Accepted for presentation in: IEEE Globecom Workshops (GC Wkshps): Workshop on
Digital Twins over NextG Wireless Networks, Cape Town, South Africa, December 2024.

Real-Time Predictive Scheduling for Networked Robot Control Using Digital Twins and OpenRAN

Niklas A. Wagner[1], Julian Eßer[2], Irfan Fachrudin Priyanta[3], Fabian Kurtz[1], Moritz Roidl[2,3], Christian Wietfeld[1]
[1]Communication Networks Institute (CNI), TU Dortmund University, 44227 Dortmund, Germany
[2]Fraunhofer Institute for Material Flow and Logistics, 44227 Dortmund, Germany
[3]Chair of Material Handling and Warehousing, TU Dortmund University, 44227 Dortmund, Germany
Email: {niklas.wagner, irfanfachrudin.priyanta, fabian.kurtz, moritz.roidl, christian.wietfeld}@tu-dortmund.de, julian.esser@iml.fraunhofer.de

***Abstract*—Safe and efficient real-time robotics control is highly delay-sensitive. Enabling such critical applications via mobile communication networks, therefore, hinges on reliably provisioning radio resources at low latency. Here, employing the Open Radio Access Network (O-RAN) concept, networks can be built with adaptive intelligent features, such as Artificial Intelligence (AI)-based scheduling policies for optimized resource management. By harnessing the innovative concept of distributed Applications (dApps) deployed inside the Open RAN Distributed Unit (O-DU), predictive resource allocation can reliably provide low latencies for robot control at increased spectral efficiency. This work demonstrates the Key Performance Indicators (KPIs) achieved with a proposed real-time proactive scheduling dApp employing AI methods. Results are derived from a real-world testbed that integrates predictive communication with a digital twin of the two-wheeled inverted pendulum robot evoBOT, designed for intralogistics. The closed-loop locomotion control, also providing upright stability control, is performed on the mobile edge via an evolved O-RAN system hosting the proposed dApp. Relative to optimized reactive network slicing, our approach yields a 34% mean reduction for uplink delays. Moreover, radio resource usage is reduced by up to 47% compared to highly optimized reactive scheduling, exhibiting similar control performance.**

I. Introduction

Robotics depends on control algorithms that are highly sensitive to delays. However, moving functions such as energy-intensive computer vision or locomotion control from individual entities to a Mobile Edge Cloud (MEC) may be desirable to facilitate, e.g., swarm optimization while increasing the robots' battery runtime. In turn, this requires reliable, high-performance wireless communication between robots and the MEC. Yet, the scheduling of cellular radio resources incurs delays, which may be unsuitable for such use cases. Here, the concept of distributed Applications (dApps) [1] has emerged, leveraging Open Radio Access Networks (O-RANs) to deploy real-time control loops on Distributed Units (DUs).

This work builds on these concepts by transferring Reinforcement Learning (RL)-based stability and locomotion control for the Digital Twin (DT) of the two-wheeled inverted pendulum logistics robot evoBOT [2] to the MEC. This enables reduced computational load on the robot, potentially resulting in lower battery consumption and increased runtime. Furthermore, the realized interfaces enable more efficient centralized fleet control, allowing, e.g., robust cooperative perception. Using the DT, the O-RAN can be optimized before deployment on the real robot. Here, we can reduce network latencies by utilizing Machine Learning (ML)-based predictive scheduling mechanisms shown to be effective for control systems traffic in previous work [3]. In this work, we enable real-time capability, proposing a novel dApp on an open-source evolved O-RAN network. The application scenario, cf. Fig. 1, comprises two slices for safety monitoring and remote robot control respectively. It is based on a real-world testbed and coupled with an interactive DT of the evoBOT, enabling user steering of the robot to experience the impact of the studied scheduling strategies. Performance is evaluated in terms of latency and spectral efficiency, while the impact on the robot is quantified via control behavior and the resulting impact on energy consumption. Moreover, the system's performance is assessed under challenging emulated channel conditions.

The work is structured as follows: Sec. II discusses the state of the art pertaining to this paper. Next, Sec. III illustrates the ML-based approach to proactive radio resource scheduling using dApps. Robot locomotion control with the corresponding DT used for training and testing is presented in Sec. IV. Evaluation setup and results are given in Sec. V. Lastly, Sec. VI offers a conclusion and an outlook on future research.

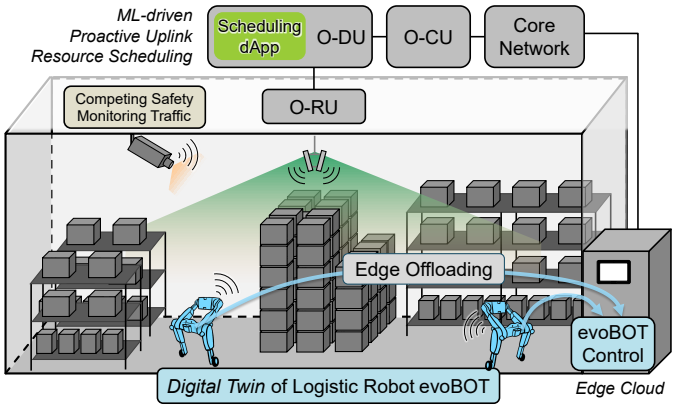

Fig. 1: Overview of the considered intralogistics scenario. The proposed data-driven proactive scheduling O-RAN dApp enables the deployment of real-time control for the robot DT at the mobile edge.

II. Related Work

Below, related works in the areas of scheduling, network architecture, ML-based robot control, and DTs are surveyed.

A. Latency-aware Scheduling and Open Network Architecture

Several works study resource scheduling strategies for Ultra-Reliable Low Latency Communications (URLLC) services in combination with broadband applications. The authors of [4] propose a nested-control-loop scheduling architecture using combined Real-Time (RT) and near-RT loops to guarantee upper bound latencies via stochastic network calculus. While the work supports multiple URLLC services, it relies on buffer reporting for resource allocation and provides results for downlink only, hence, does not exploit anticipating uplink scheduling. Grant-free architectures [5] have been studied to minimize latency; however, approaches have to face inefficient collisions under high network load. Previous work [3] shows high control performance using proactively guaranteed resources for an inverted rotary pendulum co-designing control and communication. There, a near-RT eXtended Application (xApp) [6] performs proactive scheduling using Long Short-Term Memory (LSTM) models to anticipate resource demands of User Equipments (UEs) in batches. Recently, the concept of dApps was introduced by [1] to support critical real-time control loops, enabling the direct deployment on the O-DU. Successful proactive scheduling requires accurate time series prediction for precise data transmission anticipation. The use of ML methods for critical functions in the mobile network requiring trustworthiness, like scheduling, is supported by explainable AI methods in [7]. Current advances in time series predictions [8]–[10] enable effective methods that outperform widely used LSTM methods, as further discussed in Sec. III-B.

B. RL-based Robot Control and Digital Twin Modeling

RL has emerged as a powerful technique for learning dynamic locomotion skills across various robotic platforms, including humanoids, quadrupeds, and wheeled robots. In humanoid robotics, RL has been successfully applied to achieve omnidirectional locomotion and to optimize foot placement strategies [11]. For quadruped robots, RL has enabled running at high speeds and traversing a variety of different natural terrains like grass, ice, or gravel [12]. For wheeled robots, [2] has shown the ability of RL to train agile locomotion tasks. A common scheme to simultaneously accelerate RL and improve performance for real-world robotic settings is to integrate additional knowledge into the training process [13]. In particular, to transfer trained policies from simulation to real robots, the robotics community relies on a combination of accurate simulation models (DTs) and domain randomization techniques. First, the simulation model itself needs to accurately represent the dynamics of the actual robot, typically involving real-world data collection and system identification [2], [14]. Subsequently, extensive randomization within the simulation environment is often used to increase the overall robustness of the trained control policies for real-world settings [15].

III. Proactive Resource Scheduling for Latency-Critical Applications in the Open RAN

This section illustrates the developed end-to-end O-RAN framework for proactive resource scheduling, its architecture, adaptations on the O-DU and challenges for millisecond-level traffic prediction with imprecise information.

A. Proactive Resource Management using ML-based dApps

The developed real-world communications testbed architecture shown in Fig. 2 is built entirely on open components. For the end-to-end O-RAN network, we employ the RAN stack *srsRAN Project* [16] offering open Centralized Unit (CU) and DU implementations. It enables integrating novel concepts as our scheduling dApp. Open5GS [17] is used as core network, providing slicing functionality. To enable reduced uplink latency, the predictive dApp-enabled resource scheduling anticipates the UEs' transmissions. Therefore, the Radio Link Control (RLC) layer's uplink Service Data Unit (SDU) sizes are continuously monitored and serve as input for the proposed scheduling dApp. Hence, the approach is fully provided on O-RAN side without cooperation of the UE.

A central connector extends the O-DU Physical Uplink Shared Channel (PUSCH) scheduler to integrate the proposed scheduling dApp, providing predictive scheduling on top of the reactive mechanisms. Every 10 ms, new inference requests are made of the scheduling dApps to provide new resource predictions. The models for each service type are served via Representational State Transfer (REST) protocol by a local containerized NVIDIA Triton instance hosted on the O-DU.

B. Real-Time Traffic Prediction using Machine Learning

Predictive scheduling requires traffic not to be entirely random. Hence, the scheduling mechanism exploits common patterns of control system traffic. Anticipating exact timing and packet sizes is the main challenge, as arrival process information at the base station is distorted by queuing times as well as packet loss and fragmentation. Additionally, the use of Time Division Duplex (TDD) systems poses the challenge of jitter caused by dead times in case the current transmission direction in the TDD pattern is not aligned with the data transmissions. By accurate millisecond-level prediction of grants, we prevent scheduling requests and minimize uplink buffer queuing times induced by the scheduling scheme.

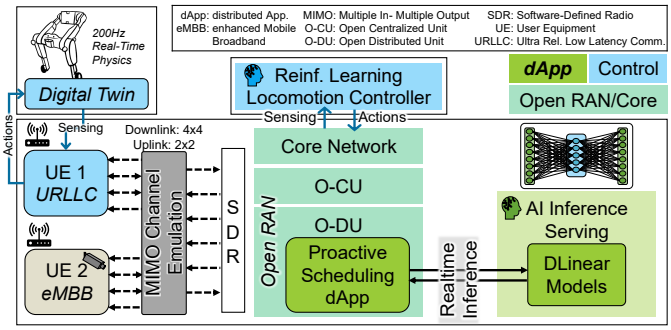

Fig. 2: Depiction of the O-RAN framework and communication streams for data-driven proactive scheduling by the proposed dApp.

Fig. 3: Experimental testbed setup, including the O-RAN-based network (right), a channel emulator (middle), and safety monitoring as well as latency-critical evoBOT robot control traffic within the interactive Digital Twin (left). (Demo Video: https://tiny.cc/DTControlDemo)

Training and hyper-parameter optimization are performed within the PyTorch [18] framework. To determine a suitable fit, a diverse range of models, including traditional (e.g. LSTM) and novel methods such as DLinear [9], PatchTST [8] and TimesNet [10] are evaluated. While TimesNet, whose architecture includes spectral methods, achieves the highest performance on the low timescale traffic prediction problem, model execution time is currently insufficient for real-time control loops of less than 10 ms. Hence, the evaluation in Sec. V is based on the straightforward DLinear model, combining fast inference time of below 400 µs with the second-best test performance. The optimized parameters comprise a learning rate of 10^{-3} and a batch size of 64; the model is trained for 200 epochs with Mean Squared Error (MSE) using the ADAM optimizer. Input datasets are pre-processed to mimic real-world randomized queuing delays and packet losses during the training process. After training, the models are optimized using constant folding and exported into Open Neural Network Exchange (ONNX) format for inference.

C. Provisioning of the Proactive Network Slices

For integration in the O-DU, we extend the *srsRAN Project* implementation with a Network Slice Selection Assistance Information (NSSAI) assignment inside the DU representation of UEs to support user stream identification. Based on the NSSAI assigned by the core network, the service type is identified, e.g., by a Slice Service Type (SST) of 2 [19] for URLLC. Thus indicating whether a predictive dApp is needed to optimize the latency-critical traffic. Specific learned traffic models are matched based on the Slice Descriptor (SD), enabling 2^{24} distinct slices. When a UE connects to an O-DU, its dApp is registered during initialization. When disconnecting, models are unregistered for resource efficiency. Using the concept of xApps, the dynamic dApp management can later be realized via the near-RT RAN Intelligent Controller (RIC). Using standard-conforming slice identifiers facilitates future end-to-end latency budgeting, including the core.

IV. evoBOT Locomotion Control and Real-Time Digital Twin

This section describes the locomotion control of the intralogistics robot evoBOT and the corresponding DT.

A. Real-Time evoBOT Digital Twin in NVIDIA Isaac Sim

The evoBOT, developed by the Fraunhofer Institute for Material Flow and Logistics (IML), is a highly dynamic autonomous mobile robot designed for intralogistics applications [2]. It transports objects of up to 40 kg at a max. speed of 10 m/s and operates on two wheels using an inverted pendulum mechanism. This versatility allows it to perform tasks traditionally divided among multiple types of robots.

The evoBOT simulation model closely matches the real robot, offering similar dynamics and sensor data. This DT thus decouples hard- and software development, reducing deployment cycles by prototyping ahead of physical construction via integration into advanced simulation tools like NVIDIA Isaac Sim and Isaac Gym. Hence, the model facilitates cost-effective testing of new control and navigation methods, object interactions, constructions, and sensors in a safe virtual environment. Further model details are available as open-source [2].

To effectively utilize the evoBOT DT, we employ the framework NVIDIA Isaac Sim. Critically, Isaac Sim does not allow for decoupling physics and rendering frequencies. We address this via two separate Isaac Sim instances. The first operates in headless mode and computes physics at a frequency of 200 Hz. This ensures that the control algorithms receive accurate and timely physics data, which is crucial for evoBOT's stability and performance. The second Isaac Sim instance handles rendering, visualizing the DT at a lower frequency of e.g. 50 Hz. This decoupling combines high-fidelity control with real-time visualization. By synchronizing both instances, we create a robust and efficient DT that closely mirrors evoBOT's real-world behavior. The resulting close integration within the real-world O-RAN testbed is depicted in Fig. 3.

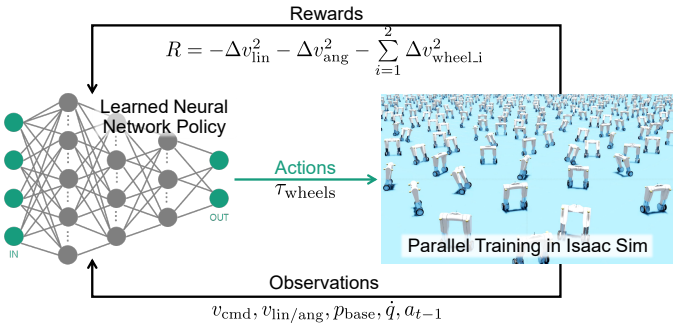

Fig. 4: Training setup using RL to train and deploy control policies using the evoBOT robot's Digital Twin with NVIDIA Isaac Sim.

B. Reinforcement Learning-based Locomotion Controller

To train control policies in the simulation-based environment, we utilize Isaac Gym [20], a cutting-edge GPU-accelerated physics simulation framework designed for robot learning. It allows us to train 4096 robot instances simultaneously, significantly accelerating the process. We employ Proximal Policy Optimization (PPO) [21] as robust learning algorithm. Resulting control policies are developed using fully-connected neural networks, with three hidden layers each.

Training control policies for dynamic motions involves defining the problem as Markov Decision Process (MDP) (see Fig. 4). Observations from the DT's physics simulation include linear and angular velocities $v_{\text{lin/ang}}$, pitch orientation p_{base}, wheel joint velocities $\dot{q}$, previous actions a_{t-1}, and velocity commands v_{cmd}. These are scaled, clipped, and normalized for consistency before local or edge inference of the neural network. The action space comprises the continuous torque commands τ_{wheels} for the wheels, scaled to allow dynamic responses to sharp changes. The reward function combines rewards for linear and angular velocity tracking and energy consumption based on the wheel's accelerations. The approach enhances tracking performance and encourages agile motions, enabling evoBOT to perform precise, dynamic tasks. Further details on both training parameters and reward formulations can be found in [2].

To evaluate the RL-based control performance, we utilize the following two metrics. The Mean Absolute Error (MAE), assessing velocity tracking accuracy by comparing commanded to actual velocities, is calculated as:

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^{n} |v_{\text{cmd},i} - v_{\text{act},i}| \tag{1}$$

with $v_{\text{cmd},i}$ for the commanded and $v_{\text{act},i}$ for the actual velocity at time step i. The average motor power is estimated as energy consumption based on the wheel torque and velocity over time:

$$P_{\text{avg}} = \frac{1}{T} \sum_{i=1}^{n} \left(|\tau_{\text{left},i} \cdot \omega_{\text{left},i}| + |\tau_{\text{right},i} \cdot \omega_{\text{right},i}| \right) \Delta t_i \tag{2}$$

where $\tau_{\text{left},i}$ and $\omega_{\text{left},i}$ as well as $\tau_{\text{right},i}$ and $\omega_{\text{right},i}$ are torques respectively angular velocities for the left and right wheels at experiment time step i. Δt_i is the interval between consecutive time steps, while T corresponds to the total elapsed time.

V. Experimental Setup and Results

In the following, the employed evaluation setup is detailed, as well as the results observed for latency-critical control.

A. Evaluation Scenario and Experimental Testbed

The experimental evaluation employs the intralogistics scenario introduced in Sec. I. The O-RAN testbed uses open source and Commercial Off-The-Shelf (COTS) components as depicted in Fig. 3. Two NI USRP X310 Software-Defined Radios (SDRs), synchronized via an OctoClock time and frequency reference to enable Multiple Input Multiple Output (MIMO) capability, are used as radio units for the srsRAN Project v24.04 based O-RAN base station. MIMO provides additional robustness via antenna diversity and is configured as 4x4 (4 layers) in the down- and 2x2 (1 layer) in the uplink.

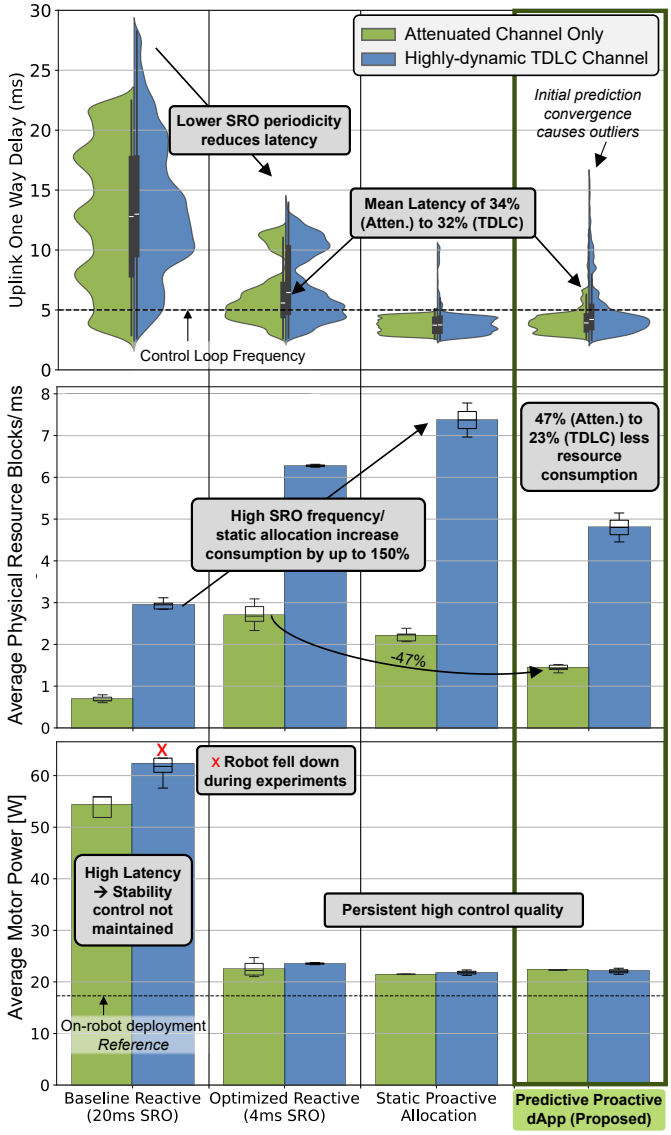

Fig. 5: Measured uplink one-way delay, Physical Resource Block (PRB) usage, and the experienced evoBOT robot power consumption based on (2) for the discussed scheduling strategies. The proposed proactive dApp offers the best latency/spectrum eff./power tradeoff.

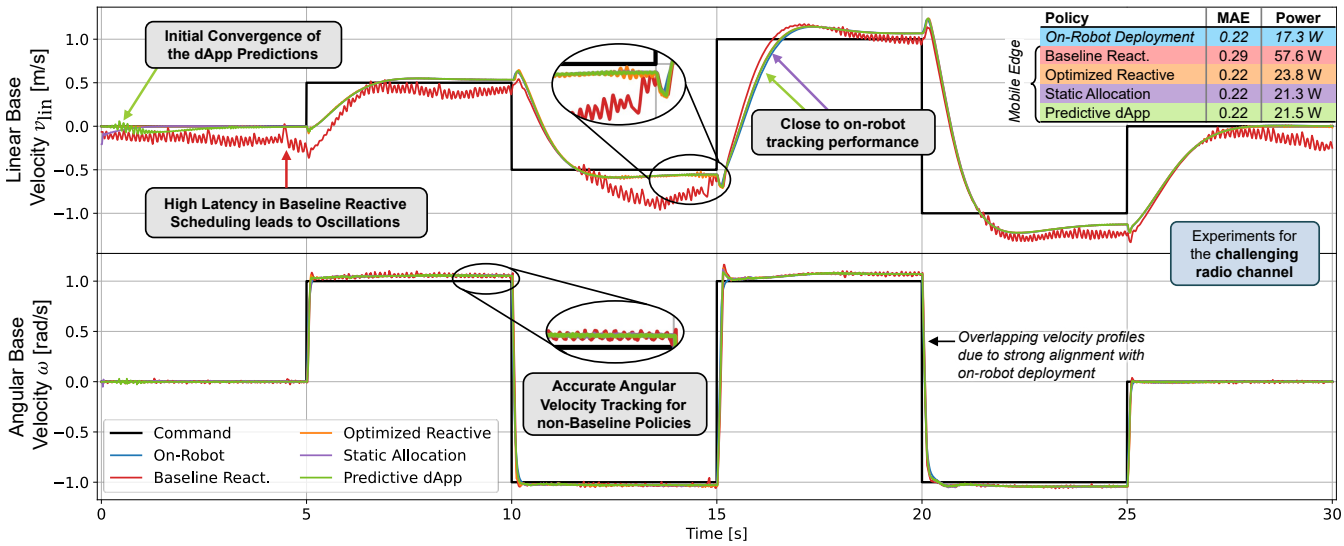

Fig. 6: Dynamic trajectory tracking performance of the networked RL-trained controller, evaluating the different scheduling policies in comparison to on-robot, i.e. on DT-host server deployment. The strategies show close to on-robot MAE tracking performance and similar power consumption, with significantly worse values for only baseline reactive scheduling (cf. table in upper right corner).

The specific cell parameters are as follows: 2 ms TDD periodicity and DDSU pattern enabling low latency, 30 kHz Subcarrier Spacing (SCS), 20 MHz bandwidth, chosen to mimic the effects of resource constraints of larger-scale scenarios. Scheduling Request Occasion (SRO) periodicity is varied between 20 and 4ms, while the time domain parameters $K1 = K2 = 2$ are optimized for latency, providing the best reactive scheduling possible. The evoBOT DT built within NVIDIA Isaac Sim is accelerated by an NVIDIA RTX 4090 GPU. The base station and DT are each hosted by servers comprised of AMD EPYC 7443 processors, 256 GB memory, and Ubuntu 22.04, with the O-RAN stack running on real-time kernel. Best effort traffic is meanwhile generated by a UE utilizing a Latte Panda Delta 3 single-board computer. DT and best effort UEs employ a Quectel RM520Q-GL 5G modem. Realistic radio conditions are created via a Keysight Propsim F64 channel emulator, including medium-correlation MIMO processing. For challenging channel conditions, a Tapped-Delay Line-C (TDLC) non-line-of-sight model defined by the 3GPP in [22] is emulated with a delay spread of 300, maximum doppler frequency of 100 Hz, and a robot speed of 5.56 m/s. The channel attenuation between base station and robot UE is 57 dB; 42 dB for the competing safety video UE.

B. Results of the Application-based Evaluation

The implemented radio resource scheduling approaches are compared by observed one-way latency and spectral efficiency. Furthermore, we discuss the impact of this communication delay on the evoBOT's control performance. We evaluate the one-way delay as application layer network traversal time between packet generation by the DT (i.e. robot sensor) and reception at the controller. Running locomotion control on the DT, i.e. on the robot itself, serves as baseline. For measurements, a reference trajectory with actions of desired angular base velocity $\omega = [-1, 1]$ rad/s and linear velocity $v_{\text{lin}} = [-1, 1]$ m/s is to be followed by the robot in real-time. Fig. 5 shows the relations between delay (top), spectral efficiency (center), and estimated motor power consumption after (2) by the evoBOT robot (bottom). Starting with a traditional absolute-priority slicing with the aforementioned latency optimizations, we evaluate the reactive approach on the left. Here, we focus first on a baseline SRO periodicity of 20 ms, following a latency-aware 4 ms-SRO adaptation, enabling higher-frequency uplink scheduling requests. Next, we allocate PRB statically, proactively providing dedicated resources to minimize latency (see third column of Fig. 5). At the very right, we evaluate the proposed ML-aided scheduling performed by the real-time dApp running the highly efficient DLinear models. At all settings, we evaluate the Key Performance Indicators (KPIs) for 60,000 closed-loop operations of the control policy, each using the 30 s reference trajectory. Experiments are performed using an ideal attenuated-only as well as a challenging TDLC channel, as introduced previously. In parallel, the competing safety monitoring UE fully loads its transmission queue for maximum load.

Starting with the baseline, the 20 ms SRO reactive slicing, the observed one-way latency exhibits substantial variation between 2.8 and 22.5 ms in congruence with expectations of up to 20 ms waiting time for sending scheduling requests. Due to the low K2 value, the time between grant issuing and uplink transmission is comparably short. While the difference in latency is slight between ideal and challenging channel conditions, resource consumption measured in PRBs per millisecond increases more than 4 times due to more robust coding. The high latency heavily influences control performance for the DT evoBOT, inducing more than 3 to 3.7 times (cf. table in Fig. 6) more energy consumption than local on-robot deployment. During an experiment with the challenging channel, control lost balance, causing the robot to fall. Thus, latency-optimized strategies are targeted in the following.

Reducing SRO periodicity for shorter inter-scheduling-request times, the optimized reactive scheduling strategy shows a significantly lower mean latency of 6.3 ms for the ideal and 7.3 ms for the TDLC channel. However, resource usage is vastly increased by more than 3 (ideal channel) and 2 times (TDLC) due to high scheduling overhead. Yet, control performance highly improves, with power rising by only 6.5 W versus on-robot operation. Static resource assignment reduces latency to a mean of 3.7 to 4 ms. These assignments are fixed and wasted if not used. Notably, for ideal channels, this is more power efficient than the optimized reactive approach. Yet, for adverse channels, more PRBs are consumed when the overhead of reactive scheduling reduces. Locomotion performance is close to on-robot with only 4 W power increase, still at the cost of high PRB usage, especially for complex channels.

The proposed approach using the predictive scheduling dApp providing tailored resources in time can maintain the delay performance of the static approach with minor outliers, resulting in a mean latency of 4.1 to 4.9 ms. At the start of transmissions, we initially observe higher latencies, which vanish after convergence of the model's predictions to the transmission pattern. Compared to the optimized reactive policy, under attenuated conditions, latencies reduce by 34 % while using 47 % less resources, while for challenging TDLC channels latencies are reduced by 33 % combined with 23 % less resource usage. Still, control performance is maintained with just a 4.2 W power increase over on-robot deployment.

Trajectory tracking in Fig. 6 illustrates control performance by plotting commanded velocity (black) for linear and angular base velocity compared to all studied approaches for the TDLC channel. While the reactive baseline policy shows strong oscillations in linear velocity, the other approaches align tightly with on-device controller performance. Still, the optimized reactive approach shows some oscillations, while static and dApp-based policies closely mimic the on-robot trajectory after initial convergence. Angular velocity is tightly tracked by all policies despite the reactive baseline.

VI. Conclusion and Outlook

This work demonstrates networked real-time control of a two-wheeled inverted pendulum robot via an O-RAN network. With the proposed ML-based proactive resource scheduling dApp, we enable 24 % decreased MSE on a reference trajectory tracking versus the baseline reactive approach. The employed DLinear model enables the prediction of exact packet arrival times and sizes to reduce the mean uplink latency by 34 % and additionally lower PRB usage by 47 % for ideal conditions and 23 % for challenging TDLC channels. In future work, we will further evaluate the influence of mobility and integrate channel prediction to improve robustness.

Acknowledgment

This work has been partly funded by the Federal Ministry of Education and Research (BMBF) via the project *6GEM* under funding reference 16KISK038, the Lamarr-Institute for Machine Learning and Artificial Intelligence (LAMARR22B), and the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under project number 508759126. The authors would like to thank Nvidia's Isaac Sim team for their continuous support.

References

[1] S. D'Oro, M. Polese, L. Bonati, H. Cheng, and T. Melodia, "dApps: Distributed Applications for Real-Time Inference and Control in O-RAN," *IEEE Comm. Magazine*, vol. 60, no. 11, pp. 52–58, 2022.

[2] P. Klokowski, J. Eßer, *et al.*, "evoBOT – Design and Learning-Based Control of a Two-Wheeled Compound Inverted Pendulum Robot," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023, pp. 10 425–10 432.

[3] 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 Computer Communications Workshops (INFOCOM WKSHPS)*, 2024, pp. 01–06.

[4] 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)*, 2024.

[5] 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.

[6] R. Wiebusch, N. A. Wagner, D. Overbeck, F. Kurtz, and C. Wietfeld, "Towards Open 6G: Experimental O-RAN Framework for Predictive Uplink Slicing," in *IEEE International Conference on Communications (ICC)*, 2023, pp. 4834–4839.

[7] C. Fiandrino, E. Perez Gomez, P. Férnandez Pérez, H. Mohammadalizadeh, M. Fiore, and J. Widmer, "AIChronoLens: Advancing Explainability for Time Series AI Forecasting in Mobile Networks," in *IEEE Int. Conf. on Computer Communications (INFOCOM)*, 2024.

[8] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam, "A Time Series is Worth 64 Words: Long-term Forecasting with Transformers," in *International Conference on Learning Representations*, 2023.

[9] 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*, 2023.

[10] H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, and M. Long, "TimesNet: Temporal 2D-Variation Modeling for General Time Series Analysis," in *International Conference on Learning Representations*, 2023.

[11] D. Rodriguez and S. Behnke, "DeepWalk: Omnidirectional Bipedal Gait by Deep Reinforcement Learning," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 3033–3039.

[12] G. B. Margolis, G. Yang, K. Paigwar, T. Chen, and P. Agrawal, "Rapid locomotion via reinforcement learning," *The International Journal of Robotics Research*, vol. 43, no. 4, pp. 572–587, 2024.

[13] J. Eßer, N. Bach, C. Jestel, O. Urbann, and S. Kerner, "Guided Reinforcement Learning: A Review and Evaluation for Efficient and Effective Real-World Robotics [Survey]," *IEEE Robotics & Automation Magazine*, vol. 30, no. 2, pp. 67–85, 2023.

[14] M. Wiedemann, O. Ahmed, A. Dieckhöfer, R. Gasoto, and S. Kerner, "Simulation Modeling of Highly Dynamic Omnidirectional Mobile Robots Based on Real-World Data," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2024, pp. 16 923–16 929.

[15] W. Zhao, J. P. Queralta, and T. Westerlund, "Sim-to-Real Transfer in Deep Reinforcement Learning for Robotics: a Survey," in *IEEE Symposium Series on Comp. Intelligence (SSCI)*, 2020, pp. 737–744.

[16] Software Radio Systems (SRS), "srsRAN Project: Open source O-RAN 5G CU/DU solution. v24.04," Apr. 2024. [Online]. Available: https://docs.srsran.com/projects/project/en/latest/.

[17] S. Lee *et al.*, "Open5GS. Open Source implementation for 5G Core and EPC (Release-17) v2.7.0," Apr. 2024. [Online]. Available: https://open5gs.org/open5gs/docs/.

[18] A. Paszke, S. Gross, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, 2019, pp. 8024–8035.

[19] 3rd Gen. Partnership Project (3GPP), "System architecture for the 5G System (5GS); Stage 2," TS 23.501, Jun. 2024, v19.0.0.

[20] V. Makoviychuk, L. Wawrzyniak, *et al.*, "Isaac Gym: High Performance GPU Based Physics Simulation For Robot Learning," in *35th Conf. on Neural Information Processing Systems (NeurIPS)*, 2021.

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

[22] 3rd Gen. Partnership Project (3GPP), "Study on channel model for frequencies from 0.5 to 100 GHz," TR 38.901, Apr. 2024, v18.0.0.