gopalasingham Aravinthan, Ahan Kak, and Nakjung Choi. Tinyric: Supercharging o-ran base stations with real-time control. In *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking*, pages 1–3, 2023.

[9] Robert Schmidt, Mikel Irazabal, and Navid Nikaein. Flexric: An sdk for next-generation sd-rans. In *Proceedings of the 17th International Conference on Emerging Networking EXperiments and Technologies*, CoNEXT '21, page 411–425, New York, NY, USA, 2021. ACM.

[10] Woo-Hyun Ko, Ushasi Ghosh, Ujwal Dinesha, Raini Wu, Srinivas Shakkottai, and Dinesh Bharadia. {EdgeRIC}: Empowering real-time intelligent optimization and control in {NextG} cellular networks. In *21st USENIX Symposium on Networked Systems Design and Implementation (NSDI 24)*, pages 1315–1330, 2024.

[11] Leonardo Lo Schiavo, Gines Garcia-Aviles, Andres Garcia-Saavedra, Marco Gramaglia, Marco Fiore, Albert Banchs, and Xavier Costa-Perez. Cloudric: Open radio access network (o-ran) virtualization with shared heterogeneous computing. In *Proceedings of the 30th Annual International Conference on Mobile Computing and Networking*, pages 558–572, 2024.

[12] VMware. Simplifying Multi-Vendor RAN Operations through Openness and Programmability with VMware RIC. https://www.vmware.com/docs/vmware-ran-intelligent-controller-datasheet, 2023.

[13] srsRAN. srsran: Open source 4g/5g software radio access network. https://github.com/srsran/srsRAN, 2023.

[14] Frederik Jonathan Zumegen, Ish Kumar Jain, and Dinesh Bharadia. Beamarmor demo: Anti-jamming system in cellular networks with srsran software radios. In *MILCOM 2023-2023 IEEE Military Communications Conference (MILCOM)*, pages 245–246. IEEE, 2023.

[15] Mavenir. Mavenir's RAN Intelligent Controller (RIC). https://www.mavenir.com/wp-content/uploads/2022/01/Mavenir_RIC_Solution_Brief_011722.pdf, 2023.

[16] Juniper Networks. Juniper RAN Intelligent Controller. https://www.juniper.net/content/dam/www/assets/datasheets/us/en/network-automation/juniper-ran-intelligent-controller-datasheet.pdf, 2023.

[17] OAI. Open Air Interface 5G Radio Access Network Project. https://openairinterface.org/oai-5g-ran-project/, 2023.

[18] Vikram R Anapana, Nathan H Stephenson, and Vijay K Shah. Milli-o-ran: A flexible, reconfigurable o-ran enabled mmwave network testbed. In *2024 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, pages 181–182. IEEE, 2024.

[19] Ish Kumar Jain, Raghav Subbaraman, and Dinesh Bharadia. Demo and dataset for mmwave multi-beam tracking using mmobile 28 ghz testbed. In *Proceedings of the 12th ACM Wireless of the Students, by the Students, and for the Students (S3) Workshop*, pages 8–8, 2021.

[20] Ish Kumar Jain, Raghav Subbaraman, and Dinesh Bharadia. A compact and real-time millimeter-wave experiment framework with true mobility capabilities. In *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking*, pages 1–3, 2023.

[21] Ish Kumar Jain et al. mmobile: Building a mmwave testbed to evaluate and address mobility effects. In *mmNets*, pages 1–6, 2020.

[22] Ish Kumar Jain, Raghav Subbaraman, and Dinesh Bharadia. Two beams are better than one: towards reliable and high throughput mmwave links. In *Proceedings of the 2021 ACM SIGCOMM*, pages 488–502, 2021.

[23] Ish Kumar Jain, Rohith Reddy Vennam, Raghav Subbaraman, and Dinesh Bharadia. mmflexible: Flexible directional frequency multiplexing for multi-user mmwave networks. In *IEEE INFOCOM 2023-IEEE Conference on Computer Communications*, pages 1–10. IEEE, 2023.

[24] Frederik Jonathan Zumegen, Ish Kumar Jain, and Dinesh Bharadia. Beamarmor: Seamless anti-jamming in 5g cellular networks with mimo null-steering. In *Proceedings of the 25th International Workshop on Mobile Computing Systems and Applications*, pages 121–126, 2024.

[25] Yongzhou Chen, Ruihao Yao, Haitham Hassanieh, and Radhika Mittal. {Channel-Aware} 5G {RAN} slicing with customizable schedulers. In *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23)*, pages 1767–1782, 2023.

[26] Corrado Puligheddu, Jonathan Ashdown, Carla Fabiana Chiasserini, and Francesco Restuccia. Sem-o-ran: Semantic and flexible o-ran slicing for nextg edge-assisted mobile systems. In *IEEE Infocom 2023-IEEE Conference on Computer Communications*, pages 1–10. IEEE, 2023.

[27] Salvatore D'Oro, Leonardo Bonati, Michele Polese, and Tommaso Melodia. Orchestran: Network automation through orchestrated intelligence in the open ran. In *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*, pages 270–279. IEEE, 2022.

[28] Claudio Fiandrino, Leonardo Bonati, Salvatore D'Oro, Michele Polese, Tommaso Melodia, and Joerg Widmer. Explora: Ai/ml explainability for the open ran. *Proceedings of the ACM on Networking*, 1(CoNEXT3):1–26, 2023.

[29] Archana Bura, Ushasi Ghosh, Dinesh Bharadia, and Srinivas Shakkottai. Windex: Realtime neural whittle indexing for scalable service guarantees in nextg cellular networks. *arXiv preprint arXiv:2406.01888*, 2024.

[30] Woo-Hyun Ko, Ujwal Dinesha, Ushasi Ghosh, Srinivas Shakkottai, Dinesh Bharadia, and Raini Wu. Edgeric: Empowering realtime intelligent optimization and control in nextg networks. *arXiv preprint arXiv:2304.11199*, 2023.

[31] Qing Yang, Xiaoxiao Li, Hongyi Yao, Ji Fang, Kun Tan, Wenjun Hu, Jiansong Zhang, and Yongguang Zhang. Bigstation: Enabling scalable real-time signal processingin large mu-mimo systems. *ACM SIGCOMM Computer Communication Review*, 43(4):399–410, 2013.

[32] Jian Ding, Rahman Doost-Mohammady, Anuj Kalia, and Lin Zhong. Agora: Real-time massive mimo baseband processing in software. In *Proceedings of the 16th international conference on emerging networking experiments and technologies*, pages 232–244, 2020.

[33] Junzhi Gong, Anuj Kalia, and Minlan Yu. Scalable distributed massive {MIMO} baseband processing. In *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23)*, pages 405–417, 2023.

[34] Roshan Ayyalasomayajula et al. Deep learning based wireless localization for indoor navigation. In *Mobicom*, pages 1–14, 2020.

[35] Yaxiong Xie, Jie Xiong, Mo Li, and Kyle Jamieson. md-track: Leveraging multi-dimensionality for passive indoor wi-fi tracking. In *The 25th Annual International Conference on Mobile Computing and Networking*, pages 1–16, 2019.