5G-NetBotSim: A Synchronized Co-Simulation Framework for Connected Robots over ROS and srsRAN-based 5G Network

Narges Golmohammadi, Bhaskara Venkata Raju Garuda, Ayan Joshi, and Sabur Baidya
Department of Computer Science and Engineering, University of Louisville, KY, USA
e-mail: n0golm01@louisville.edu, b0garu01@louisville.edu, ayan.joshi@louisville.edu, sabur.baidya@louisville.edu

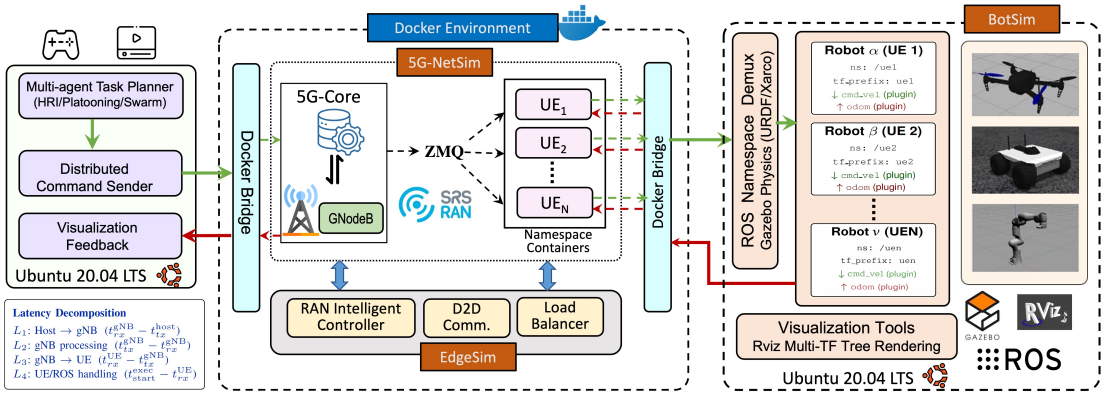


Fig. 1. 5G-NetBotSim architecture integrating an executable srsRAN-based 5G with isolated ROS/Gazebo robotic applications.

***Abstract*—Connected robotic applications increasingly rely on wireless communication for teleoperation, coordination, and distributed autonomy. However, robotic simulators often use ideal links or abstract delay models that do not capture cellular scheduling and protocol-processing effects. We present *5G-NetBotSim*, a real-time co-simulation framework integrating ROS/Gazebo workloads with an executable, full-stack srsRAN 5G network emulator. The framework provides isolated User Equipment (UE)–robot contexts and stage-level timing across the host, gNB, UE, ROS, and robot-execution stages. We evaluate human-to-robot teleoperation and robot-to-robot leader–follower coordination under controlled channel-quality conditions. Results show how CQI and MCS variations propagate to throughput, communication latency, and robot performance. In leader–follower coordination, higher channel quality reduces mean tracking error from 0.558 m to 0.460 m, a 17.6% reduction. These results demonstrate protocol-stack-in-the-loop co-simulation for joint network and robotic-application evaluation.**

***Index Terms*—5G; Robotics; co-simulation; ROS; srsRAN; networked robots**

I. Introduction

Connected robotic systems increasingly rely on wireless communication for teleoperation, multi-robot coordination, distributed perception, and collaborative autonomy. In such applications, delay, jitter, throughput variation, and packet loss can directly affect robot response, trajectory accuracy, coordination, and task completion. Robotic simulators typically model dynamics, sensing, planning, and control in detail, but often represent communication using ideal links or application-level delay models. These abstractions do not capture cellular scheduling, modulation and coding adaptation, protocol processing, per-device isolation, or RAN configuration. Conversely, cellular-network platforms mainly report network-level metrics without integrating the complete ROS-enabled command, execution, and feedback path. This makes it difficult to determine how network conditions propagate to robot-level performance.

To address this gap, we present **5G-NetBotSim**, a synchronized protocol-stack-in-the-loop co-simulation framework that integrates ROS/Gazebo robotic workloads with an executable, standards-aligned srsRAN 5G stack, shown in Fig. . The framework executes the complete 5G protocol-processing path, including physical (PHY), medium access control (MAC), radio link control (RLC), packet data convergence protocol (PDCP), and higher-layer components. thereby capturing cellular scheduling, protocol processing, and channel-dependent communication behavior within the robotic control loop. Rather than replacing the network with a fixed or stochastic delay model, 5G-NetBotSim preserves the full protocol-processing path and instruments each stage to relate network metrics directly to task-facing behavior.

5G-NetBotSim consists of three integrated components: *5G-NetSim*, which provides the virtual-radio 5G communication path using the executable srsRAN protocol stack; *BotSim*, which provides ROS/Gazebo-based robot simulation with separate ROS and network contexts for individual robots; and *EdgeSim*, which extends the RAN-side architecture with edge intelligence using EdgeRIC.

The main contributions are:

- We develop a synchronized protocol-stack-in-the-loop frame-

Framework	Robot–network co-simulation	Multi-robot support	5G/RAN aware	Executable 5G stack	Stage-level timing	Robot-task metrics	Per-robot isolation	RAN-edge integration	D2D MAC bypass
CORNET [1]	✓	–	–	–	–	✓	–	–	–
SynchroSim [2]	✓	✓	–	–	–	✓	–	–	–
OROS [3]	✓	✓	✓	–	–	✓	–	✓	–
FogROS2 family [4]–[6]	–	✓	–	–	–	✓	–	✓	–
NetROS-5G [7]	✓	–	✓	–	–	✓	–	✓	–
5G Network Digital Twin [8]	✓	–	✓	–	–	✓	–	–	–
EdgeRIC [9]	–	–	✓	✓	–	–	–	✓	–
5G-NetBotSim	✓	✓†	✓	✓	✓	✓	✓	✓	✓

†Multi-robot execution is experimentally demonstrated with two independently isolated robot–UE instances; Future scope involves more experiments with a swarm of a larger number of robots.

TABLE I: Capability comparison with representative robot-network, edge-robotics, and RAN-control platforms.

work coupling ROS/Gazebo with an executable srsRAN 5G stack.

- We provide per-robot UE, network, and ROS isolation with sequence-aligned timing across host, gNB, UE, ROS, and robot-execution stages.
- We integrate EdgeRIC through EdgeSim for RAN-side intelligent control and communication-aware robotic applications.
- We evaluate teleoperation and leader–follower coordination under controlled CQI regimes, linking network conditions to robot performance.

II. Related Work

Prior robot–network co-simulation frameworks such as CORNET and SynchroSim jointly evaluate robotic behavior and communication-network performance [1], [2]. However, these platforms primarily focus on general network modeling and simulation synchronization rather than executing a cellular protocol stack within the complete robotic command and execution path. Several studies investigate 5G-enabled robotics, network orchestration, and cloud/edge support. OROS jointly manages collaborative robot operation and wireless resources [3], while task-aware 5G resource-allocation approaches improve communication and control performance in latency-sensitive telerobotic applications [10]–[12]. NetROS-5G integrates ROS 2 with network slicing and edge/cloud resources [7], network digital-twin approaches model communication-aware 5G-connected robots [8], and FogROS2 supports cloud/fog deployment of robotic components [4]–[6]. EdgeRIC further enables near-real-time application-aware RAN control [9]. Communication-aware robotic studies show that network delay and middleware behavior can directly affect teleoperation and multi-robot performance [13], [14]. Motivated by these observations, 5G-NetBotSim integrates ROS/Gazebo workloads with an executable srsRAN 5G stack and EdgeRIC-based RAN intelligence, while providing isolated UE–robot contexts and stage-level timing across the network, middleware, and robot-execution stages.

III. System Design and Architecture

The 5G-NetBotSim integrates a 5G environment into ROS/Gazebo applications. Each robot is mapped to an independent UE, Linux namespace, ROS namespace, and control process. Commands are transferred through ZeroMQ to the Dockerized gNB, traverse the executable 5G downlink, and are forwarded from the selected UE to the corresponding ROS process. Sequence identifiers and monotonic timestamps align network and robot-execution events.

A. 5G-NetSim: Executable 5G Network Emulation

5G-NetSim uses the simulation mode of the open-source srsRAN stack. The gNB and UE processes run inside Docker and retain PHY, MAC, RLC, PDCP, and higher-layer processing through a virtual-radio interface, enabling repeatable network configuration. Each UE operates in an independent network context mapped to a robot, supporting per-device routing, configuration, logging, and measurements such as CQI, MCS, throughput, and packet latency. The experiments use virtual-radio rather than over-the-air RF transmission, focusing on controlled protocol-stack execution and its impact on robotic applications.

B. BotSim: Isolated Robotic Execution

BotSim provides the robotic application layer using ROS Noetic and Gazebo 11. Each UE endpoint connects to a dedicated ROS namespace and robot instance, forwarding received commands to the corresponding ROS control node for planning or execution.

The one-to-one UE-to-robot mapping provides routing and measurement isolation. ZeroMQ is used only at the host–Docker boundaries to keep ROS/Gazebo and srsRAN independently configurable and does not replace the evaluated gNB–UE path. This separation also allows robotic workloads and network configurations to be modified independently.

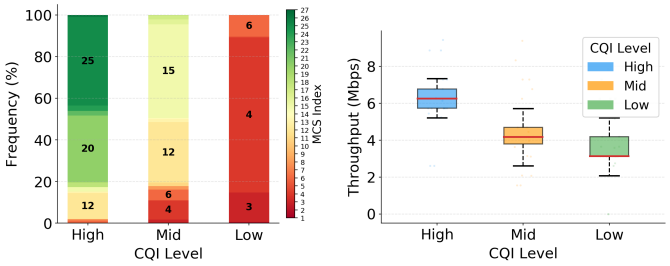
C. EdgeSim: RAN-Side Edge Integration

EdgeSim provides RAN-side support for edge-assisted communication and control using EdgeRIC, enabling near-real-time interaction with the RAN and application-aware network control without modifying the ROS/Gazebo layer.

D. End-to-End Integration and Timing Instrumentation

Each command k carries a sequence identifier and timestamps generated using `time.monotonic_ns()`. Because the Dockerized 5G and host-side ROS processes run on the same computer, they share a common monotonic timing reference. The stage-level delays are

$$\begin{aligned} L_1 &= t_{\mathrm{rx}}^{\mathrm{gNB}} - t_{\mathrm{tx}}^{\mathrm{app}}, \quad & L_2 &= t_{\mathrm{tx}}^{\mathrm{gNB}} - t_{\mathrm{rx}}^{\mathrm{gNB}}, \\ L_3 &= t_{\mathrm{rx}}^{\mathrm{UE}} - t_{\mathrm{tx}}^{\mathrm{gNB}}, \quad & L_4 &= t_{\mathrm{start}}^{\mathrm{exec}} - t_{\mathrm{rx}}^{\mathrm{UE}}, \\ L_5 &= t_{\mathrm{done}}^{\mathrm{exec}} - t_{\mathrm{start}}^{\mathrm{exec}}. \end{aligned} \tag{1}$$



(a) Downlink MCS (b) Throughput measurements

Fig. 1: Selected MCS for different CQI levels and corresponding throughput results.

Here, L_1–L_5 represent host-to-gNB transfer, gNB processing, gNB-to-UE communication, UE-to-ROS handoff, and robot execution, respectively. Thus,

$$L_{\text{delivery}} = L_1 + L_2 + L_3. \tag{2}$$

This instrumentation separates communication delay from application-side queueing, planning, and execution. The single-host setup also avoids cross-machine clock-synchronization uncertainty. All experiments run on Ubuntu 20.04 with one gNB, two UEs, 10 MHz bandwidth, 15 kHz subcarrier spacing, 1 ms TTI, and a MAX-CQI scheduler. Low-, medium-, and high-CQI regimes are evaluated with fixed robotic workloads and control parameters.

IV. Performance Evaluation

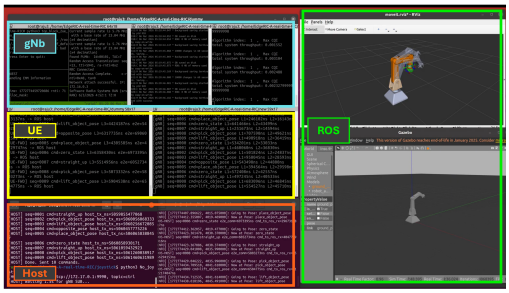
This section evaluates whether the configured CQI conditions propagate consistently through the executable 5G stack to produce measurable effects at the network and application levels, from MCS selection and throughput to communication latency and robot-task performance.

A. Network-Level Performance

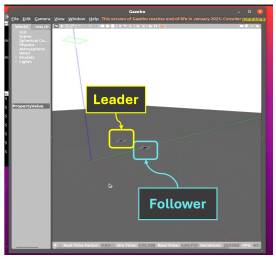
Fig. 1 shows the selected MCS and corresponding user-plane throughput under low-, medium-, and high-CQI conditions. Higher CQI results in higher MCS indices and greater achievable throughput, while lower CQI leads to more robust modulation and coding and reduced data rate. These results confirm that the configured CQI conditions affect the actual scheduling and transmission behavior of the executable 5G stack rather than remaining only simulation parameters. The stage-level latency measurements, shown in Fig. 3, further decompose command delivery across the host-to-gNB transfer, gNB processing, gNB-to-UE transmission, and robot-side handling. This decomposition helps distinguish network delay from application-side processing and execution effects, which is important because low communication latency alone does not necessarily imply low task-facing response time.

B. Application-Level Performance

We evaluate 5G-NetBotSim using two representative connected-robot workloads: human-to-robot teleoperation and robot-to-robot leader–follower coordination, shown in Fig. 2.



(a) Human-to-Robot Teleoperation



(b) Robot-to-Robot Leader–Follower Coordination

Fig. 2: Representative 5G-NetBotSim: (a) human-to-robot teleoperation and (b) robot-to-robot leader-follower coordination.

1) Human-to-Robot Teleoperation: For waypoint-based commands, Fig. 3, communication and application delays affect the task response time while the requested target remains unchanged. For velocity-based commands, newly received commands may interrupt the current trajectory and trigger replanning from the robot's current state. Thus, variations in command-delivery time can change the realized trajectory.

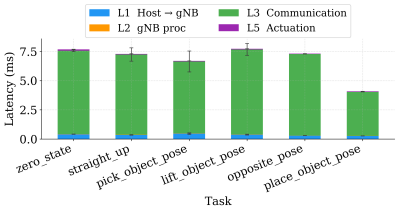
2) Robot-to-Robot Leader–Follower Coordination: The leader transmits its state to the follower at 10 Hz, and the follower uses the latest received state to maintain a desired 2 m offset. Under high CQI, (Fig. 4), the mean tracking error decreases from 0.558 m to 0.460 m, corresponding to a 17.6% reduction compared with low CQI. The improvement is most visible during turns and waypoint transitions, where delayed state updates increase the follower's spatial lag.

V. Discussion

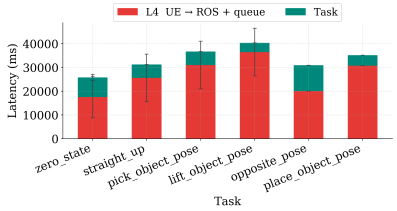
The experiments demonstrate the value of protocol-stack-in-the-loop simulation when protocol behavior and its impact on robotic applications are of interest. Unlike abstract delay/loss models, an executable stack exposes effects such as CQI-dependent MCS selection, scheduling, per-UE behavior, and stage-level processing. The current evaluation uses virtual-radio operation on a single host with two robot–UE pairs. Thus, the results characterize controlled protocol/application interactions rather than over-the-air fidelity or large-scale scalability. Physical-radio and larger multi-robot evaluations remain future work.

VI. Conclusion

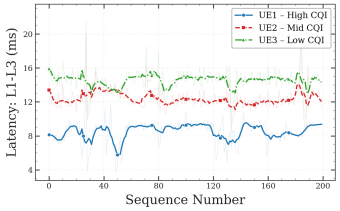
This paper presented 5G-NetBotSim, a protocol-stack-in-the-loop co-simulation framework integrating an executable srsRAN virtual-radio network with ROS/Gazebo robotic applications. Per-robot communication and ROS isolation together



(a) E2E latency

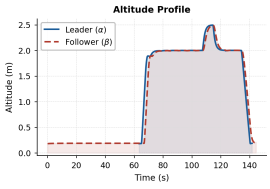


(b) Task latency

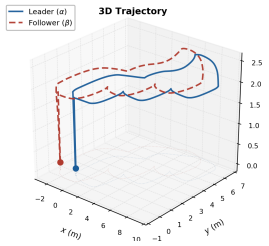


(c) Total latency

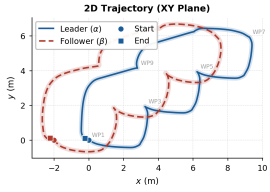
Fig. 3: Application- and task-level latency measurements.



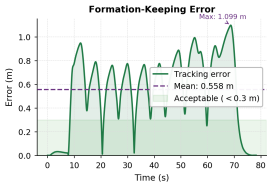
(a) Low CQI: Altitude



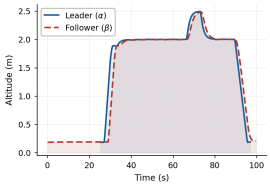
(b) Low CQI: 3D trajectory



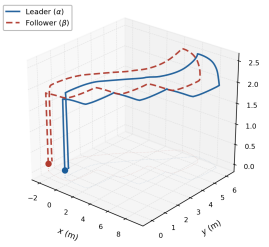
(c) Low CQI: XY trajectory



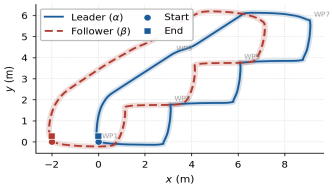
(d) Low CQI: Tracking error



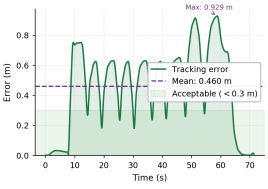
(e) High CQI: Altitude



(f) High CQI: 3D trajectory



(g) High CQI: XY trajectory



(h) High CQI: Tracking error

Fig. 4: Leader-follower trajectory and tracking performance under low- and high-CQI conditions. The first row shows the low-CQI case, and the second row shows the high-CQI case.

with stage-level timing allow network measurements to be related directly to task-facing robotic behavior. Teleoperation and leader–follower experiments demonstrate how controlled channel-quality conditions propagate through MCS selection, throughput, communication latency, and robot execution. In the leader–follower experiment, improved channel quality reduces mean tracking error from 0.558 m to 0.460 m, corresponding to a 17.6% improvement. Future work will investigate physical-radio validation, distributed execution, and larger multi-robot deployments.

References

[1] S. Acharya, A. Bharadwaj, Y. Simmhan, A. Gopalan, P. Parag, and H. Tyagi, "Cornet: A co-simulation middleware for robot networks," in *2020 International Conference on COMmunication Systems & NETworkS (COMSNETS)*. IEEE, 2020, pp. 245–251.

[2] E. Dey, J. Hossain, N. Roy, and C. Busart, "Synchrosim: An integrated co-simulation middleware for heterogeneous multi-robot system," in *2022 18th international conference on distributed computing in sensor systems (DCOSS)*. IEEE, 2022, pp. 334–341.

[3] A. Romero, C. Delgado, L. Zanzi, X. Li, and X. Costa-Pérez, "Oros: Online operation and orchestration of collaborative robots using 5g," *IEEE Transactions on Network and Service Management*, 2023.

[4] K. Chen, R. Hoque, K. Dharmarajan, E. LLontopl, S. Adebola, J. Ichnowski, J. Kubiatowicz, and K. Goldberg, "Fogros2-sgc: A ros2 cloud robotics platform for secure global connectivity," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023, pp. 1–8.

[5] K. Chen, K. Hari, T. Chung, M. Wang, N. Tian, C. Juette, J. Ichnowski, L. Ren, J. Kubiatowicz, I. Stoica *et al.*, "Fogros2-ft: Fault tolerant cloud robotics," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2024.

[6] J. Ichnowski, K. Chen, K. Dharmarajan, S. Adebola, M. Danielczuk, V. Mayoral-Vilches, N. Jha, H. Zhan, E. LLontop, D. Xu *et al.*, "Fogros2: An adaptive platform for cloud and fog robotics using ros 2," in *2023 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2023, pp. 5493–5500.

[7] A. Dalgkitsis and C. Verikoukis, "Netros-5g: Enhancing personalization through 5g network slicing and edge computing in human-robot interactions," *arXiv preprint arXiv:2312.06475*, 2023.

[8] L. Roda-Sanchez, L. Zanzi, X. Li, G. Garí, and X. Costa-Pérez, "Network digital twin for 5g-enabled mobile robots," in *2025 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2025, pp. 1–6.

[9] W.-H. Ko, U. Dinesha, U. Ghosh, S. Shakkottai, D. Bharadia, and R. Wu, "Edgeric: Empowering realtime intelligent optimization and control in nextg networks," *arXivorg*, no. 2304.11199., 2023.

[10] N. Golmohammadi, M. M. Rayguru, and S. Baidya, "A hierarchical optimization framework using deep reinforcement learning for task-driven bandwidth allocation in 5g teleoperation," *arXiv preprint arXiv:2505.15977*, 2025.

[11] ——, "Lyapunov-optimized 5g-sliced communications for telerobotic applications," in *IEEE INFOCOM 2024-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2024, pp. 1–6.

[12] N. Golmohammadi, M. Mohan Rayguru, and S. Baidya, "Drastic: A dynamic resource allocation framework over 6g network slicing in task-aware closed-loop tactile internet applications," *arXiv e-prints*, pp. arXiv–2603, 2026.

[13] P. Farajiparvar, H. Ying, and A. Pandya, "A brief survey of telerobotic time delay mitigation," *Frontiers in Robotics and AI*, 2020.

[14] D. Rakita, B. Mutlu, and M. Gleicher, "Effects of onset latency and robot speed delays on mimicry-control teleoperation," in *HRI'20: Proceedings of the 2020 ACM/IEEE International Conference on Human-Robot Interaction*, 2020.