Accepted for presentation in: IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), Tokyo, Japan, May 2026.

O-RACES: Proactive AI-driven Scheduling in Open RAN for 6G-Networked Humanoid Robots

Niklas A. Wagner, Christian Wietfeld
Chair of Communication Networks, TU Dortmund University, 44227 Dortmund, Germany
Email: {niklas.wagner, christian.wietfeld}@tu-dortmund.de

***Abstract*—Real-time communications is an essential feature of future 6G networks, enabling time-critical applications including cooperative robotic tasks in industry environments. Here, humanoid robots pose particular challenges due to their high complexity. When operating closed-loop via networked control, they exhibit stringent requirements on the wireless communications system. While single robots can be controlled locally, challenging collaborative tasks require high-fidelity networked control via a central entity. Reinforcement Learning (RL)-trained and edge-cloud-computed neural network policies enable agile networked robotics that do not rely on energy-intensive computing hardware on the robot. However, conventional reactive scheduling in 5G networks introduces significant uplink latency, limiting time-critical robotic control. To overcome this limitation, we propose the *Open RAN Real-time AI-Coordinated Efficient Scheduling (O-RACES)* proactive scheduling framework operating on open interfaces and designed to meet the stringent latency requirements of networked humanoid robots. We evaluate the scalability of *O-RACES* in a hybrid experimental environment using 16 virtual humanoid robots integrated within an end-to-end experimental Open RAN 6G research testbed. Results show that *O-RACES* reduces latency by up to 50% compared to reactive scheduling and resource usage by 40% versus static scheduling, paving the way for scalable collaborative 6G-networked robotics.**

***Index Terms*—6G, Edge Computing, Humanoid Robots, Networked Robotics, Open RAN, Proactive Scheduling**

I. Introduction

Humanoid robots and physics-focused Artificial Intelligence (AI) are increasingly recognized as a promising technology for industrial and intralogistics applications [1], offering human-like flexibility in dynamic environments such as warehouses, manufacturing floors, and logistics hubs. Humanoid robots seamlessly integrate into human-centric workflows, eliminating the need for specialized automation hardware.

Realizing safe and efficient Real-Time (RT) control of cooperative humanoid robots requires communication systems that provide ultra-low latency and high reliability at scale. Especially bipedal locomotion demands RT feedback loops between perception, decision-making, and actuation, with dynamics that tolerate only low millisecond-level delays. Offloading computationally intensive tasks to an AI Edge Cloud (AI-EC), as shown in Fig. 1, enables centralized multi-agent optimization while reducing onboard power consumption and enabling lighter robot designs. However, this architectural shift introduces stringent wireless connectivity requirements. Here, reactive uplink resource scheduling incurs delays and jitter that can destabilize millisecond-critical locomotion control.

While conventional proactive scheduling, e.g., by timer-based uplink prescheduling and grant-free uplink scheduling, reduces latency, it often suffers under high network load due to inefficient spectrum utilization or collisions. To overcome these limitations, proactive predictive scheduling allows radio resources to be allocated prior to scheduling requests, targeting low latency and maintaining spectral efficiency.

Our previous study [2] demonstrated AI-driven scheduling using a single two-wheeled inverted pendulum robot within a Software-Defined Radio (SDR)-based testbed. Building upon this foundation, we extend the system to large-scale edge-controlled humanoid robotics with the proposed *Open RAN Real-time AI-Coordinated Efficient Scheduling (O-RACES)* distributed App (dApp) framework. We now investigate up to 16 virtual humanoid robots operating in parallel, each performing Reinforcement Learning (RL)-trained locomotion tasks across uneven terrain with accurate RT physics. Each robot interacts with its controller in the AI-EC via an Open Radio Access Network (RAN) wireless experimental testbed incorporating *O-RACES* dApps and a large-scale indoor cell with high bandwidth using a Split 7.2 Open Radio Unit (O-RU), demonstrating scalable proactive resource management across multiple simultaneous critical control streams.

This paper provides the following three key contributions:

- A scalable AI-driven proactive uplink scheduling architecture for Open RAN to efficiently reduce uplink latency.

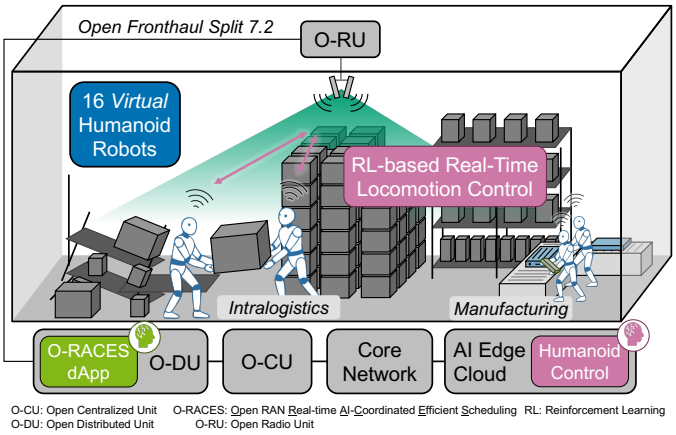

Fig. 1: Scenario within an intralogistics environment. *O-RACES* supports the control of humanoid robots for centralized whole-body locomotion control from the AI-EC via predictive uplink scheduling.

- RL-trained humanoid locomotion policies integrated in an AI-EC for millisecond-level RT feedback control.
- Experimental validation demonstrating that predictive resource allocation enables stable multi-robot bipedal locomotion while reducing spectrum consumption.

Thus, we aim to support the practical transition of next-generation 6G networking research into networked robotics for intelligent industrial and intralogistics robotic agents, operating safely and efficiently in a shared wireless network.

The remainder of this article is structured as follows: Section II provides previous works relevant to this paper. Next, Section III gives details about the *O-RACES* approach and its implementation as dApp within the Open RAN. The RL-based training procedure of the humanoid locomotion controller for uneven terrain is presented in Section IV. Subsequently, the experimental testbed and corresponding results are presented in Section V. Finally, Section VI concludes the paper.

II. Related Work

In this section, we review related work on mobile network scheduling, Open RAN architecture, RL-based robot control, and integration in networked control systems.

A. Latency-aware Scheduling and Open RAN Architecture

One of the focus points of current 6G research is RT communications for critical control applications, their co-design, and the coexistence of heterogeneous traffic. Several approaches propose latency-aware scheduling in RT and near-RT control loops to guarantee deterministic delay bounds [3]. Recent work explores AI-driven proactive scheduling mainly through simulation, including multi-variate channel and Buffer Status Report (BSR) prediction [4], Bi-directional Long Short-Term Memory (LSTM)-based uplink BSR prediction for vehicular scenarios [5], and Deep RL schedulers with separated inter-arrival time and grant size agents [6]. Additionally, [7] introduces grant-free access with k-repetition for robustness.

Within real-world studies, mostly supported by Open RAN, [8] proposes extensive End-to-End (E2E) latency optimization in an SDR-based testbed including static proactive scheduling. Data-driven scheduling using LSTM within near-RT eXtended Apps (xApps) enable co-designing communication and control [9]. Building on the xApp concept, the introduction of distributed dApps [10] enabled the deployment of AI-driven microservices, e.g., for sensing [10] and scheduling [2] directly on the O-DU, thereby supporting sub-millisecond-level control loops and inference close to the lower layers. Overall, the literature on proactive scheduling is mostly based on simulation studies, and lacks E2E analysis of real-world networked multi-robot control scenarios needed for collaborative robotics.

B. Reinforcement Learning in Humanoid Robotics Control

In parallel with networking advancements, RL-based training of robotics control has progressed rapidly toward autonomous locomotion and manipulation across diverse environments. Especially for humanoid robots, RL has proven effective in learning complex motion patterns, achieving omnidirectional walking strategies that adapt to uneven terrain or external perturbations [11]. The survey reviews control approaches from model-based to learning-based methods, highlighting foundation models for generalist agents and whole-body control. Isaac GR00T [12] exemplifies such foundation models with a unified policy for multiple robots through pre-training with large datasets. As these are comparably large in size, they require powerful computing hardware: on-robot inference currently achieves only 11 Hz on-robot (Jetson Thor) but 32 Hz on a Desktop GPU, making edge-computing architectures useful for future RT control using physical AI.

Deploying simulation-trained policies on physical humanoids remains challenging due to sim-to-real transfer, data efficiency, and perception robustness under real-world uncertainties [11]. Recent advancements enable training without simulator overfitting by learning in imagination [13]. Within this work, we focus on full-body locomotion control with proprioceptive state observations and Light Detection And Ranging (LiDAR) measurements, focusing on the most challenging part of robot control to evaluate RT networking aspects of edge-controlled humanoid robots on joint level.

III. Proactive O-RACES Scheduling Approach

This section presents the developed *O-RACES* concept, supporting scalable predictive proactive resource scheduling. We describe its architectural design, O-DU modifications, and the challenge of millisecond-scale traffic forecasting.

A. Approaches to Uplink Resource Allocation

The considered approaches for dynamic uplink scheduling are illustrated in Figure 2. The reactive baseline scheduling relies mainly on Scheduling Requests (SRs) when no constant traffic flow is present, causing variable latency. Here, the User Equipment (UE) requests uplink resources during SR Occasions (SROs) when it has data to send, and the base station schedules grants in response, leading to queuing delays and jitter. After the grant is sent, the UE can transmit data k2 slots later, where the minimum k2 value depends on the position in the Time Division Duplex (TDD) pattern and minimum decoding time limits of the UE. Within this work, the time-domain resource allocation variables are chosen as

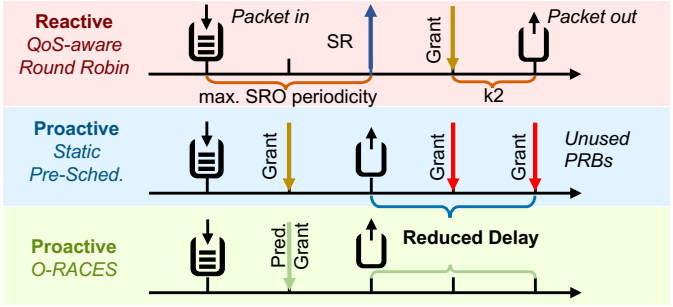

Fig. 2: Comparison of reactive, static proactive, and proposed *O-RACES* scheduling strategies for dyn. uplink resource allocation.

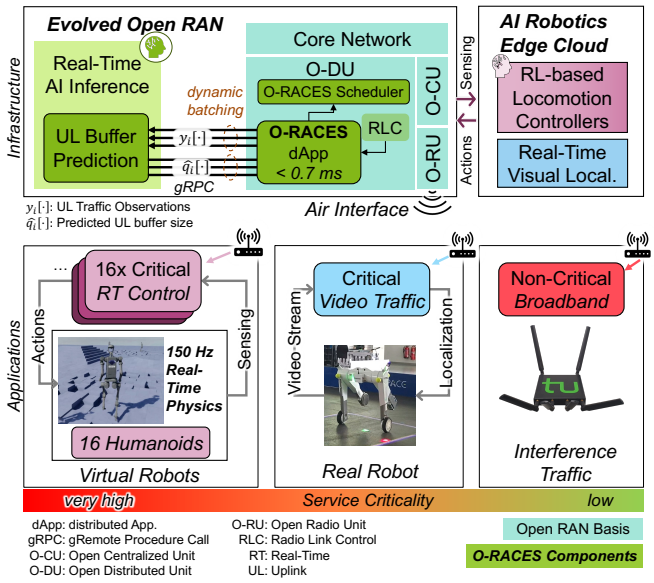

Fig. 3: Architecture and applications of the proposed *O-RACES* proactive scheduling framework supporting heterogeneous communication flows, including humanoid robots.

low as supported by RAN and UE to minimize latency. This baseline is a Quality of Service (QoS) Round Robin slicing scheduler, which the following approaches build upon. Static proactive scheduling additionally pre-allocates resources for the RT-critical robotics traffic on a fixed schedule for low latency. However, this static approach wastes spectrum when no data is sent. Hence, we propose the RT component of our *O-RACES* approach, which predicts resource needs on a millisecond timescale, achieving low latency and efficient spectrum use if accurately predicted.

B. Architecture of the O-RACES Proactive Scheduling

The architecture of *O-RACES* presented in Fig. 3 is built on open components and interfaces. The predictive dApp-enabled scheduling reduces uplink latency by anticipating needed resources based on monitoring Radio Link Control (RLC) uplink Service Data Unit (SDU) traffic as input for uplink buffer prediction, as discussed in more detail in [2]. The approach operates entirely on base station side without dedicated UE cooperation. Within this work, the *O-RACES* proactive scheduler component extends a QoS-aware Round-Robin Physical Uplink Shared Channel (PUSCH) scheduler supporting heterogeneous traffic with different criticality levels with UEs in slice groups via Network Slice Selection Assistance Information (NSSAI) information. For RT-critical UEs, new inference requests are issued every 20 ms by the *O-RACES* dApps to obtain resource predictions. For scalable inference, predictions are requested via gRPC Remote Procedure Calls (gRPC) from a containerized NVIDIA Triton instance hosted on the O-DU. Each slice group utilizes a dedicated model to support traffic with varying criticality requirements. UEs sharing the same model are dynamically batched to reduce inference overhead. When new uplink transmissions are anticipated, the scheduler pre-allocates Physical Resource Block (PRB) grants in subsequent uplink slots.

1) Traffic Model and Estimation: For a RT-critical UE i, let $q_i[k]$ denote the true uplink buffer size at time slot k, representing the actual Media Access Control (MAC) Protocol Data Unit (PDU) sizes per discrete time slot. The observed measurements on the base station side $y_i[k]$ are distorted by network effects such as packet fragmentation, queuing delays, and TDD alignment jitter, modeled by a causal discrete-time distortion operator $\mathcal{H}$. The observation can be described by:

$$y_i[k] = \mathcal{H}\{q_i[k]\} \tag{1}$$

To predict needed radio resources, the *O-RACES* traffic prediction component predicts $\hat{q}_i[k+1:k+L] = f(y_i[k-W:k])$ from $W = 100$ past observations, compensating for the network-induced distortions $\mathcal{H}$ and forecasts traffic prediction horizon $L = 30$ slots ahead. By learning the inverse mapping $\mathcal{H}^{-1}$ implicitly through training data containing random fragmentation and delays, the neural network estimator learns the relationship between noisy observable traffic characteristics and actual resource demands.

2) Real-Time Traffic Prediction using Machine Learning: The scheduling mechanism exploits common patterns of control system traffic. Accurate millisecond-level prediction of grants prevents scheduling requests and minimizes uplink buffer queuing times. The predictor employed is a lightweight single-hidden-layer DLinear model [14] trained in PyTorch [15] with a fast inference time of around 600 µs for 16 parallel UEs. To optimize the model, a custom weighted Mean Square Error (wMSE) loss function is used to penalize underestimation, preventing resource shortages:

$$\mathcal{L}_{\text{wMSE}} = \frac{1}{N}\sum_{i=1}^{N}\left(1+\lambda\,\mathbb{1}_{\{q_i[k]>0\}}\right)|\hat{q}_i[k]-q_i[k]|^2 \tag{2}$$

where λ is an optimized hyperparameter controlling the penalty strength, $\mathbb{1}$ is the indicator function, and N is the number of samples.

IV. Real-Time Virtual Humanoid Robots and RL-based Locomotion Control

In this section, the approach to humanoid robot locomotion control for closed-loop operation via the *O-RACES*-enabled Open RAN network is presented.

A. Virtual Humanoid Simulation and Locomotion Control

For the virtual humanoid robot platform, we utilize the Unitree H1 model, a bipedal robot with $n_j = 19$ actuated joints for dynamic locomotion tasks. The objective is to achieve stable and efficient locomotion across uneven terrain, which is critical for real-world applications in intralogistics, industrial environments, and rescue operations. The robot is simulated within Isaac Sim [16] for accurate RT physics. For the environment, we employ a rough terrain generator that creates randomized uneven surfaces, challenging the robot's balance and adaptability. The simulated sensors include an

Inertial Measurement Unit (IMU), joint encoders for position and velocity, and a down-facing LiDAR sensor for terrain mapping. The RT physics simulation loop, including all sensor updates, operates at 150 Hz to enable high-fidelity simulation. Control commands of the locomotion policy are also applied with 150 Hz, ensuring balance on the dynamic terrain.

B. Reinforcement Learning-based Locomotion Controller

The RL locomotion controller is implemented in RSL-RL [17]. In the robotics training environment Isaac Lab [18], we train 4096 robot instances in parallel to learn the policy using an actor-critic Proximal Policy Optimization (PPO) algorithm [19]. The actor network, mapping observations to actions, and the critic network both utilize feedforward neural architectures with an input dimension of n_{obs}. Each neural network comprises three hidden layers with 512, 256, and 128 neurons, respectively, using Exponential Linear Unit (ELU) activation function. The actor's output layer produces continuous actions in $\mathbb{R}^{n_j}$, while the critic outputs a scalar value estimate, both using linear activation.

1) Markov Decision Process Formulation: The bipedal locomotion task formulated as Markov Decision Process (MDP) is defined by the tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R})$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ the action space, $\mathcal{P}(s'|s,a) : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0,1]$ the transition probability, and $\mathcal{R}(s,a) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ the reward. The communication-relevant components are the following:

2) Observation State Space: The observation vector $\mathbf{o}_t \in \mathbb{R}^{n_{\text{obs}}}$ at timestep t is constructed from proprioceptive measurements as well as topographic height measurements. The observation has a total dimension of $n_{\text{obs}} = 256$.

$$\mathbf{o}_t = \begin{bmatrix} \mathbf{v}_{\text{base}} & \boldsymbol{\omega}_{\text{base}} & \mathbf{g}_{\text{proj}} & \mathbf{v}_{\text{cmd}} & \mathbf{q} & \dot{\mathbf{q}} & \mathbf{a}_{t-1} & \mathbf{h}_{\text{scan}} \end{bmatrix}^T \quad (3)$$

where $\mathbf{v}_{\text{base}}, \boldsymbol{\omega}_{\text{base}}, \mathbf{g}_{\text{proj}}, \mathbf{v}_{\text{cmd}} \in \mathbb{R}^3$ denote base linear velocity, angular velocity, projected gravity, and velocity command, respectively. Furthermore, $\mathbf{q}, \dot{\mathbf{q}}, \mathbf{a}_{t-1} \in \mathbb{R}^{n_j}$ represent the joint positions relative to their default positions, joint velocities, and previous actions for the n_j actuated joints. The down-facing LiDAR measurements $\mathbf{h}_{\text{scan}} \in \mathbb{R}^{187}$ represent discretized terrain elevations around the robot base.

3) Action Space: The action space represents normalized target joint positions $\mathbf{a}_t \in \mathcal{A} = [-1,1]^{n_j}$. These normalized actions are then mapped to joint torques via a Proportional-Derivative (PD) controller:

$$\boldsymbol{\tau}_t = K_p(0.5 \cdot \mathbf{a}_t - \mathbf{q}) - K_d\,\dot{\mathbf{q}} \quad (4)$$

where K_p and K_d are the stiffness and damping matrices. Torques are clamped to motor limits before joint actuation.

4) Reward Function: The total reward at each timestep t during training is defined as:

$$r_t = \sum_i w_i \cdot r_i(s_t, a_t) \quad (5)$$

where w_i are the reward weights and r_i are individual reward terms comprising velocity tracking, gait rewards (air time, sliding penalties), contact rewards, and regularization penalties (orientation, action rate, joint acceleration, joint limits, and joint deviations). Termination penalties are applied when episodes end prematurely, e.g., by falling.

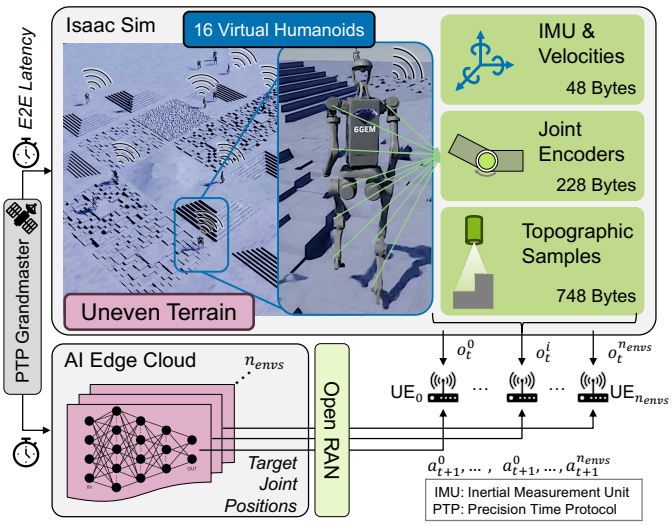

Fig. 4: RL-trained actor policy for humanoid locomotion control embedded in the Open-RAN-networked control architecture.

C. Evaluation of the control performance

To evaluate the locomotion control performance, we assess the Mean Absolute Error (MAE) of velocity tracking and the combined motor power consumption. The MAE is defined as:

$$\text{MAE} = \frac{1}{T}\sum_{t=1}^{T} \|\mathbf{v}_{\text{cmd},t} - \mathbf{v}_{\text{base},t}\|_1 \quad (6)$$

with $\mathbf{v}_{\text{cmd},t}$ for the commanded and $\mathbf{v}_{\text{base},t}$ for the actual velocity at time step t. The average motor power $\bar{P}_{\text{motor}}$ over time horizon T is computed from the joint torques $\tau_{t,j}$ and angular velocities $\omega_{t,j}$ for each joint j as:

$$\bar{P}_{\text{motor}} = \frac{1}{T}\sum_{t=1}^{T}\sum_{j=1}^{n_j} |\tau_{t,j} \cdot \omega_{t,j}| \quad (7)$$

D. Coupling of Virtual Humanoids and the Physical Testbed

To enable RT closed-loop control through the Open RAN network, each virtual humanoid robot instance is associated with a dedicated physical modem as shown in Fig. 4. This one-to-one mapping ensures identical communication and computational characteristics to physical robots, enabling realistic evaluation of E2E scalability. For each robot i, the observation vector $\mathbf{o}_t^i$ generated within Isaac Sim is serialized and transmitted via a unique UDP socket associated with modem i. The packets are then transmitted to the base station and forwarded to the AI-EC for inference of the actor policy. Afterward, the resulting action vector $\mathbf{a}_t^i$ is sent back from AI-EC worker to its respective receiving socket through UDP transmission.

Within each simulation step, the most recent action packet is applied as control input to the joints. Hence, the high-frequency physics simulation paired with real-world networks enables realistic evaluation of scalable distributed robot control over the experimental Open RAN network.

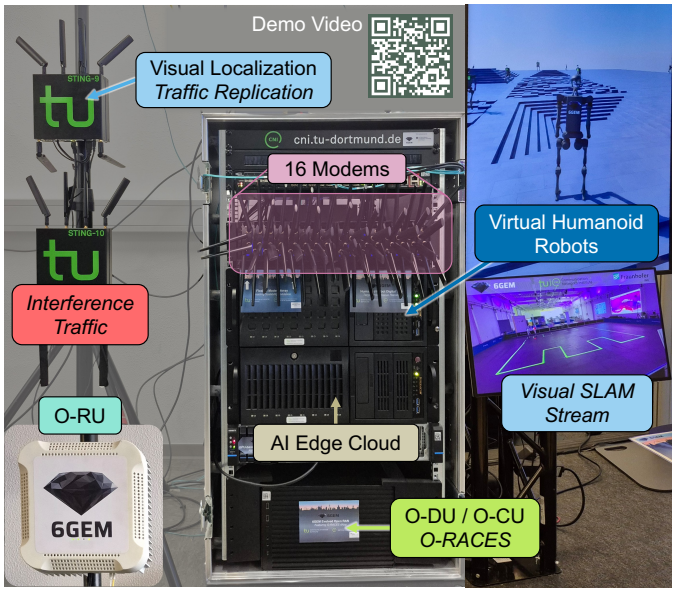

Fig. 5: Overview of the physical E2E *O-RACES* testbed components. Demo video available under: https://tiny.cc/O-RACES-Humanoid

TABLE I: Details on the Testbed Components incl. Open RAN

	Parameter	Description/Value
Open RAN	Radio Unit	Benetel RAN550 (O-RAN Split 7.2)
	Fronthaul Switch	Fibrolan Falcon RX
	CU / DU Platform	AMD Ryzen 9 9950X, 128 GB RAM,
	CU / DU Software	srsRAN v25.04 [20], OS: Ubuntu 24.04
	Operating Frequency	3.75 GHz, 100 MHz Bandwidth
	MIMO Configuration	4x4 DL (4 Layers) / 2x2 UL (1 Layer)
	Max. TX Power	24 dBm
	Slot Structure	30 kHz SCS, TDD pattern: DDSUU
UEs	Device Model	Quectel RM520N-GL
	Max. TX Power	23 dBm / 26 dBm (High Power UE)
	Capabilities	Release 16, incl. NR Standalone
Core	Software	Open5GS v2.7 [22]
	Deployed Functions	AMF, SMF, UPF, AUSF, NRF, UDM, UDR, PCF, NSSF, BSF, SCP
AI-EC	Software	Ubuntu 24.04, ONNX Runtime
	Hardware	AMD EPYC 7443, 256 GB RAM, NVIDIA RTX 4090
Robots	Model	Unitree H1 Humanoid Robot
	Simulator	Isaac Sim 4.5 [16]
	Hardware	Intel Xeon 6444Y, 256 GB RAM, NVIDIA RTX 6000 ADA

V. Experimental E2E Testbed and Evaluation

This section describes the experimental E2E testbed and provides scalability experiments for evaluating the proposed *O-RACES* proactive scheduling dApp for RT communications with wireless locomotion control of up to 16 humanoid robots.

A. Evaluation Scenario and Experimental Testbed

The mixed-critical experimental testbed comprises open-source and Commercial Off-The-Shelf (COTS) components detailed in Table I and Fig. 5. It consists of 18 physical UEs connected wirelessly to a single indoor Open RAN base station operating at the German industry campus network frequency of 3.75 GHz and Multiple Input Multiple Output (MIMO) support for robustness. The Open RAN stack, including *O-RACES*, is implemented within the open source RAN stack srsRAN Project [20]. To optimize the reactive scheduling performance, SRO periodicity is set to 5 ms with time domain parameters $k1 = k2 = 2$. On the application side, 16 modems enable highly time-critical closed-loop control of the RT virtual humanoid robots. Additionally, the Spatially distributed Traffic and Interference Generation (STING) system [21] provides one UE generating critical traffic of 20 Mbps, matching a cloud-processed 360° localization video stream of an intralogistics robot, and one UE generating non-critical background traffic to achieve full network congestion generated by iPerf3.

On the robot side, the humanoid platform is simulated within Isaac Sim (cf. Table I). The RL-trained locomotion actor policy (cf. Sec. IV) is deployed on the AI-EC server. AI-EC and the core's User Plane Function (UPF) are connected via 25 Gbps Ethernet to minimize latency. Each virtual robot connects through one physical UE, ensuring realistic communication patterns. The scalability of the end-to-end system is assessed by increasing the number of simultaneously controlled humanoid robots from 1 to 16, doubling at each step. Robots are initialized at fixed random positions with seeded random target velocities within linear $v_{\text{cmd},x} = [1.0, 3.0]\,\text{m/s}$ and angular $v_{\text{cmd},\omega} = [0.5, 0.8]\,\text{rad/s}$ velocity. This integration enables reproducible evaluation of the proposed scheduling strategies across virtual robots, physical network, and AI-EC.

All humanoid robots send their sensor observations at 150 Hz, i.e., every 6.67 ms to the AI-EC, where the locomotion policy is computed for each robot. The actions are then sent back in downlink to the robots. As all robots sample the sensors simultaneously, uplink traffic peaks occur every 6.67 ms, requiring efficient resource allocation to maintain low latency. The data rate per robot is 1.3 Mbps, resulting in a total net data rate of 20.8 Mbps for 16 robots, which is within the system bandwidth. As the humanoid robot UEs are prioritized with highest weight over video and competing background traffic, the O-RACES scheduler allocates resources to critical users first and can completely starve best effort users. Each experimental parameter set is run 16 times for 60 s.

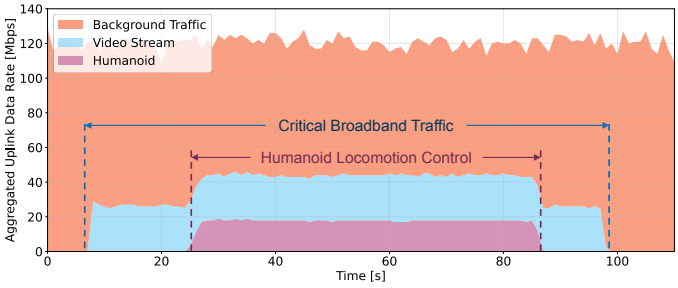

Fig. 6: Aggregated uplink data rate on MAC layer per traffic class. The observable prioritization in all schedulers, here *O-RACES*, weights critical robot control over video traffic, whereas background traffic is served with the remaining resources.

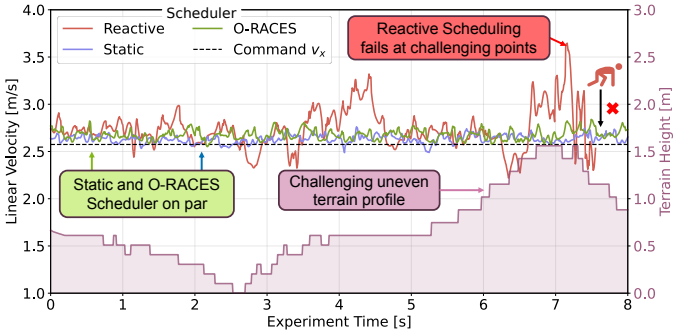

Fig. 7: Cutout trajectory of humanoid robot 2 running over uneven terrain using the different uplink resource schedulers.

B. Results of the Application-based Evaluation

For the evaluation, we measure application-level uplink one-way latency from observation packet generation within the virtual humanoid robot to the reception at the physical AI-EC, relative PRB allocation within the PUSCH scheduler and MAC layer data rate per traffic class, velocity reference MAE (6), and the experienced total motor power consumption $\bar{P}_{\text{motor}}$ (7) of the humanoids. The evaluation compares all scheduling strategies: Reactive scheduling as baseline, Static proactive allocation leading to over-provisioning, and the proposed predictive proactive *O-RACES* scheduler for 1 to 16 robots.

First, we show the prioritization mechanism in all of the employed scheduling approaches. In Fig. 6, the baseline slicing behavior is observable with best effort traffic being active for the whole duration. After around 7 s, the video traffic is activated reducing the background traffic data rate accordingly. The same behavior is seen when the humanoid robots are additionally operated between 25 s to 85 s, further reducing the available background traffic data rate. All schedulers implement this approach by giving highest priority to the RT-critical humanoid robot traffic. The critical video traffic is scheduled with medium priority to support a guaranteed bitrate as far as possible. Remaining resources are available for best effort background traffic by STING.

To visualize the locomotion task, Figure 7 shows a cutout of one challenging trajectory and topographic profile of humanoid robot 2 at the same randomly initialized position on the uneven terrain for all uplink scheduling strategies. In this case, the seeded random target linear velocity is 2.6 m/s forward. While both proactive schedulers, Static and *O-RACES*, enable stable locomotion, the robot controlled via the reactive baseline suffers from significant disturbances and falls after 7 s due to high and variable deadtime in the control loop. As seen in [9], there is a strong correlation between latency and control performance, also when using delay compensation.

During the following statistical evaluation, all traffic classes are active for the whole evaluated time intervals. Figure 8 presents PRB usage, application-layer latency, robot error rate, and its experienced MAE in tracking and motor power consumption for all scheduling strategies. Reactive scheduling maintains relatively constant median latency of around 15 ms but exhibits high tail latencies of a 99% percentile of 52 ms. As only needed resources are requested, the PRB usage is low with 2.3% to 30.7%. Static proactive scheduling achieves the lowest median latency of 6 ms through over-provisioning, but consumes up to 100% of available PRBs at 16 robots, starving video and background traffic. The proposed *O-RACES* achieves a balanced tradeoff with median latencies of 6.3 ms

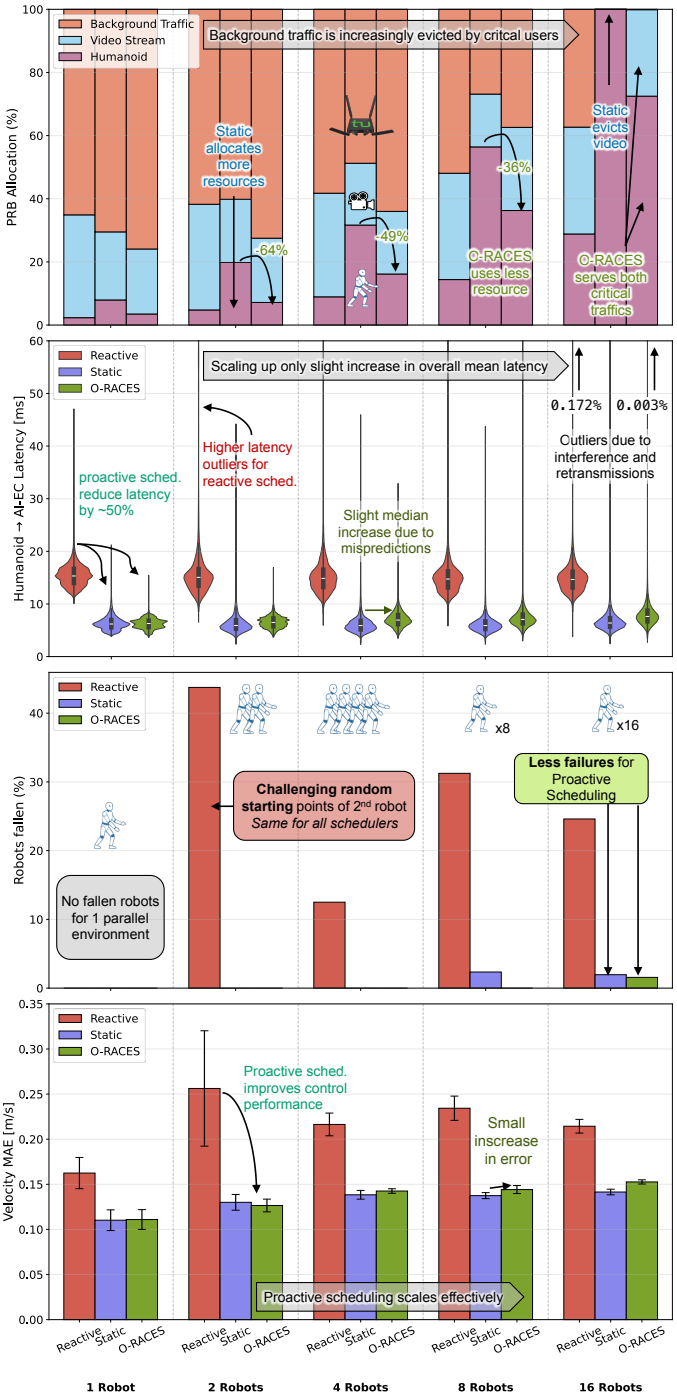

Fig. 8: Measured PRB usage, humanoid to AI-EC application latency, and the MAE of velocity tracking for different scheduling strategies. *O-RACES* offers best latency, resource, and control quality tradeoff. The control performance scales effectively with proactive scheduling.

TABLE II: Average total robot motor power $\bar{P}_{\text{motor}}$ for all schedulers and robot counts. Lower power indicates higher control efficiency.

Scheduler	1 Robot	2 Robots	4 Robots	8 Robots	16 Robots
Reactive	204 W	296 W	255 W	266 W	246 W
Static	178 W	205 W	216 W	208 W	202 W
O-RACES	178 W	201 W	217 W	211 W	208 W

to 7.6 ms, controlled tail latencies of 99% below 18 ms for 16 robots, and moderate PRB usage of 3.5% to 72.5%. Notably, *O-RACES* exhibits lower tail latency up to 4 robots than all other schedulers. However, the interference between UEs is a challenge since all robots always send their observations simultaneously, leading to higher tail latency for all schedulers at 16 robots. Overall, *O-RACES* maintains low latency while still supporting the video traffic even at maximum robot count, demonstrating superior resource efficiency.

Regarding control performance, the robots that fell during a 60 s run and the velocity MAE are considered. Reactive scheduling exhibits high failure rates of up to 43.8% for 2 robots and 31.2% for 8 robots, showing reduced performance, especially in challenging environments (cf. Fig. 7). In contrast, both static and *O-RACES* schedulers maintain low failure rates across all configurations, with *O-RACES* achieving the least failures of all schedulers with 0% up to 8 robots. The 1.6% failure rate at 16 robots is still below that of the static scheduling with 2%. The slight increase in latency compared to static allocation is reflected in a marginally higher MAE of 0.02 and motor power of 0 W to 6 W as seen in Table II. *O-RACES* outperforms reactive scheduling in motor power consumption in all scales. The results show that proactive scheduling effectively scales. The 80 GB measurement dataset of robot and RAN data is available [23].

VI. Conclusion and Outlook

This work proposed *O-RACES* for RT closed-loop control of networked humanoid robots within an experimental Open RAN 6G research testbed. Results showed that conventional reactive uplink scheduling introduces prohibitive latency for millisecond-critical bipedal locomotion control. The proposed proactive AI-driven scheduling *O-RACES* using a scalable dApp implementation enhances networked RL-trained locomotion via an AI-EC. The approach reduces uplink median latency by up to 50% while reducing resource consumption compared to static proactive scheduling by up to 40%. We challenged *O-RACES* with 16 virtual humanoid robots with dedicated physical modems per robot. Results demonstrate that proactive scheduling sustains stable control across multiple robots simultaneously, achieving superior latency and spectral efficiency tradeoff compared to reactive and static scheduling.

Future work includes integrating RL-based schedulers for improved traffic anticipation and increased resource efficiency as well as predictive channel-aware scheduling to reduce latency due to high interference. Additionally, we aim to introduce model-based RL for robot controllers for enhanced latency robustness toward real industrial environments.

Acknowledgment

This work has been funded by the Federal Ministry of Research, Technology and Space (BMFTR) via the *6GEM* research hub and the *6GEM+* transfer hub under funding references 16KISK038 and 16KIS2412. The authors thank Robin Wiebusch for the implementation support and the Fraunhofer IML for providing intralogistics domain requirements.

References

[1] "Physical AI: Powering the new age of industrial operations," World Economic Forum, White Paper, Sep. 2025.

[2] N. A. Wagner, J. Eßer, I. F. Priyanta, F. Kurtz, M. Roidl, and C. Wietfeld, "Real-time predictive scheduling for networked robot control using digital twins and OpenRAN," in *IEEE Globecom Workshops (GC Wkshps)*, Dec. 2024.

[3] O. Adamuz-Hinojosa, L. Zanzi, V. Sciancalepore, A. Garcia-Saavedra, and X. Costa-Pérez, "ORANUS: Latency-tailored orchestration via stochastic network calculus in 6G O-RAN," in *IEEE International Conference on Computer Communications (INFOCOM)*, May 2024.

[4] S. Lai, J. Li, D. Zhang, and M. Zhang, "LL-PGS: A lightweight and low-latency proactive grant scheduling algorithm for industrial IoT," *IEEE Wireless Communications Letters*, vol. 14, no. 2, pp. 295–299, 2025.

[5] V. Kumar Gautam, V. R. Chintapalli, B. R. Tamma, and C. S. Ram Murthy, "Enhancing uplink scheduling in 5G enabled vehicular networks: A cross-layer approach with predictive buffer status reporting," in *IEEE Veh. Technol. Conf. (VTC-Spring)*, Jun. 2024.

[6] K. Boutiba, M. Bagaa, and A. Ksentini, "On using deep reinforcement learning to reduce uplink latency for uRLLC services," in *IEEE Global Communications Conference*, Dec. 2022, pp. 407–412.

[7] Z. Zhao, Q. Du, and G. K. Karagiannidis, "Improved grant-free access for urllc via multi-tier-driven computing: Network-load learning, prediction, and resource allocation," *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 3, pp. 607–622, 2023.

[8] A. Gong, A. Maghsoudnia *et al.*, "Towards URLLC with open-source 5G software," in *1st Workshop on Open Research Infrastructures and Toolkits for 6G*. ACM, Sep. 2025.

[9] D. Overbeck, N. A. Wagner, R. Wiebusch, J. Püttschneider, T. Faulwasser, and C. Wietfeld, "Data-driven proactive uplink slicing enabling real-time control within an Open RAN testbed," in *IEEE Int. Conf. on Comp. Commun. Workshops (INFOCOM WKSHPS)*, May 2024.

[10] A. Lacava, L. Bonati *et al.*, "dApps: Enabling real-time AI-based Open RAN control," *Computer Networks*, vol. 269, 2025.

[11] Z. Gu, J. Li *et al.*, "Humanoid locomotion and manipulation: Current progress and challenges in control, planning, and learning," *IEEE/ASME Transactions on Mechatronics*, 2025.

[12] NVIDIA, J. Bjorck *et al.*, "GR00T N1: An open foundation model for generalist humanoid robots," Mar. 2025, arXiv:2503.14734.

[13] C. Li, A. Krause, and M. Hutter, "Robotic world model: A neural network simulator for robust policy optimization in robotics," in *Embodied World Models for Decision Making Workshop, NeurIPS*, Dec. 2025.

[14] A. Zeng, M. Chen, L. Zhang, and Q. Xu, "Are transformers effective for time series forecasting?" in *Proceedings of the AAAI Conference on Artificial Intelligence*, Feb. 2023.

[15] A. Paszke, S. Gross *et al.*, "PyTorch: An imperative style, high-performance deep learning library," in *Proc. of the Int. Conf. on Neural Information Processing Systems 32*, 2019, pp. 8026–8037.

[16] NVIDIA. Isaac Sim v4.5. [Online]. Available: https://github.com/isaac-sim/IsaacSim

[17] C. Schwarke, M. Mittal, N. Rudin, D. Hoeller, and M. Hutter, "RSL-RL: A learning library for robotics research," 2025, arXiv:2509.10771.

[18] M. Mittal, P. Roth *et al.*, "Isaac Lab - a GPU-accelerated simulation framework for multi-modal robot learning," 2025, arXiv:2511.04831.

[19] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, arXiv:1707.06347.

[20] Software Radio Systems (SRS), "srsRAN Project v25.04," 2025. [Online]. Available: https://docs.srsran.com/projects/project/en/latest/

[21] C. Arendt, S. Böcker, C. Bektas, and C. Wietfeld, "Better safe than sorry: Distributed testbed for performance evaluation of private networks," in *IEEE Future Networks World Forum (FNWF)*, 2022.

[22] S. Lee *et al.* (2025) Open5GS. Open Source implementation for 5G Core v2.7.5. [Online]. Available: https://open5gs.org/open5gs/docs/

[23] N. A. Wagner and C. Wietfeld, "O-RACES measurement dataset," 2026, doi:10.17877/TUDODATA-2026-MLFQ463Q.