Intelligent Spectrum Coexistence in UAV-Enabled NextG Networks: Algorithms and Testing Framework

by

Jiangqi Hu

June 1 2024

A dissertation submitted to the

Faculty of the Graduate School of

the University at Buffalo, The State University of New York

in partial fulfilment of the requirements for the

degree of

Doctor of Philosophy

Department of Electrical Engineering

"Scientists investigate that which already is; engineers create that which has never been."

- *Albert Einstein*

To my parents, friends, and beloved ones.

Acknowledgments

I extend my deepest appreciation to my advisor, Dr. Zhangyu Guan, whose unwavering enthusiasm for research and teaching has been truly inspiring. His guidance and steadfast support have been instrumental in bringing this dissertation to fruition.

I am grateful to my committee members, Dr. Weifeng Su, Dr. Nicholas Mastronarde, and Dr. Shaofeng Zou, whose invaluable feedback and guidance have contributed significantly to the completion of this dissertation.

I would like to acknowledge Dr. Katherine Bartelo, my academic coordinator, for her assistance not only with academic matters but also with personal endeavors. I also extend my thanks to Jason Tillman, Kimberly Kriz, and Mary Busch for their administrative support.

My heartfelt thanks go to my fellow researchers at the UB WINGS Lab - Sabarish Krishna Moorthy, Maxwell McManus, Yuqing Cui, Zhaoxi Zhang, Zhiyuan Zhao, Ankush Harindranath, Sidharth Santhinivas, Ishita Dhopeshwar, Haolun Li, and Hao Li - for their collaboration and unwavering support.

Finally, heartfelt thanks to my parents, whose unwavering support and sacrifices have been the bedrock of my journey. For over three decades, they have stood by me, selflessly nurturing and believing in me. Their love and support have been invaluable, and I am deeply grateful for their role in helping me reach this milestone.

Abstract

Unmanned Aerial Vehicles (UAVs) are envisioned as pivotal technologies for the next generation of wireless networks due to their rapid deployment, high mobility, and compact size. To facilitate the widespread adoption of unmanned vehicles and unlock a diverse array of applications, numerous challenges must be addressed, such as limited spectrum resources and the inherent complexities of environments that are difficult to capture using traditional mathematical models.

To mitigate the long-term spectrum crunch problem, the FCC recently opened up the 6 GHz frequency band for unlicensed use. However, the existing spectrum sharing strategies cannot support the operation of UAVs. This is primarily because of the directionality-based spectrum sharing among the incumbent systems in this band and the high mobility of the moving vehicles, which together make it challenging to control the cross-system interference. To circumvent the limitations of traditional mathematical modeling in designing algorithms and ensure rigorous and repeatable experimental evaluations of wireless networked systems, the community has made significant efforts to develop experimentation platforms. Yet, these existing platforms have primarily focused on the data plane—that is, the forwarding infrastructure—without adequately addressing the control plane.

To tackle these challenges, we propose *SwarmShare*, a mobility-resilient spectrum sharing framework for swarm UAV networking in the 6 GHz band. In *SwarmShare*, we first present a mathematical formulation of the *SwarmShare* problem, where the objective is to maximize the spectral efficiency of the UAV network by jointly controlling the flight and transmis-

sion power of the UAVs and their association with the ground users, under the interference constraints of the incumbent system. We find that there are no closed-form mathematical models that can be used to characterize the statistical behaviors of the aggregate interference from the UAVs to the incumbent system. Then we propose a data-driven three-phase spectrum sharing approach, including *Initial Power Enforcement*, *Offline-dataset Guided Online Power Adaptation*, and *Reinforcement Learning-based UAV Optimization*. We validate the effectiveness of *SwarmShare* through an extensive simulation campaign. Results indicate that, based on *SwarmShare*, the aggregate interference from the UAVs to the incumbent system can be effectively kept below the target level *without* requiring the real-time cross-system channel state information. The mobility resilience of *SwarmShare* is also validated in coexisting networks with *no* precise UAV location information.

In the meantime, to bridge the gap between the control plane and data plane design, and to facilitate AI/ML-driven control of wireless communication networks and thorough testing the resulting algorithms in realistic settings, we develop *NeXT*, a software-defined playground with integrated wireless network simulation, experimentation and optimization capabilities. We first design the data plane, which integrates an event-driven broadband wireless network simulator called *UBSim* and a software-defined wireless network testing facility called *RoboNet*. We then design *NeXT*'s control plane, where a software toolchain is developed and deployed to support both traditional model-based optimization and new data-driven control techniques. We showcase the experimentation capability of *NeXT* considering a series of optimization and control problems in different wireless networks.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Unmanned aerial vehicles (UAVs) have been envisioned as a key technology for next-generation (i.e., B5G or 6G) wireless networks [1–8]. Because of their features of fast deployment, high mobility and small size, UAVs have a great potential to enable a wide set of new applications, including UAV-aided guidance [9, 10], small cells with flying base stations [11], emergency wireless networking in the aftermath of disasters [12], among others. The foreseen wide adoption of UAV systems can pose a significant burden on the capacity of the underlying wireless networks. In this dissertation, we aim to explore new approaches that can enable UAV operations in the 6 GHz band to harvest the additional 1.2 GHz spectrum bandwidth [13].

The primary challenge towards this goal is in the spectrum sharing approaches adopted by the incumbent systems in this frequency band. The 6 GHz band consists of four sub-bands, i.e., U-NII-5 (5.925-6.425 GHz), U-NII-6 (6.425-6.525 GHz), U-NII-7 (6.525-6.875 GHz), and U-NII-8 (6.875-7.125 GHz). These bands have been previously occupied by a set of non-government services, including fixed point-to-point services, fixed-satellite service (Earth-to-space), broadcast auxiliary services and cable television relay services [13]. These incumbent systems coexist with each other by sharing the spectrum on a directional basis, i.e., they use highly directional antennas to concentrate the signal energy in a particular

direction such that the mutual interference can be effectively mitigated as long as their antennas are not pointed toward each other. As a result, traditional carrier-sensing-based spectrum sharing as in Wi-Fi networks is not applicable to extend those wireless systems with omnidirectional antennas to this frequency band, because of the low detectability of the incumbent systems [14–20]. For this reason, two operation modes have been proposed by the FCC, i.e., standard-power and low-power modes. The former allows both indoor and outdoor operations on the U-NII-5 and U-NII-7 bands with maximum transmission power of 30 dBm. The latter focuses on indoor operations in the U-NII-6 and U-NII-8 bands with maximum transmission power of 24 dBm.

However, neither of the above two modes supports UAV operations in the 6 GHz bands [13, 21]. A major concern is that the high mobility of the UAV systems makes it difficult to model and control their aggregate interference to the incumbent systems. The situation gets even worse when considering the altitude-dependent interference range of UAVs and the higher probability of line-of-sight signal propagation at higher altitudes. Additionally, it is also challenging for the distributed UAVs to control their aggregate interference collaboratively by jointly considering their spectrum access strategies and association to the ground users. This complexity in managing UAV interference underscores the pressing need for innovative approaches to wireless network research and development. Specifically, it highlights the importance of creating robust experimental platforms that can simulate the real-world conditions UAVs operate in, allowing for the testing and refinement of solutions to these challenges.

In the past decades, the evolution of wireless network systems has significantly changed and will continue to change the way we live and work, our commercial activities as well as national security. However, as of today the wireless research community is still lacking a mature ecosystem to support rigorous and repeatable experimental evaluation of wireless networked systems. To fill this gap, significant efforts have been made by the community. A recent milestone is the NSF Platforms for Advanced Wireless Research (PAWR) program,

which attempts to develop four large-scale outdoor experimentation platforms for advanced wireless research [22]. As of today, all of them have already been developed and are available to the wireless community. These are POWDER-RENEW for experiments in the sub-6 GHz frequency bands [23], COSMOS for experiments in both sub-6 GHz and mmWave frequency bands as well as edge computing [24], AERPAW for experiments with wireless unmanned aerial vehicles (UAVs) [25] and ARA for smart and connected rural communities [26].

While existing community shared facilities have significantly advanced experimental research for new wireless systems, it is still challenging to fully meet the needs of experimental wireless research in the era of data-driven networking. First, to simplify the modeling, control and optimization of heterogeneous NextG networks, data-driven control based on Artificial Intelligence (AI) and Machine Learning (ML) has attracted significant research attention [27–30]. However, the effectiveness of AI/ML algorithms largely relies on sufficient well-labeled data for policy training [31–34]. It is typically time consuming and sometimes unsafe to collect training data in real-world environments [35–37]. Second, the design, prototyping and verification of new network control algorithms require engineers to grapple simultaneously with mathematical modeling, distributed control, protocol design across different layers of the protocol stack, as well as their implementation and deployment. This process is typically complex, tedious and error-prone.

In this dissertation, we firstly focus on a new spectrum sharing scenario in the 6 GHz band called *SwarmShare*, where a set of UAVs collaboratively provide data streaming services to ground users, by sharing the spectrum with the incumbent systems on the 6 GHz band under the cross-system interference constraints. Within this framework, we model and analyze the aggregate interference from the UAVs to the incumbent users, and propose a mobility-resilient stochastic spectrum sharing approach, based on which the interference can be mitigated from the coexisting UAV networks to the incumbent users in the 6 GHz band, while increasing the SINR of the UAV networks.

Secondly, to address the challenges outlined previously, a pivotal element of our disser-

tation is the development of *NeXT, a software-defined wireless Network X-Control Testbed, where "X" refers to optimization, simulation and experimentation* [38,39]. In a nutshell,*NeXT* provides an integrated testing framework, in which researchers are allowed to generate in an automated manner distributed cross-layer network optimization algorithms, simulate the generated algorithms in software, and then validate the simulation results based on testbed experiments. There are two planes in *NeXT*, *Data Plane* and *Control Plane*. The former provides simulation and experimentation capabilities, and the latter implements network optimization and control functionalities.

The key contributions of this dissertation are:

- We first present a mathematical formulation of the *SwarmShare* problem, where the objective is to maximize the spectral efficiency of the wireless UAV network by jointly controlling the UAVs' transmission power and flight trajectory as well as their association to the ground users, under the interference constraints of the incumbent system. It is shown that the resulting problem is a mixed integer nonlinear non-convex programming (MINLP) problem.

- We analyze the statistical behavior of the aggregate interference from the UAVs to the incumbent system, and find that no existing models can be used to characterize the statistical behavior of the interference. With this observation, we propose to solve the above MINLP spectrum sharing problem following a data-driven three-phase approach: *Initial Power Enforcement*, *Offline-dataset Guided Online Power Adaptation*, and *Reinforcement Learning-based UAV Optimization*.

- We validate the effectiveness of *SwarmShare* by conducting an extensive simulation campaign over UBSim. It is found that, with *SwarmShare*, effective spectrum sharing can be achieved *without* real-time cross-system channel state information and, which is somewhat surprising, even with *no* precise location information of the UAVs.

- We design the data plane for the*NeXT* testbed. In this plane, we first integrate

UBSim with*NeXT* for software-based network simulation. UBSim is a newly developed Universal Broadband Simulator for broadband (microwave, mmWave and terahertz bands) aerial and ground wireless networking. We also develop a testing facility for mobile networks based on software defined radios (SDRs).

- We then design *NeXT*'s control plane, which supports traditional model-based control and new data-driven control techniques. For the former, Wireless Network Operating System (WNOS) [40] has been deployed to enable automated generation of distributed cross-layer control algorithms. For the latter, a reinforcement learning (RL) repository is developed supporting various RL algorithms. A scheme to automatically adjust robots' posture and positions is proposed to mitigate the error introduced by the mobile hotspots.

- We enhance the Software radio suite Radio Access Network (srsRAN) by proposing the integration of Evolved Node B (srsENB) and Evolved Packet Core (srsEPC) through wireless links. This approach enables the effortless inclusion of multiple potentially mobile srsENBs into experimental setups. Two compelling demonstrations are conducted to validate the effectiveness and scalability of this innovative srsRAN architecture.

- We showcase the optimization, simulation and experimentation capabilities of the*NeXT* testbed considering a series of wireless network control problems. These include narrowband multi-hop communications, srsRAN-based cellular networks and millimeter wave (mmWave)-band communications. A set of application programming interfaces (APIs) have been designed to simplify access to *NeXT*'s data and control planes.

The rest of the dissertation is organized as follows. In Chapter 2, we discuss the related works. The system model, problem formulation, the spectrum sharing framework of *SwarmShare* and performance evaluation results are presented in Chapter 3. Chapter 4 elaborates on the design of the data plane and control plane of *NeXT*, the scalability of srsRAN,

example experimental results, and the new research opportunities enabled. Finally, the main conclusions are drawn in Chapter 5.

Chapter 2

Related Work

UAV-assisted Networks UAV systems have attracted significant research attention in both academia and industry [1, 41–45]. For example, in [1] the authors optimize the achievable rate of UAV-aided cognitive IoT networks. Wang et al. propose in [41] a dynamic hyper-graph coloring approach for spectrum sharing in UAV-assisted networks. In [42], the authors optimize mobile terminals' throughput by jointly controlling UAV trajectory, bandwidth allocation and user partitioning between the UAV and ground base stations. In [43], a UAV is used as a relay to assist D2D communications. In [44], the authors studies machine learning based spectrum sharing for UAV-assisted emergency communications. The authors of [45] adopt UAVs to harvest the primary RF signal as an energy source of the secondary users. Readers are referred to [46], [47] and references therein for a survey of the main results in this area.

Spectrum sharing Spectrum sharing in cognitive radio networks has also been a hot research topic for a long time with a sizable and increasing body of literature. In [48], the authors aim to maximize the average secrecy rate of the secondary network by robustly optimizing the UAV's trajectory and transmit power. In [49], the authors maximize the revenue of the newly joined systems in cognitive radio networks by controlling the channel access of new users. The authors of [50] propose a cognitive backscatter network to maximize

the data rate of the newly joined networks. By leveraging recent advances in MIMO, the authors enable transparent spectrum sharing for a small cognitive radio network in [51]. A deep reinforcement learning based power control scheme is designed in [52] to meet the QoS requirements of both primary and secondary users. Based on a combination of model-free and model-based reinforcement learning, the authors of [53] propose a dynamic spectrum access scheme for secondary users with imperfect sensing.

Spectrum sharing between directional- and omnidirectional-antenna wireless systems has also been studied in existing literature. For example, authors in [54] optimize the performance of LTE-Unlicensed networks while guaranteeing the performance of the co-located radar system. The authors of [55] propose RadChat, a distributed networking protocol for mitigation of interference among frequency modulated continuous wave radars. A cooperative spectrum sharing model is proposed in [56] to mitigate the mutual interference among radar and communication systems. In [57], the authors propose a framework for spectrum sharing between satellite and terrestrial networks and analyze the interference in both downlink and uplink from terrestrial cellular systems and nongeostationary systems to geostationary systems within the framework. Please refer to [58], [59] and references therein for a good survey of the main results in this field.

Further comparison of the above discussed references in different aspects is summarized in Table 2.1. From the table, it can be seen that no existing work considers spectrum sharing in the 6 GHz band between directional and omnidirectional mobile wireless communication systems.

Experimental Platforms A lot of testbeds have been proposed and established to meet the needs of experimentation and verifying algorithms in the real world. For example, the NSF PAWR program aims to enable experimental wireless communications research across devices, communication techniques, networks, systems, and services conceived by the US academic and industrial wireless research community and deployed in partnership with local communities [22]. POWDER is a platform for testing future wireless communications and

Table 2.1: Comparison of related work in different aspects

Reference	Spectrum Band	Mobility Support	Antenna Type	Performance Metric
[1]	2.4 GHz	Yes	Omnidirectional	Throughput
[41]	Unknown	Yes	Omnidirectional	User Number
[42]	Unknown	Yes	Omnidirectional	Throughput
[43]	Unknown	Yes	Omnidirectional	Throughput
[44]	2 GHz	Yes	Omnidirectional	Throughput
[45]	840.5 - 845.5 MHz	No	Omnidirectional	Secrecy Outage Probability
[48]	Unknown	Yes	Omnidirectional	Throughput
[49]	Unknown	No	Omnidirectional	Throughput
[50]	Unknown	No	Omnidirectional	Throughput
[51]	2.48 GHz	No	Omnidirectional	Interference
[52]	Unknown	No	Omnidirectional	Throughput
[53]	Unknown	No	Omnidirectional	Sample Efficiency
[54]	5 GHz	No	Directional	Power
[55]	Unknown	No	Directional	Interference
[56]	Below 100 MHz	No	Directional	Throughput
[57]	18 GHz	No	Directional	Interference

networking technologies in a city-scale "living laboratory" [23]. COSMOS aims at design, development, and deployment of a city-scale advanced wireless testbed to support real-world experimentation on next-generation wireless technologies and applications [24]. Colosseum is the world's largest network emulator providing researchers with testing at scale, offsetting the site specificity of a physical testbed [60]. AERPAW is the first aerial wireless experimentation platform spanning 5G technologies and beyond and with the potential to create transformative wireless advances for aerial systems [25]. In [61], the world's first fully programmable and open-source massive-multiple input multiple output (MIMO) platform named RENEW is introduced. ARA enables the research and development of rural-focused wireless technologies that provide affordable, high-capacity connectivity to rural communities and industries such as agriculture [26]. However, these platforms either do not consider mobile nodes or do not provide data-driven tools that simplify the experimentation process.

A unique national research infrastructure called FABRIC is proposed in [62] to enable cutting-edge and exploratory research at-scale in networking, cybersecurity, distributed computing and storage systems, machine learning, and science applications. *DeterLab* is a shared testbed providing a platform for research in cybersecurity and serving a broad user commu-

nity [63]. An open-source platform called M^3 is designed in [64, 65] to facilitate research in 5G vehicular networking and automotive sensing. In [66], a community-shared, open-source, open-architecture infrastructure for mobile underwater wireless networks called *mu-Net* is proposed. Readers are referred to [67, 68] for more information about the aforementioned testbeds. Arena is an open-access wireless testing platform [69] that can be used to test key wireless technologies, such as synchronized MIMO transmission schemes, multi-hop ad hoc networking, multi-cell long term evolution (LTE) networks, and spectrum sensing for cognitive radio. The authors in [70] propose the SkyHaul platform for channel modeling in mobile scenarios. An integrated testbed TeraNova for ultra-broadband wireless communications is developed in [71], which supports the testing and validation of new terahertz (THz) channel models and physical layer solutions. A testbed based on FlockLab [72] deployed in a campus-scale is designed in [73] to better support testing of long-range communications. However, these primarily focus on the physical platform's development, while neglecting the potential benefits of pairing the physical testbed with a simulator (e.g., using the simulator to accelerate the training of AI/ML algorithms).

In this dissertation, firstly we aim to *design a new, mobility-resilient spectrum sharing framework between UAVs and incumbent wireless systems in the 6 GHz band to improve spectral efficiency and maximize the throughput of the UAV network for elastic applications such as data collection, file transfer, and messages.* Secondly unlike the testing facilities previously discussed that primarily concentrate on developing the forwarding infrastructure, or the data plane, our focus extends to both data and control planes. To this end we develop *a software-defined testbed with integrated simulation, experimentation, and optimization capabilities for mobile wireless networks.*

Chapter 3

SwarmShare: Mobility-Resilient Spectrum Coexistence in 6 GHz Band

In this chapter, we present *SwarmShare*, an innovative mobility-resilient stochastic spectrum sharing approach that empowers UAVs to operate within the 6 GHz band. To achieve this objective, we begin by outlining the system model and problem formulation in Chapter 3.1. Subsequently, we delve into the intricacies of designing a spectrum coexistence framework aimed at addressing the *SwarmShare* control problem, elucidated in Chapter 3.2. Finally, we evaluate the performance of the proposed framework within a simulated environment in Chapter 3.3.

3.1 System Model and Problem Formulation

As shown in Figure 3.1, we consider a wireless UAV network coexisting with an incumbent communication pair (Tx and Rx) by sharing the same portion of spectrum B in the 6 GHz band. The UAV network consists of a set $\mathcal{K}$ of UAVs collaborating with each other to serve a set $\mathcal{M}$ of ground users. The transmission time is divided into a set $\mathcal{T}$ of consecutive time slots. In each time slot $t \in \mathcal{T}$, denotes the coordinate vector of UAV $k \in \mathcal{K}$ as $\mathbf{cod}_k^t = [x_k^t, y_k^t, z_k^t]^{\mathrm{T}}$, with T being the transpose operation and x_k^t, y_k^t and z_k^t representing the x-, y- and z-axis

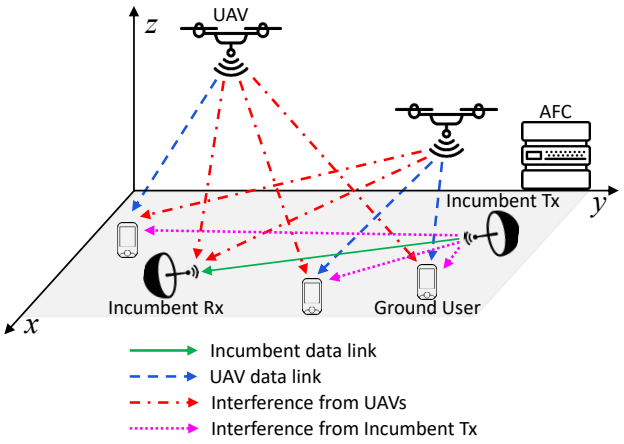


Figure 3.1: Spectrum sharing between coexisting UAV network and the incumbent network in the 6 GHz bands.

components, respectively. Similarly, denote respectively $\mathbf{cod}_{\mathrm{Tx}} = [x_{\mathrm{Tx}}, y_{\mathrm{Tx}}, z_{\mathrm{Tx}}]^{\mathrm{T}}$, $\mathbf{cod}_{\mathrm{Rx}} = [x_{\mathrm{Rx}}, y_{\mathrm{Rx}}, z_{\mathrm{Rx}}]^{\mathrm{T}}$ and $\mathbf{cod}_i = [x_i, y_i, z_i]^{\mathrm{T}}$ as the coordinate vectors of incumbent transmitter Tx, incumbent receiver Rx and ground node $i \in \mathcal{M} \cup \{\mathrm{Tx}, \mathrm{Rx}\}$. Denote $\mathcal{A} = \mathcal{K} \cup \mathcal{M} \cup \{\mathrm{Tx}, \mathrm{Rx}\}$ as the set of all the nodes in the heterogeneous network. The objective of the UAV network is to maximize its own spectral efficiency under the interference constraints of the incumbent system. Before presenting the formal formulation of the spectrum sharing problem, we first describe the considered channel, antenna and throughput models.

3.1.1 Channel Model

We consider both large-scale path-loss and small-scale fading. For path-loss, we consider line-of-sight (LoS) wireless channels between the incumbent transmitter Tx and its receiver Rx. This is feasible because the incumbent systems are usually carefully deployed such that their antennas are well-aligned without any obstructions in the link. We consider non-line-of-sight (NLoS) links between the incumbent nodes and the ground users of the coexisting networks. For UAV networks, we consider both LoS and NLoS links. Specifically, we consider as in [74] a probabilistic path-loss model for the links between UAVs and ground nodes. Then, the LoS and NLoS path-loss (in dB) between UAV $k \in \mathcal{K}$ and ground node

$i \in \mathcal{M} \cup \{\text{Tx}, \text{Rx}\}$ can be given as, in time slot $t \in \mathcal{T}$,

$$H_{ki}^{\text{LoS},t} = 20 \log \left(\frac{4\pi d_{ki}^t f}{c} \right) + \eta^{\text{LoS}}, \tag{3.1}$$

$$H_{ki}^{\text{NLoS},t} = 20 \log \left(\frac{4\pi d_{ki}^t f}{c} \right) + \eta^{\text{NLoS}}, \tag{3.2}$$

where the first term on the right-hand side of (3.1) and (3.2) represents the free space path-loss with $d_{ki}^t = ||\mathbf{cod}_k^t - \mathbf{cod}_i||_2$ being the distance between UAV k and receiver i in time slot t, f is the carrier frequency of UAV k, c is the speed of light, and η^{LoS} and η^{NLoS} are the additional attenuation factors due to LoS and NLoS transmissions, respectively. Let $\Pr(H_{ki}^{\text{LoS},t})$ represent the probability of LoS transmissions in time slot t, then $\Pr(H_{ki}^{\text{LoS},t})$ can be expressed as [75],

$$\Pr(H_{ki}^{\text{LoS},t}) = (1 + X \exp(-Y[\phi_{ki} - X]))^{-1}, \tag{3.3}$$

where X and Y are given environment-dependent constants and $\phi_{ki} = \sin^{-1}(z_k^t / d_{ki}^t)$. Accordingly, the probability of NLoS transmissions between UAV $k \in \mathcal{K}$ and receiver $i \in \mathcal{M} \cup \{\text{Tx}, \text{Rx}\}$ can be given as $\Pr(H_{ki}^{\text{NLoS},t}) = 1 - \Pr(H_{ki}^{\text{LoS},t})$.

Finally, for small-scale fading we consider Rician fading for LoS transmissions and Rayleigh fading for NLoS. Denote K_{ij} as the Rician factor for the wireless channel between nodes $i, j \in \mathcal{A}$, then K_{ij} can be given as $K_{ij} = 13 - 0.03 d_{ij}$ for LoS transmissions and 0 for NLoS, where d_{ij} is the distance between the two nodes. Denote the resulting small-scale fading coefficient as $h_{ij}^t \triangleq h_{ij}^t(K_{ij})$ for nodes $i, j \in \mathcal{A}$.

3.1.2 Antenna Model

As described in Chapter 1, in our research we consider directional transmissions for the incumbent wireless systems and omnidirectional transmissions for the coexisting UAV network. Specifically, we consider as in [76] a bi-sectorized antenna model to characterize

the interference between directional and omnidirectional antennas. Denote θ_{Tx} and θ_{Rx} as the signal beamwidth of the incumbent transmitter and receiver's antennas, respectively. Let $\theta_m \in [-\pi, \pi]$ denote the offset angle of the boresight direction of the Tx's antenna with respect to the reference direction for ground user $m \in \mathcal{M}$. Here, the reference direction refers to the direction along which the Tx's antenna would be exactly pointed to user m. Then the antenna gain of incumbent transmitter Tx with respect to ground user $m \in \mathcal{M}$ in time slot t, denoted as $w^t_{m\mathrm{Tx}}$, can be written as

$$w^t_{m\mathrm{Tx}} = \begin{cases} w^{\max}_{\mathrm{Tx}}, & \text{if } \theta_m \leq \theta_{\mathrm{Tx}} \\ w^{\min}_{\mathrm{Tx}}, & \text{otherwise} \end{cases}, \tag{3.4}$$

where $w^{\max}_{\mathrm{Tx}}$ and $w^{\min}_{\mathrm{Tx}}$ represent the maximum and minimum transmit gains of the incumbent transmitter, respectively. Similarly, the receive gain of the incumbent receiver Rx with respect to UAV $k \in \mathcal{K}$, denoted as $w^t_{k\mathrm{Rx}}$, can be given as

$$w^t_{k\mathrm{Rx}} = \begin{cases} w^{\max}_{\mathrm{Rx}}, & \text{if } \theta_k \leq \theta_{\mathrm{Rx}} \\ w^{\min}_{\mathrm{Rx}}, & \text{otherwise} \end{cases}, \tag{3.5}$$

with $w^{\max}_{\mathrm{Rx}}$ and $w^{\min}_{\mathrm{Rx}}$ being the maximum and minimum receive gains of the incumbent receiver, respectively. The transmit and receive gains are set to the maximum values for incumbent transmissions, i.e., $w^{\max}_{\mathrm{Tx}}$ and $w^{\max}_{\mathrm{Rx}}$, respectively.

3.1.3 Throughput Model

Based on the above channel and antenna models, the signal-to-interference-plus-noise ratio (SINR) of the incumbent receiver Rx, denoted as γ^t_{Rx} for time slot t, can be written as

$$\gamma^t_{\mathrm{Rx}} = \frac{p_{\mathrm{Tx}} w^{\max}_{\mathrm{Tx}} w^{\max}_{\mathrm{Rx}} \cdot (h^t_{\mathrm{TxRx}})^2 / H^{\mathrm{LoS}}_{\mathrm{TxRx}}}{\sum\limits_{k \in \mathcal{K}} p^t_k w^t_{k\mathrm{Rx}} w_k \cdot (h^t_{k\mathrm{Rx}})^2 / H^t_{k\mathrm{Rx}} + (\sigma_{\mathrm{Rx}})^2} \tag{3.6}$$

where p_{Tx} and p_k^t represent the transmission power of the incumbent transmitter Tx and UAV $k \in \mathcal{K}$ in time slot $t \in \mathcal{T}$, respectively; w_k denotes the transmit gain of the UAV and is considered to be constant for omnidirectional antennas; and $(\sigma_{\mathrm{Rx}})^2$ is the power of Additive White Gaussian Noise (AWGN) at the incumbent receiver.

The objective of *SwarmShare* is to guarantee satisfactory SINR for the incumbent system (i.e., γ_{Rx}^t) by controlling the transmission power of the coexisting UAVs. To this end, we consider a single-home association strategy for the ground users of the UAV network, i.e., in each time slot $t \in \mathcal{T}$ each ground user can be served by at most one UAV. Denote α_{km} as the association variable, with $\alpha_{km} = 1$ if ground user $m \in \mathcal{M}$ is associated with UAV $k \in \mathcal{K}$ and $\alpha_{km} = 0$ otherwise. Then we have

$$\sum_{k \in \mathcal{K}} \alpha_{km}^t \leq 1, \forall k \in \mathcal{K}, m \in \mathcal{M}, t \in \mathcal{T} \tag{3.7}$$

$$\alpha_{km}^t \in \{0, 1\}, \forall k \in \mathcal{K}, m \in \mathcal{M}, t \in \mathcal{T} \tag{3.8}$$

Denote $\mathcal{M}_k^t \triangleq \{m | m \in \mathcal{M}, \alpha_{km}^t = 1\}$ as the set of ground users served by UAV k in time slot t.

We further consider FDMA-based spectrum access among the UAVs in $\mathcal{K}$ and TDMA for the ground users served by the same UAV. Then, the SINR of ground user $m \in \mathcal{M}$ in time slot t, denoted as $\gamma_m^t = \gamma_m^t(H_{m\mathrm{Tx}}^t)$ can be expressed as

$$\gamma_m^t = \frac{p_{k(m)}^t \cdot (h_{k(m)m}^t)^2 / H_{k(m)m}^t}{(p_{\mathrm{Tx}}/|\mathcal{K}|) \cdot w_{m\mathrm{Tx}}^t \hat{w}_m \cdot (h_{m\mathrm{Tx}}^t)^2 / (H_{m\mathrm{Tx}}^t) + \sigma_m^2}, \tag{3.9}$$

where $k(m)$ and $\hat{w}_m$ represent the serving UAV and receive gain of ground user m, respectively; $|\mathcal{K}|$ denotes the number of UAVs in $\mathcal{K}$; $H_{k(m)m}^t \in \{H_{k(m)m}^{\mathrm{NLoS},t}, H_{k(m)m}^{\mathrm{LoS},t}\}$ is the path-loss from UAV $k(m)$ to ground user m in time slot t with $H_{k(m)m}^{\mathrm{NLoS},t}$ and $H_{k(m)m}^{\mathrm{LoS},t}$ defined in Chapter 3.1.1; and σ_m^2 is the power of the AWGN noise at ground user m. Notice in (3.9) that only $\frac{1}{|\mathcal{K}|}$ of the incumbent transmitter's power (i.e., $p_{\mathrm{Tx}}/|\mathcal{K}|$) is considered for each UAV

and its associated ground users because of the UAVs' FDMA-based spectrum access. It is worth pointing out that we consider FDMA- and TDMA-based spectrum access for the UAV networks because we want to focus this work on the interference control between the UAV and the incumbent systems. The resulting cross-system spectrum sharing scheme can also be extended to other more advanced spectrum access schemes for UAVs [77, 78].

Finally, the capacity achievable by user m in time slot t, denoted as $C^t_{k(m)m}$, can be expressed as

$$\begin{aligned} C^t_{k(m)m} =& \frac{B}{|\mathcal{K}||\mathcal{M}^t_k|}\big[\Pr\left(H^{\mathrm{NLoS},t}_{k(m)m}\right)\log_2\left(1+\gamma^t_m(H^{\mathrm{NLoS}}_{k(m)m})\right) \\ &+\Pr\left(H^{\mathrm{LoS},t}_{k(m)m}\right)\log_2\left(1+\gamma^t_m(H^{\mathrm{LoS}}_{k(m)m})\right)\big], \end{aligned} \tag{3.10}$$

where $\Pr(\cdot)$ is the probability of LoS and NLoS transmissions defined in Chapter 3.1.1 and $\gamma^t_m(\cdot)$ is the SINR of ground user m defined in (3.9).

3.1.4 Problem Formulation

Define $\mathbf{P} = (p^t_k)^{t\in\mathcal{T}}_{k\in\mathcal{K}}$ as the transmission power vector of the UAVs, $\mathbf{A} = (\alpha^t_{km})^{t\in\mathcal{T}}_{k\in\mathcal{K},m\in\mathcal{M}}$ as the UAV-user association vector, and $\mathbf{Q} = (\mathbf{cod}^t_k)^{t\in\mathcal{T}}_{k\in\mathcal{K}}$ as the UAV location vector. Then the objective of the *SwarmShare* control problem is to maximize the aggregate capacity of the UAV network by jointly controlling the transmission power of the UAVs and their flight trajectory as well as association with the ground users, while meeting the cross-system

interference constraints, as formulated as

$$\underset{\mathbf{P},\mathbf{A},\mathbf{Q}}{Maximize} \quad \frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}}\sum_{m\in\mathcal{M}} C^t_{k(m)m} \tag{3.11}$$

$$\text{Subject to}: \ 0 \le p^t_k \le p_{\max}, \forall k \in \mathcal{K}, t \in \mathcal{T}, \tag{3.12}$$

$$Association\ Constraints \ \ (3.7),\ (3.8) \tag{3.13}$$

$$\underbrace{\frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}} \mathrm{I}(\gamma^t_{\mathrm{Rx}} \le \gamma^{\mathrm{th}}_{\mathrm{Rx}}) \le \mathrm{Pr}^{\max}_{\mathrm{Rx}}}_{Cross-system\ Interference\ Constraint} \tag{3.14}$$

where $C^t_{k(m)m}$ is defined in (3.10), $p_{\max}$ is the maximum transmission power of each UAV, $\mathrm{I}(\cdot)$ is the indication function taking value of 1 if the condition holds and 0 otherwise, and $\gamma^{\mathrm{th}}_{\mathrm{Rx}}$ and $\mathrm{Pr}^{\max}_{\mathrm{Rx}}$ denote respectively the threshold SINR and the maximum tolerable SINR outage probability of the incumbent system.

3.2 Spectrum Coexistence Design

The *SwarmShare* problem formulated in (3.11)-(3.14) is a mixed integer nonlinear non-convex programming (MINLP) problem, because of the binary UAV-user association variables α^t_{km} and the underlying complicated mathematical expressions in (3.11) and (3.14).

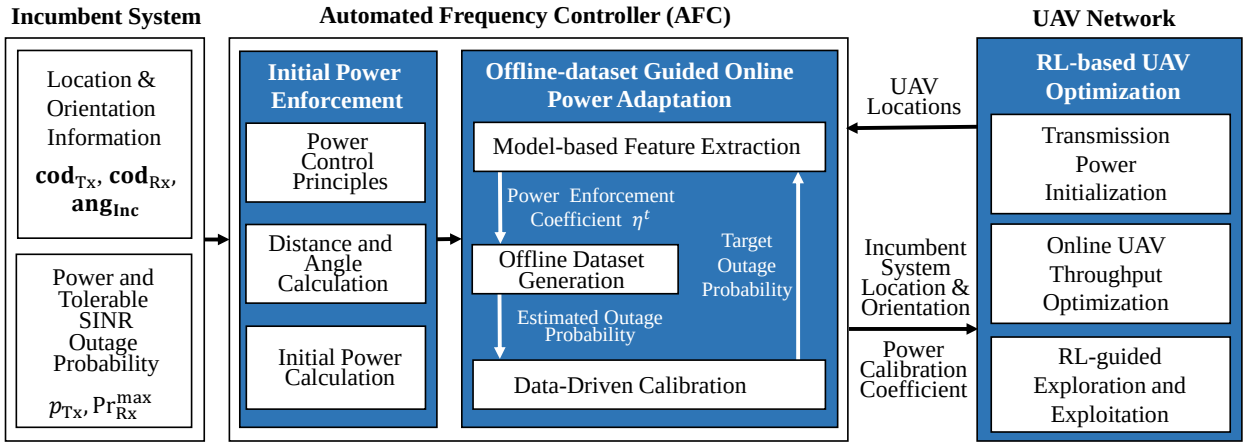


Figure 3.2: Diagram of the *SwarmShare* spectrum sharing framework.

Moreover, to solve the problem directly it requires knowing the real-time channel state information (CSI) between the UAV network and the incumbent system, which is unavailable, as discussed in Chapter 1, because of the low-detectability of the directional incumbent signals.

To address the above challenges, in this work we consider an AFC (Automated Frequency Controller)-assisted spectrum sharing. AFC has been adopted for spectrum sharing in the TV whitespace band as well as the 6 GHz band by determining certain exclusion zones nearby the incumbent systems [13]. Our work differs from this with our objective to enable exclusion-zone-free hence more flexible spectrum sharing, and study the statistical behavior of the aggregate interference from the UAV networks to the incumbent system, while keeping the cross-system signaling at a minimum level. The diagram of the proposed spectrum coexistence framework is illustrated in Figure 3.2, where there are three major components, i.e., *Initial Power Enforcement*, *Offline-dataset Guided Online Power Adaptation*, and *Reinforcement Learning-based UAV Optimization*.

3.2.1 Initial Power Enforcement

The objective of this phase is to determine, following a set of *Power Control Principles*, a rough transmission power for each of the UAVs. In this work, we consider three basic principles to accommodate the effects of the UAVs' flight altitudes and their locations on the interference to the incumbent system, while more sophisticated principles can be incorporated in the future. These principles are i) UAVs that are closer to the incumbent receiver should transmit at lower power; (ii) with the same distance to the incumbent receiver, UAVs flying higher should transmit at lower power; and (iii) with the same distance and altitude, UAVs with smaller angles relative to the boresight axis of the incumbent receivers' directional antenna should transmit at lower power. Particularly, the rationale of the second principle is that, with the hybrid LoS/NLoS channel model described in Chapter 3.1.1, it is more likely for a UAV to establish LoS links to the incumbent receiver when flying higher and hence cause more interference. Similarly, for the third principle, based on the directional

antenna model described in Chapter 3.1.2, a UAV will cause higher interference when it is more aligned with the incumbent receiver's antenna.

In *SwarmShare*, an initial power enforcement coefficient, $\text{Enf}(\mathbf{cod}_k^t, \mathbf{cod}_{\text{Rx}}, \mathbf{ang}_{\text{Inc}})$, will be calculated for each UAV $k \in \mathcal{K}$ in time slot $t \in \mathcal{T}$ based on the above three principles. This is accomplished using three Sigmoid-family functions $Sig_1(\cdot)$, $Sig_2(\cdot)$ and $Sig_3(\cdot)$, as follows:

$$\text{Enf}(\mathbf{cod}_k^t, \mathbf{cod}_{\text{Rx}}, \mathbf{ang}_{\text{Inc}}) = \underbrace{Sig_1\left(\frac{l_{\text{euc}}(\mathbf{cod}_k^t, \mathbf{cod}_{\text{Rx}})}{l_{\text{euc}}^{\text{th}}}\right)}_{Principle\ 1} \cdot \underbrace{Sig_2\left(\frac{h(\mathbf{cod}_k^t) + h(\mathbf{cod}_{\text{Rx}})}{l_{\text{hgh}}^{\text{th}}}\right)}_{Principle\ 2} \cdot \underbrace{Sig_3\left(l_{\text{rad}}(\mathbf{cod}_k^t, \mathbf{cod}_{\text{Rx}}, \mathbf{ang}_{\text{Inc}})\right)}_{Principle\ 3}, \tag{3.15}$$

where $l_{\text{euc}}(\cdot,\cdot)$ represents the Euclidean distance between UAV k and the incumbent receiver given their coordinates; $h(\cdot)$ represents their height, and $l_{\text{rad}}(\cdot,\cdot,\cdot) \in [0, \pi]$ is the angle (in radians) of UAV k with respect to the boresight axis of the incumbent receiver antenna; finally, $l_{\text{euc}}^{\text{th}}$ and l_{hgh}^{th} in equation (3.15) are respectively, the threshold distance and height beyond which $Sig_1(\cdot)$ and $Sig_2(\cdot)$ become nearly constant. It is worth pointing out that, since a standard sigmoid function is a differentiable, monotonically increasing, real function taking values in $[0, 1]$, we design $Sig_1(\cdot)$, $Sig_2(\cdot)$ and $Sig_3(\cdot)$ by scaling, shifting and reversing the standard sigmoid function to consider the effects of the UAV location, flight altitude and relative angle to the incumbent receiver. For example, $Sig_1(x) = \frac{1}{1+e^{-3(x/70-2)}}$ has been adopted for principle 1 in this work, while $Sig_2(\cdot)$ and $Sig_3(\cdot)$ can be defined similarly. With the obtained power enforcement coefficient $\text{Enf}(\mathbf{cod}_k^t, \mathbf{cod}_{\text{Rx}}, \mathbf{ang}_{\text{Inc}})$, each UAV's power can be initialized as, in time slot $t \in \mathcal{T}$,

$$p_k^{\text{ini}} = p^{\max}\text{Enf}(\mathbf{cod}_k^t, \mathbf{cod}_{\text{Rx}}, \mathbf{ang}_{\text{Inc}}),\ \forall k \in \mathcal{K}, \tag{3.16}$$

where $p^{\max}$ is the maximum transmission power of each UAV.

3.2.2 Offline-dataset Guided Online Power Adaptation

Recall in Chapter 3.1 that our goal is to enable UAV operations in the 6 GHz band while meeting the cross-system interference constraint (3.14). In *SwarmShare*, this is accomplished by fine-tuning the above obtained initial transmission power for the UAVs according to a three-step approach, as described as follows.

Model-based Feature Extraction

In this step, we first extract the network features that can be used later in *Data-Driven Calibration*, rather than using directly, the raw network topology information such as UAV location vector $(\mathbf{cod}_k^t)_{k\in\mathcal{K}}^{t\in\mathcal{T}}$. It is important to mitigate the curse of dimensionality problem [79] especially with a large number of UAVs. In *SwarmShare*, we select the power adaptation coefficient, denoted as η^t for time slot t, as the network feature. Then, given the above obtained initial transmission power p_k^{ini} for UAV $k \in \mathcal{K}$, a new transmission power p_k^t can be obtained as

$$p_k^t = \begin{cases} 0, & \text{if } \alpha_{km}^t = 0, \forall m \in \mathcal{M} \\ p_k^{\text{ini}}\eta^t, & \text{otherwise} \end{cases}. \tag{3.17}$$

That is, the transmission power will be set to 0 if a UAV is not associated to any ground users. Then the interference constraint (3.14) can be rewritten as

$$\frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}} \mathrm{I}(\gamma_{\text{Rx}}^t(\eta^t) \le \gamma_{\text{Rx}}^{\text{th}}) \le \mathrm{Pr}_{\text{Rx}}^{\max}, \tag{3.18}$$

where $\gamma_{\text{Rx}}^t(\eta^t)$ is the SINR of the incumbent receiver defined in (3.6) by substituting (3.17) into (3.6). Consider an ergodic stochastic process for the aggregate interference and denote $\text{Prob}\big(\gamma_{\text{Rx}}^t(\eta^t) \le \gamma_{\text{Rx}}^{\text{th}}\big)$ as the SINR outage probability in time slot $t \in \mathcal{T}$, then the left-hand

side of (3.18) can be equivalently represented as

$$\mathrm{Prob}\big(\gamma_{\mathrm{Rx}}^{t}(\eta^{t}) \leq \gamma_{\mathrm{Rx}}^{\mathrm{th}}\big) \tag{3.19}$$

$$= \mathrm{Prob}\left(\frac{P_{\mathrm{Rx}}^{\mathrm{sig}}}{P_{\mathrm{Rx}}^{\mathrm{itf}}} \leq \gamma_{\mathrm{Rx}}^{\mathrm{th}}\right) \tag{3.20}$$

$$= \int_{0}^{+\infty} \int_{\frac{P_{\mathrm{Rx}}^{\mathrm{sig}}}{\gamma_{\mathrm{Rx}}^{\mathrm{th}}}}^{+\infty} \underbrace{\mathrm{pdf}_{P_{\mathrm{Rx}}^{\mathrm{sig}}}(p^{\mathrm{sig}})}_{\substack{Noncentral\\ Chi-square\\ Distribution}} \cdot \underbrace{\mathrm{pdf}_{P_{\mathrm{Rx}}^{\mathrm{itf}}}(p^{\mathrm{itf}})}_{\substack{Gamma\\ Distribution}} \mathrm{d}p^{\mathrm{itf}} \mathrm{d}p^{\mathrm{sig}}, \tag{3.21}$$

where $P_{\mathrm{Rx}}^{\mathrm{sig}}$ and $P_{\mathrm{Rx}}^{\mathrm{itf}}$ are the numerator and denominator of (3.6), respectively; a Rayleigh distribution has been considered for the small-scale fading and hence noncentral chi-square distribution [80] for the receive power of the incumbent signals; and finally as in [81, 82] a Gamma distribution is considered for the aggregate interference power. It is worth pointing out that, as shown later in Chapter 3.3, the aggregate interference of UAVs can't be easily characterized using any existing statistical distributions. In our research, we consider the Gamma distribution in (3.21) because we want to obtain a rough estimation of the power adaptation coefficient η^{t}, which will be further calibrated based on an offline dataset. Notice that given the maximum tolerable SINR outage probability $\mathrm{Pr}_{\mathrm{Rx}}^{\max}$ in (3.18), the maximum η^{t} can be determined efficiently by bisection search, since the left-hand side of (3.18), which is equivalent to (3.21), is a monotonically increasing function of the UAVs' transmission power hence η^{t}.

Offline-dataset Generation

Given the above obtained network feature η^{t}, each UAV's transmission power p_{k}^{t} can be updated according to (3.17). Since the power adaptation may be inaccurate because of the inaccuracy of the Gamma distribution-based interference model in (3.21), we further calibrate the power control for UAVs with the assistance of offline measurements. Specifically, given

the transmission power vector $(p_k^t)_{k\in\mathcal{K}}$, the corresponding SINR outage probability of the incumbent system can be obtained by offline simulations. By varying the number of UAVs, their locations, as well as the maximum tolerable SINR outage probability in the simulations, we are able to obtain an SINR outage probability vector. Denote $\mathbf{Pr}_{\mathrm{Rx}}^{\max} = (\mathrm{Pr}_{\mathrm{Rx}}^{\max})$ as the vector of the maximum tolerable outage probability, and accordingly denote the simulated outage probability vector as $\overline{\mathbf{Pr}}_{\mathrm{Rx}}^{\max}(\boldsymbol{\eta}) = (\overline{\mathrm{Pr}}_{\mathrm{Rx}}^{\max}(\eta^t))$ with $\overline{\mathrm{Pr}}_{\mathrm{Rx}}^{\max}(\eta^t)$ being the SINR outage probability given the network metric η^t, and $\boldsymbol{\eta} = (\eta^t)$ the network feature vector.

Data-Driven Calibration

Finally, a mapping between $\mathbf{Pr}_{\mathrm{Rx}}^{\max}$ and $\overline{\mathbf{Pr}}_{\mathrm{Rx}}^{\max}(\boldsymbol{\eta})$ can be established through function approximation, e.g., based on linear regression [83], echo state learning [4] or deep neural networks [84, 85]. In this work we find that it is enough to approximate the mapping based on linear regression. Denote the mapping as $\mathrm{Pr}_{\mathrm{Rx}}^{\max} = f(\overline{\mathrm{Pr}}_{\mathrm{Rx}}^{\max}(\eta^t)$. Then, given $\overline{\mathrm{Pr}}_{\mathrm{Rx}}^{\max}$, the value of $\mathrm{Pr}_{\mathrm{Rx}}^{\max}$ and the corresponding network feature η^t can be obtained at network run time and further used for UAV power control based on (3.17).

3.2.3 Reinforcement Learning-based UAV Optimization

As illustrated in Figure 3.2, the above obtained η^t will be broadcast to the UAVs, which will then calculate their transmission power based on (3.17). Meanwhile, the UAVs will update their flight and association strategies to serve their users with higher spectral efficiency. To this end, we consider as in [76] the shortest-distance-based association strategy. Then, in each time slot $t \in \mathcal{T}$ the association variables $\alpha_{k,m}$ defined in Chapter 3.1 can be determined as

$$\alpha_{k,m} = \begin{cases} 1, & \text{if } k = \arg\min_k \|\mathbf{cod}_k^t - \mathbf{cod}_m\|^2 \\ 0, & \text{otherwise} \end{cases} . \tag{3.22}$$

Finally, as in [86], reinforcement learning with the ϵ-greedy search is adopted to guide the exploitation and exploration during the UAV's flight control. To this end, we further divide the whole network area into a set of three dimensional rectangles, each corresponding to a state of the RL control problem. Denote $\mathcal{S}$ as the set of all possible states. In each time slot, each UAV is allowed to either move to one of its adjacent rectangles or stay in the current. For each of the candidate rectangles, the UAV will first calculate the achievable capacity given the transmission power calculated in Chapter 3.2.1 and 3.2.2 and the set of ground users it serves.

Let $a \in \mathcal{A}$ be the action of the considered UAV, where $\mathcal{A} = \{F, B, U, D, R, L, S\}$ is the set of candidate actions, including moving forward, backward, upper, lower, right and left and staying at the current location. Then the action A_k^t chosen by UAV k at time slot t can be given by, for state $s \in \mathcal{S}$,

$$A_k^t = \begin{cases} \arg\max\limits_{a \in \mathcal{A}} Q_k^t(a, s), & \text{with probability } 1 - \epsilon \\ a' \in \mathcal{A}/a, & \text{with probability } \frac{\epsilon}{|\mathcal{A}/a|} \end{cases}, \tag{3.23}$$

where $Q_k^t(a, s)$ is the estimated throughput value of UAV k if action a is chosen at state s in time t, and $|\cdot|$ represents the cardinality of a set. For each state, the state-action value (i.e., $Q_k^t(a, s)$) is estimated using a table-based approach with discount factor 0 [87]. Based on the selected action A_k^t at time slot t, coordinate vector $\mathbf{cod}_k^{t+1} = (x_k^{t+1}, y_k^{t+1}, z_k^{t+1})$ of UAV k for time slot $t+1$ can then be updated accordingly. The overall spectrum sharing algorithm is summarized in Algorithm 1.

3.2.4 Complexity Analysis

The most time-consuming operation of Algorithm 1 is offline-dataset guided online power adaptation in Chapter 3.2.2. In Chapter 3.2.2, we need to determine the network feature η^t given the UAV network's topology and the maximum tolerable SINR outage probability.

Algorithm 1: *SwarmShare* Algorithm

Data: Node coordinates $\mathbf{cod}_i = (x_i, y_i, z_i); \forall i \in \mathcal{A}$, incumbent Tx transmission power p_{Tx}, tolerable SINR outage probability $\mathrm{Pr}_{\mathrm{Rx}}^{\max}$, total duration of simulation $\mathcal{T}$

Result: For each UAV $k \in \mathcal{K}$, the transmission power p_k^t for time slot $t \in \mathcal{T}$ and the coordinates $\mathbf{cod}_k^{t+1}$ for time slot $t+1 \in \mathcal{T}$

while $t \in \mathcal{T}$ **do**
  Initial Power Enforcement
  for *each UAV* $k \in \mathcal{K}$ **do**
    Calculate initial power enforcement coefficient $\mathrm{Enf}(\cdot)$ based on (3.15); Calculate initial power p_k^{ini} based on (3.16);
  end
  Offline-dataset Guided Online Power Adaptation
  Determine the network feature η^t based on (3.18)-(3.21);
  for *each UAV* $k \in \mathcal{K}$ **do**
    Determine the transmission power p_k^t based on (3.17);
  end
  Reinforcement Learning-based UAV Optimization
  for *each UAV* $k \in \mathcal{K}$ **do**
    Determine the association variables $\alpha_{k,m}$ based on (3.22);
    Determine A_k^t based on (3.23) and update $\mathbf{cod}_k^{t+1}$ for next time slot.
  end
end

Since the SINR outage probability is a monotonically increasing function of η^t, the value of η^t can be obtained efficiently based on bisection search with a constant computational complexity for fixed range of η^t (i.e., [0, 1]) and the margin of error (1% in this work). It is worth pointing out that, for each iteration of bisection search, the AFC needs to calculate the mean and variance of the aggregate interference of UAVs given their current locations and their assigned transmission power based on (3.17). However, the resulting computational complexity is only negligible since it can be done by simple linear operations.

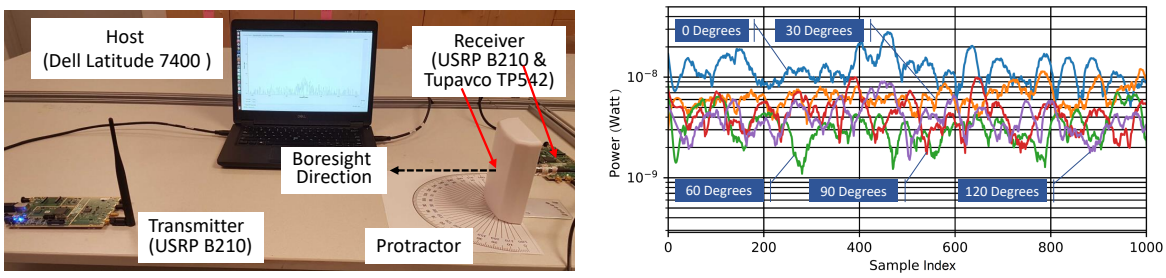
Finally, the dataset generation and regression in Chapter 3.2.2 can be conducted offline, and in Chapter 3.2.2 $\mathrm{Pr}_{\mathrm{Rx}}^{\max}$ can be determined online by solving a linear problem.

Regarding communication overhead, as illustrated in Figure 3.2, in Chapter 3.2.1 the AFC needs to collect one-time location and orientation information of the incumbent system and broadcast the collected information to the UAVs. If the incumbent system does not move frequently (which is usually the case, e.g., fixed point-to-point applications), the resulting

communication overhead can be neglected. The AFC also needs to collect periodically the UAVs' locations and broadcast the updated power adaptation coefficient η^t to the UAVs. Since it is enough to represent this information in 16 bytes (three float numbers for location and one for the power adaptation coefficient, and each float number takes 4 bytes), the resulting broadcast overhead is low as well. Moreover, we will show later in Chapter 3.3 that the UAVs do not need to report their locations to the AFC in real-time, without obviously increasing the SINR outage probability of the incumbent system. This will further reduce the communication overhead.

3.3 Performance Evaluation

In this chapter we validate the effectiveness of the *SwarmShare* framework described in Chapters 3.1 and 3.2. We consider a network area of $500 \times 500 \times 50$ m^3, with 50 ground users randomly located in the network and the number of UAVs varying from 3 to 24. The incumbent transmitter and receiver are deployed with coordinates of (200, 200, 10) and (250, 250, 10), respectively. The center frequency of the shared spectrum is set to 6 GHz with a total bandwidth of 10 MHz. The maximum transmission power of the incumbent transmitter and the UAVs are set to 1 W and 0.25 W, respectively. For the bisectorized antenna model desribed in Chapter 3.1.2, the maximum and minimum gains are set to 1 and 0.5, respectively. The power density of the AWGN is set to -174 dBm/Hz. The probability of LoS and NLoS links are set to 0.7 and 0.3, respectively. The threshold parameters $l_{\text{euc}}^{\text{th}}$ and $l_{\text{hgh}}^{\text{th}}$ in (3.15) are set to 70 m and 30 m, respectively. Next, before discussing the interference control results, we first determine the threshold angle for the directional antenna model described in Chapter 3.1.2 and validate the effectiveness of the data-driven calibration scheme proposed in Chapter 3.2.2.



(a) (b)

Figure 3.3: (a) Snapshot of the testbed setup for threshold angle measurement; (b) Examples of the measurement results.

3.3.1 Threshold Angle Measurement

We first determine the threshold angle for the directional antenna model described in Chapter 3.1.2 by conducting a set of experimental measurements. A snapshot of the testbed is shown in Figure 3.3(a), where the transmitter is a USRP B210 software radio with an omnidirectional antenna, the receiver is another USRP B210 with a Tupavco TP542 antenna, and the baseband signal processing is conducted based on GNU Radio on a Dell Latitude 7400 laptop. Tupavco TP542 is a directional Wi-Fi antenna operating in a frequency range up to 5.8 GHz (very close to the 6 GHz band) with an antenna gain of 13 dBi. We measure the received power by varying the relative angle of the transmitter with respect to the boresight direction of the directional antenna (as illustrated in Figure 3.3(a)) and the transmission distance from 1 to 3 meters. Examples of the measurement results are given in Figure 3.3(b) with a transmission range of 1 meter and relative angles varying from 0 to 120 degrees at step of 30 degrees. The mapping between the received power and relative angle is established based on the logarithmic regression method [88]. Based on the regression results, we set 30 degree as the threshold angle for the bisectorized antenna model Chapter 3.1.2, which corresponds to the 3 dB angle of the Tupavco TP542 antenna.

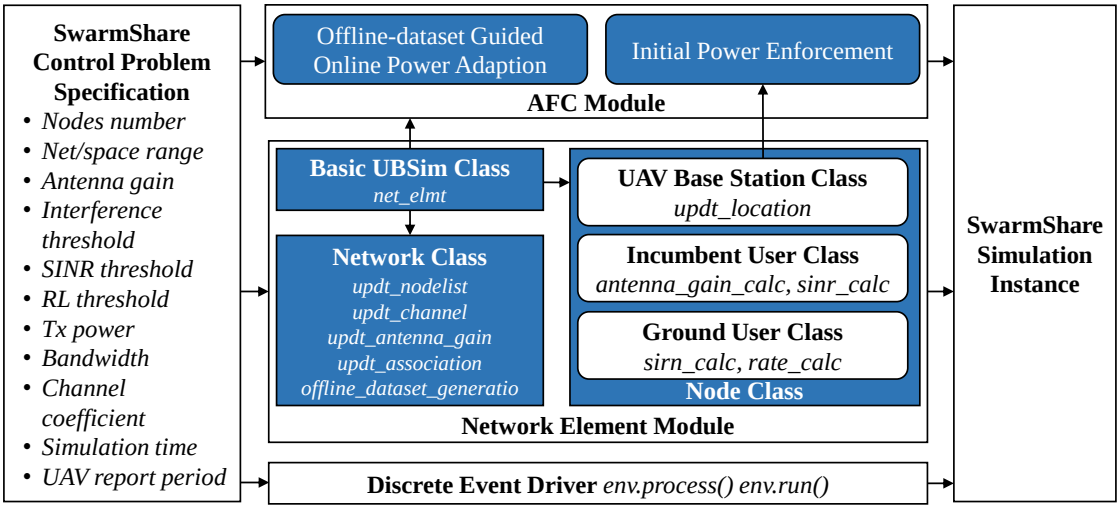
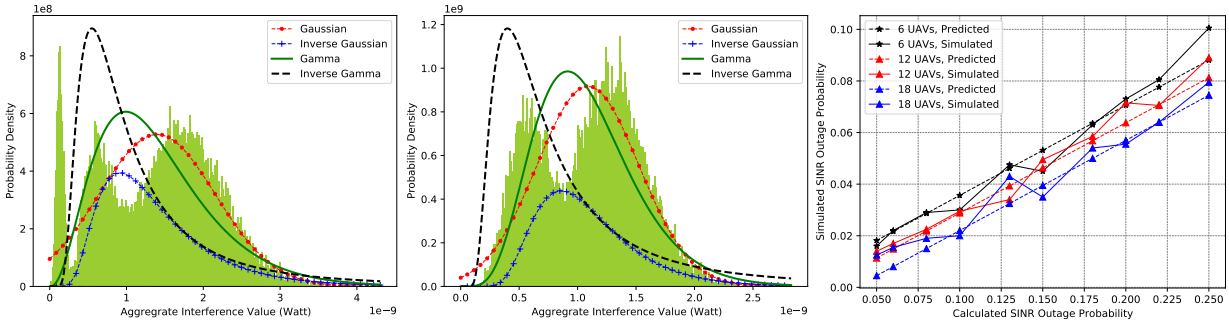


Figure 3.4: Diagram of UBSim-based *SwarmShare* Simulator.

3.3.2 Data-driven Interference Prediction

Given the above obtained threshold angle, we further characterize the statistical behavior of the aggregate interference from the coexisting UAVs to the incumbent receiver. To this end, we conduct a set of simulations over UBSim[1]. As shown in Figure 3.4, the simulator consists of four major modules: *SwarmShare Control Problem Specification Module, AFC Module, Network Element Module* and *Discrete Event Driver*. The *SwarmShare* Control Problem Specification Module provides a set of APIs, based on which experimenters can define various network parameters such as the size of network area, the number of incumbent users, UAVs and ground users, antenna gain, probability channel coefficient, interference threshold, spectrum bandwidth, among others. Then, an object will be created for each of the incumbent users, UAV base stations and ground users in the *Network Element Module*. The classes of these network elements are defined in a hierarchical manner based on the basic network element class *net_elmt*, which defines the most basic network element attributes and operations. The *AFC Module* is the place where the spectrum coordination algorithms designed in Chapter 4.2 are deployed and executed. Finally, the *Discrete Event Driver* is

[1]In this work, the performance evaluation is conducted primarily over UBSim. We measure the threshold angle based on testbed experiments because in future work we want to further test the proposed spectrum coexistence framework in real world considering the directional antenna, Tupavco TP542, for incumbent users.



(a) (b) (c)

Figure 3.5: Aggregate interference pdf with (a) 10 and (b) 20 UAVs; (c) Validation of data-driven prediction of the SINR outage probability for the incumbent system.

developed based on the open-source library SimPy [89] for discrete event-driven simulations.

The results are reported in Figures 3.5(a) and (b) with 10 and 20 UAVs, respectively. We fit as in [90,91] the collected interference values using four distributions, including Gaussian, Inverse Gaussian, Gamma and Inverse Gamma, and find that the power of the aggregate interference does not follow any of these distributions. This is actually our motivation to design *SwarmShare* based on a data-driven approach. Figure 3.5(c) reports the results of the data-driven prediction of the SINR outage probability. The offline dataset is generated based on simulations. We consider 6, 12 and 18 UAVs with multiple possible violation probabilities ranging from 0.05 to 0.25 with interval of 0.025. For each combination of UAV number and violation probability, we conduct 2000 episodes simulations with 2000 time slots in each episode. We can find that the predicted SINR outage probability matches the simulated results well.

3.3.3 Case Study

Figure 3.6 shows an example of the power control results based on *SwarmShare*. To visualize the effects of the power control principles described in Chapter 3.2.1, in this example all the 24 UAVs are deployed uniformally along 4 circles with different altitudes and radii. From the figure, it can be seen that lower transmission powers have been allocated to UAVs

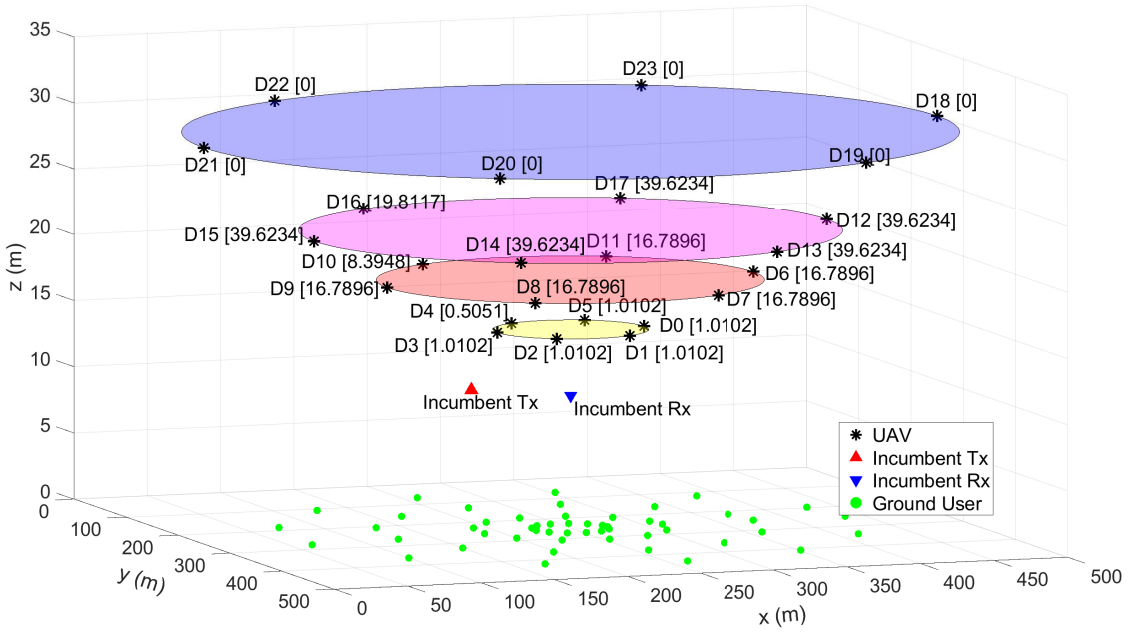
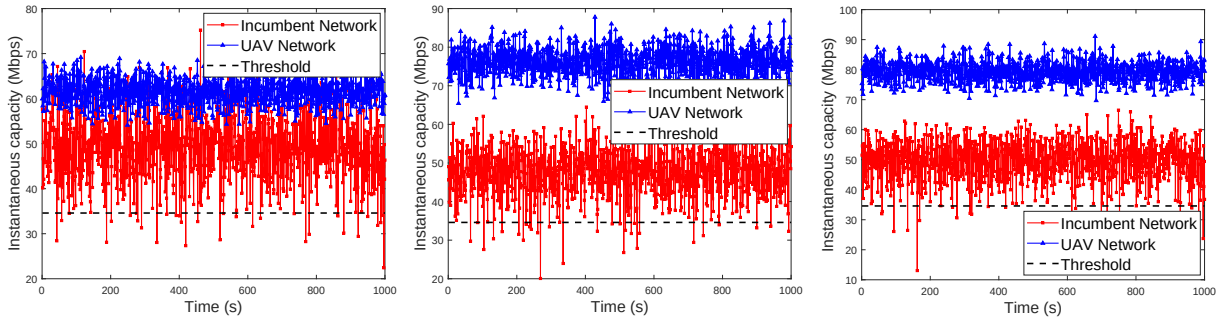


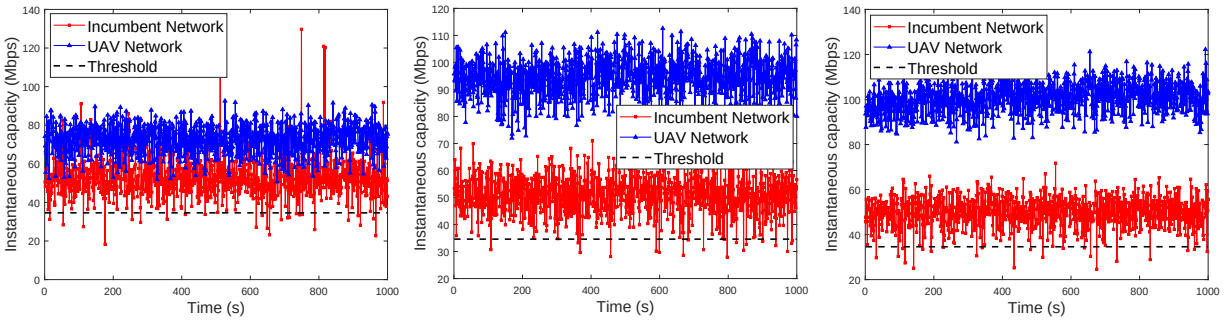
Figure 3.6: Case study of power control based on *SwarmShare*. D#1[#2]: #1 is the UAV index, and #2 denotes the transmission power of the UAV in mW.

along the lower circles. Also, because of the shorter distances from the incumbent receiver, lower transmission power has been allocated to the UAVs of the first circle from the bottom, e.g., 1.0102 mW for UAV 1 (i.e., D1[1.0102] in Figure 3.6) against 16.7896 mW for UAV 7 and 39.6234 mW for UAV 13 along the second and third circles, respectively. Moreover, along the same circle, UAVs more aligned with the incumbent receiver have been allocated lower transmission powers, e.g., 8.3948 mW for UAV 10 vs 39.6234 mW for UAV 9 along the second circle. Finally, we notice in this example that all the UAVs along the fourth (i.e., the highest) circle have been allocated zero transmission power because no users are associated with them based on the shortest-distance association strategy described in Chapter 3.2.3. This also conforms to the third power control principle, i.e., with the same distance and relative angle, higher altitudes result in lower transmission powers because of higher probability of LoS transmissions. It is worth pointing out that the power allocation results are determined by jointly considering the three basic principles described in Chapter 3.2.1. In the following experiments, we will further evaluate the effectiveness of *SwarmShare* on the cross-system interference control.



(a) 6 UAVs, Hovering (b) 12 UAVs, Hovering (c) 18 UAVs, Hovering

Figure 3.7: Instantaneous capacity of the incumbent system and the UAV network with hovering UAVs. The violation probabilities are (a) 0.032, (b) 0.029 and (c) 0.021, respectively.



(a) 6 UAVs, Moving (b) 12 UAVs, Moving (c) 18 UAVs, Moving

Figure 3.8: Instantaneous capacity of the incumbent system and the UAV network with moving UAVs. The violation probabilities are (a) 0.025, (b) 0.018 and (c) 0.022, respectively.

In Figures 3.7 and 3.8, we plot the instantaneous capacity achievable by the incumbent system and the UAV networks with different numbers of UAVs. In Figure 3.7(a), we consider 6 hovering UAVs, and the maximum tolerable SINR outage probability is set to 0.05 for the incumbent system. The achievable capacity is plotted for 1000 time slots. Results indicate that the interference constraint of the incumbent system can be very well fulfilled, with SINR outage probability of 0.032. Similar results can be obtained with 12 and 18 hovering UAVs in Figures 3.7(b) and (c), with the SINR outage probability of 0.029 and 0.021, respectively.

Figure 3.8 shows the corresponding results with moving UAVs. In this experiment, the network area is divided into a set of three-dimension rectangles each of $50 \times 50 \times 10$ m^3. The trajectory of the UAVs are controlled as described in Chapter 3.2.3, with exploitation proba-

bility of 0.98. The same as in Figure 3.7, the incumbent system's interference constraints can be satisfied in all the tested cases, with SINR outage probabilities of 0.025, 0.018 and 0.022 for 6, 12 and 18 UAVs, respectively, all below the maximum tolerable outage probability 0.05. This verifies the effectiveness of *SwarmShare* in cross-system interference control. It can also be seen that the incumbent network's capacity does not decrease obviously as the number of UAVs increases (e.g., from 12 to 18). This is because, as more UAVs are deployed in a network with fixed number of ground users, some UAVs will not serve any ground users and hence become inactive and cause no interference to the incumbent system.

3.3.4 Average Results

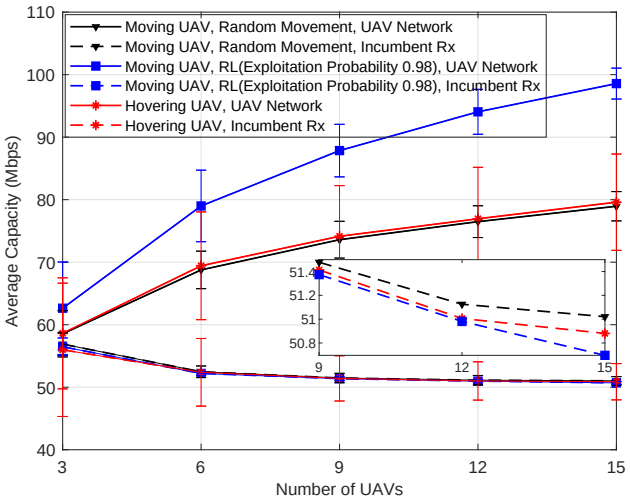


Figure 3.9: Average capacity of the incumbent system and UAV networks with moving UAVs.

In Figure 3.9 we report the average capacity achievable by the incumbent system and the UAV network with the number of UAVs varying from 3 to 15 at step of 3. Three UAV mobility patterns are considered: i) random movement; ii) reinforcement learning controlled movement with exploitation probability of 0.98; and iii) hovering UAVs. The results are obtained by averaging over 50000 time slots for each mobility pattern. It can be seen that, as expected, obvious capacity gain can be achieved by the UAV network with all the above three mobility patterns by deploying more UAVs. For example, for hovering UAVs, the

average capacity increases from around 60 Mbps with 3 UAVs to 80 Mbps with 15 UAVs. The capacity is further increased to around 100 Mbps with RL-controlled UAV movement. Particularly, we find that there is no obvious degradation in the capacity of the incumbent system when there are 6 or more coexisting UAVs. The average capacity of the incumbent system can be further increased with less UAVs, e.g., 3, because of the reduced cross-system interference.

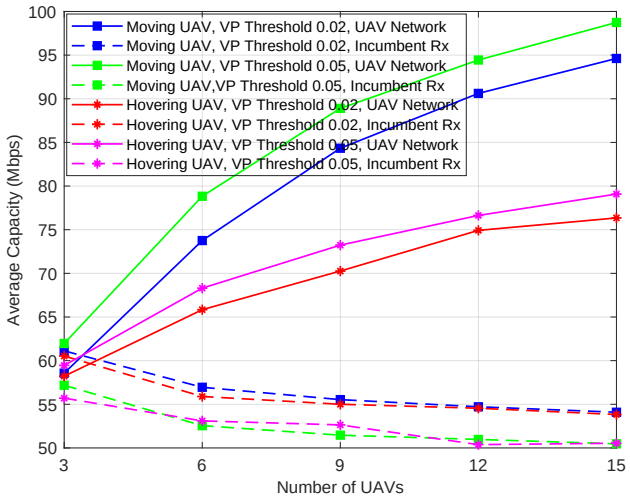


Figure 3.10: Average capacity of the incumbent system and UAV networks with different violation probability (VP) threshold.

In Figure 3.10, we report the average capacity achievable by the incumbent system and the UAVsunder different violation probability constraints. Two UAV mobility patterns are considered: i) hovering UAVs and ii) moving UAVs guided by reinforcement learning. The exploitation probability is set to 0.98 for the latter case. Similar to Figure 3.9, the aggregate capacity of the UAV networks can be increased significantly by deploying more UAVs. For example, 70 Mbps can be achieved with 9 hovering UAVs and violation probability threshold 0.02, which goes up to 76 Mbps with 15 UAVs. The corresponding incumbent user capacity are 55 Mbps and 54 Mbps. It can also be seen that significant capacity gain can be achieved by RL-guided UAV control. For example, 84 Mbps and 94 Mbps can be achieved with 9 and 15 UAVs, which are 1.52 and 1.74 times higher than that with hovering UAVs. Similar results can also be observed with a violation probability threshold 0.05.

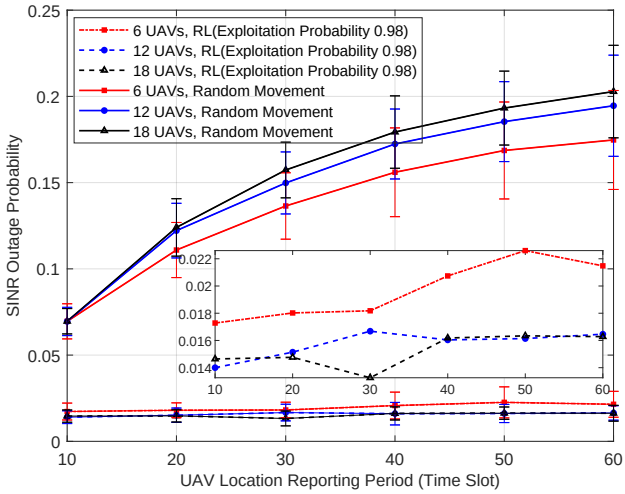
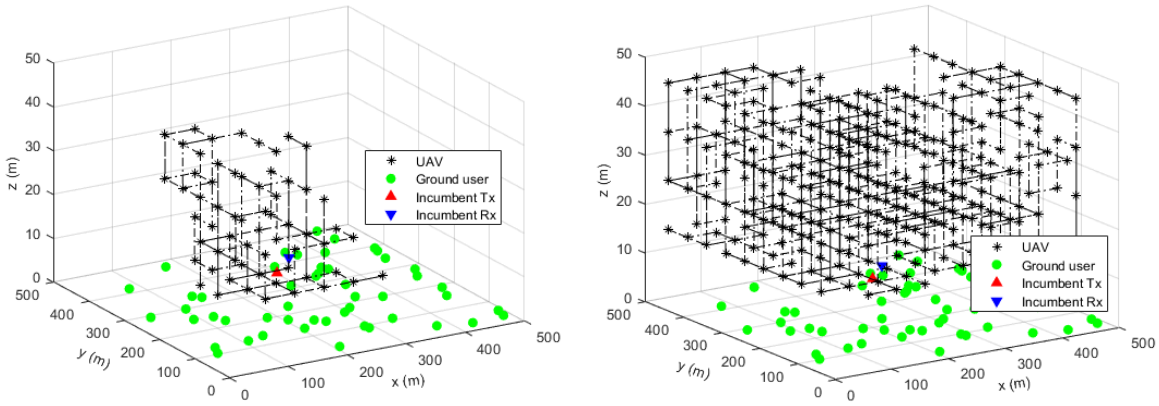


Figure 3.11: SINR outage probability vs UAV location reporting period.



(a) (b)

Figure 3.12: Example of UAV trajectory with location reporting period of 60 time slots. (a) RL-guided movement; (b) random movement.

In previous experiments (in Figure 3.7) the UAVs report their locations to the AFC in every time slot. In this experiment, we investigate the mobility resilience of *SwarmShare* for spectrum sharing in the presence of inaccurate UAV locations. The SINR outage probability results are reported in Figure 3.11, where two mobility patterns are considered for the UAVs, i.e., random movement and RL-guided movement, and the maximum tolerable SINR outage probability is set to 0.05 for the incumbent system. The location reporting period is varied from 10 time slots to 60. It can be seen that the SINR outage probability of the incumbent system increases monotonically with the location reporting period if the UAVs move in

an uncontrolled manner, i.e., completely randomly. For example, the outage probability is around 0.07 when the reporting period is 10 time slots and can be up to 0.2 for 60 time slots. In the case of controlled UAV movement, the SINR outage probability is barely affected by the location reporting period and always below the maximum tolerable. This is because, as illustrated by the example trajectories in Figure 3.12, the UAVs will stick with their current best locations at a high probability (0.98 in this experiment) while exploring new locations at a low probability (0.02). As a result, the topology of the UAV network and hence the statistical behavior of their aggregate interference to the incumbent system changes only slowly. Therefore, with controlled UAV movement, effective interference control can be achieved with *SwarmShare* in mobile scenarios even with inaccurate UAV locations, e.g., because of the temporary loss of the connections to the AFC.

3.3.5 Effects of UAV Moving Speed

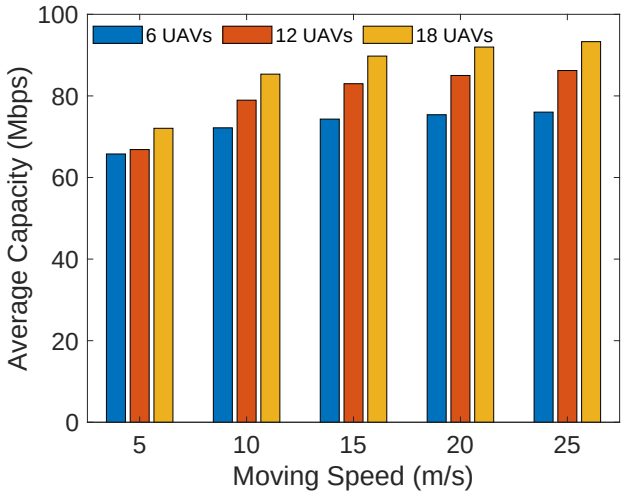
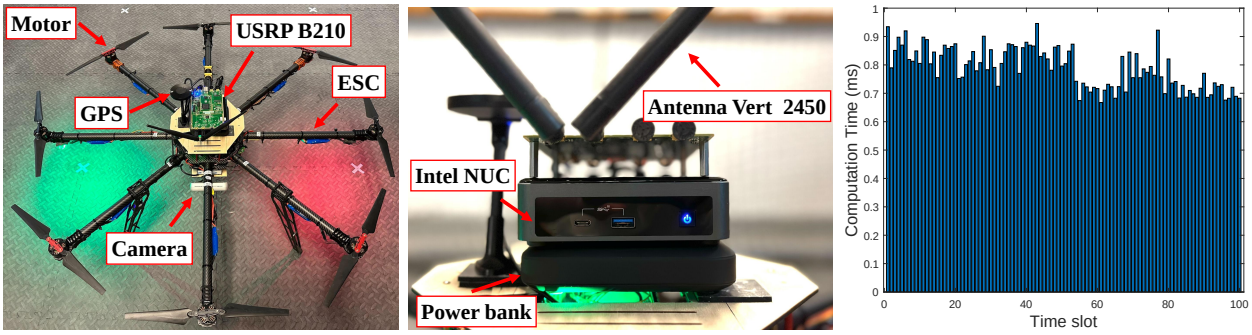


Figure 3.13: Average UAV network capacity vs. UAV moving speed.

In this experiment, we study the effects of the UAV moving speed on the throughput achievable by the UAV network. Consider 12, 15 and 18 UAVs in the network and the maximum moving speed of UAVs is set to 25 m/s. The duration of each time slot is set to 30 seconds. The results are reported in Figure 3.13. It can be seen that, as expected, a larger

average throughput can be achieved by the UAV network with higher moving speed. This is because we consider in each time slot that each UAV first moves to the new position before providing service to ground users, and hence with higher moving speed each UAV can arrive at the target position faster and start to serve the ground users sooner.

3.3.6 Computational Complexity



(a) (b) (c)

Figure 3.14: (a) Snapshot of octocopter UAV, (b) onboard computing device Intel NUC, and (c) computation time.

We further study the computational time of the UAVs. In this experiment, each UAV needs to finish two tasks in each time slot: transmission power initialization and RL-guided trajectory optimization. For the first task, each UAV calculates its initial power based on three Sigmoid functions defined in (3.15). In the second task, each UAV determines its next-step movement based on the RL algorithm described in Chapter 3.2.3.We conduct the experiments on Next Unit of Computing (NUC) with Intel® Core ™ i5-10210U CPU @ 1.60 GHz ×8, memory of 16 GB, and Ubuntu 20.04 Operating System. The dimension of NUC is $117 \times 112 \times 37$ mm^3 and with weight of 504 g. As shown in Figures 3.14(a) and (b), NUC has been integrated with the octocopter UAV custom-designed in our lab as the onboard computing device. The results are reported in Figure 3.14(c). It can be seen that it takes less than 1 ms for each UAV to finish the two tasks in each step. Since the UAV movement usually operates at a much larger time scale, this verifies the low computational complexity

$$\gamma_{\mathrm{Rx}_l}^t = \frac{p_{\mathrm{Tx}_l} w_{\mathrm{Tx}_l}^{\max} w_{\mathrm{Rx}_l}^{\max} \cdot (h_{\mathrm{Tx}_l\mathrm{Rx}_l}^t)^2 / H_{\mathrm{Tx}_l\mathrm{Rx}_l}^{\mathrm{LoS}}}{\underbrace{\sum_{k\in\mathcal{K}} p_k^t w_{k\mathrm{Rx}_l}^t w_k \cdot (h_{k\mathrm{Rx}_l}^t)^2 / H_{k\mathrm{Rx}_l}^t}_{Interference\ from\ UAVs} + \underbrace{\sum_{j\in\mathcal{L}, j\neq l} p_{\mathrm{Tx}_j} w_{\mathrm{Rx}_l} w_{\mathrm{Tx}_j} \cdot (h_{\mathrm{Tx}_j\mathrm{Rx}_l}^t)^2 / H_{\mathrm{Tx}_j\mathrm{Rx}_l}^t}_{Interference\ from\ Unpaired\ Incumbent\ Txs} + \underbrace{(\sigma_{\mathrm{Rx}_l})^2}_{Noise}} \tag{3.24}$$

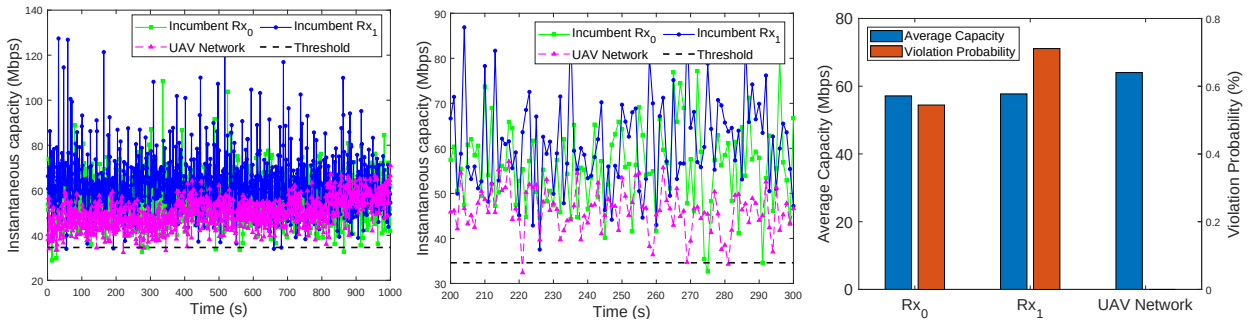
of spectrum sharing algorithm.

3.3.7 Extension to Multiple Incumbent Users

In previous Chapters, we consider the spectrum coexistence framework between a UAV network and a single incumbent user pair. The framework can also be extended to the scenarios with multiple incumbent user pairs. To this end, we need to further consider the interference among the unpaired incumbent transmitters and receivers, and consider the cross-system interference constraint for each of the incumbent user pairs.

Denote $\mathcal{L}$ as the set of the incumbent user pairs and further denote $\gamma_{\mathrm{Rx}_l}^t$ as the SINR of incumbent receiver Rx_l in time slot t. Then the SINR expression in (3.6) can be rewritten as in (3.24) at the top of this page. The SINR for UAV $k \in \mathcal{K}$ can be recalculated similarly. For each incumbent Rx $l \in \mathcal{L}$, a power adaptation coefficient η^t will be calculated following the same procedure as in Chapter 4.2. In the case that the UAV network is deployed nearby the overlapping area of multiple incumbent user pairs, the smallest η^t will be used for power adaptation for the UAV networks so that the cross-system interference constraints can be satisfied for all the incumbent user pairs. Recall in Chapter 3.2.4 that the power adaptation coefficient η^t can be calculated for each incumbent user pair based on bisection search with almost constant computational complexity. The overall computational complexity in the case of multiple incumbent user pairs is hence $O(|\mathcal{L}|)$, with $|\mathcal{L}|$ representing the number of incumbent user pairs in $\mathcal{L}$.

We further verify the effectiveness of the spectrum coexistence framework considering two and three incumbent user pairs. Consider 6 UAVs moving according to the ϵ-greedy



(a) (b) (c)

Figure 3.15: Spectrum sharing between two incumbent user pairs and 6 UAVs. (a) Instantaneous capacity of the incumbent system and the UAV network; (b) zoomed in plot of time slots 200-300; and (c) average capacity and violation probability.

RL algorithm with exploration probability 0.02. The results are reported in Figure 3.15 for two incumbent user pairs. Incumbent transmitters (Tx_0 and Tx_1) and receivers (Rx_0 and Rx_1) are deployed with coordinates (120, 250, 10), (380, 250, 10), (50, 250, 10) and (450, 250, 10), respectively. The threshold of violation probability is set to 5% for each incumbent receiver. It can be seen that the violation probability is lower than the threshold for both incumbent user pairs, which are 0.6% and 0.7% for incumbent receiver Rx_0 and Rx_1, respectively. Similar results can also be obtained for three incumbent user pairs.

Chapter 4

NeXT: Integrated Network Simulation, Experimentation and Optimization

In Chapter 3, we meticulously designed a spectrum sharing framework to foster harmonious coexistence among UAVs within the 6 GHz band, and we verified its effectiveness through simulation. However, to validate its real-world effectiveness, we now require a functional testbed. In this chapter, we present *NeXT*, an innovative software-defined testing framework designed for integrated RF network simulation, experimentation, and optimization. We delve into the data plane design in Chapter 4.1, followed by a comprehensive discussion on the control plane design in Chapter 4.2. To facilitate the seamless integration of multiple potentially mobile srsENBs in experimental research, we introduce a scheme enabling srsENBs to interface with srsEPC through wireless links, detailed in Chapter 4.3.In Chapter 4.4, we rigorously test *NeXT* and highlight its capabilities in optimization, simulation, and experimentation across various network control scenarios. Concluding our exploration, Chapter 4.5 delves into the novel avenues that *NeXT* can enable.

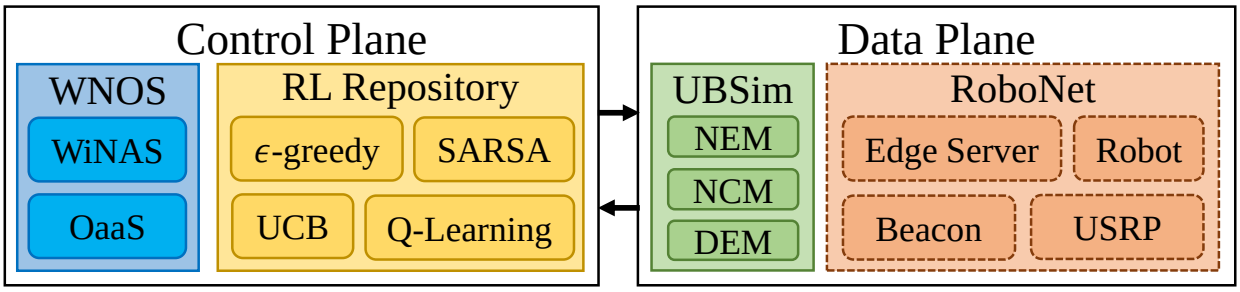


Figure 4.1: *NeXT* testbed architecture and paper organization.

4.1 Data Plane Design

The data plane provides the forwarding infrastructure for the *NeXT* testbed. As illustrated in Figure 4.1, two forwarding infrastructures have been designed: *UBSim* for software-based network simulations and *RoboNet* for experiments based on SDRs.

4.1.1 Software Simulations Based on UBSim

UBSim, evolved from simulators in [28,92,93], is a new wireless network simulator written in Python and based on the SimPy discrete-event simulation framework [89]. The simulator provides a configurable network-layer simulation supported by analytical models for various PHY- and MAC-layer protocols. This lightweight computational design enables faster-than-real-time iteration as well as on-the-fly adjustments to the protocol stack of each simulated node, which sets UBSim apart as a highly effective simulator for experiments focusing on protocol stack and topology self-configuration. The various node mobility types supported by UBSim enables investigation into aerial networking, including both UAV swarm and hybrid aerial-ground network control problems. UBSim supports general AI/ML algorithm deployments, and has been demonstrated for reinforcement learning (RL), deep RL, and multi-agent RL experiments.

As depicted in Figure 4.2, UBSim comprises three primary modules to handle the behavior definition of various network elements, as well as three APIs to support a wide range of custom networking scenarios. Specifically, the network element module (NEM) defines

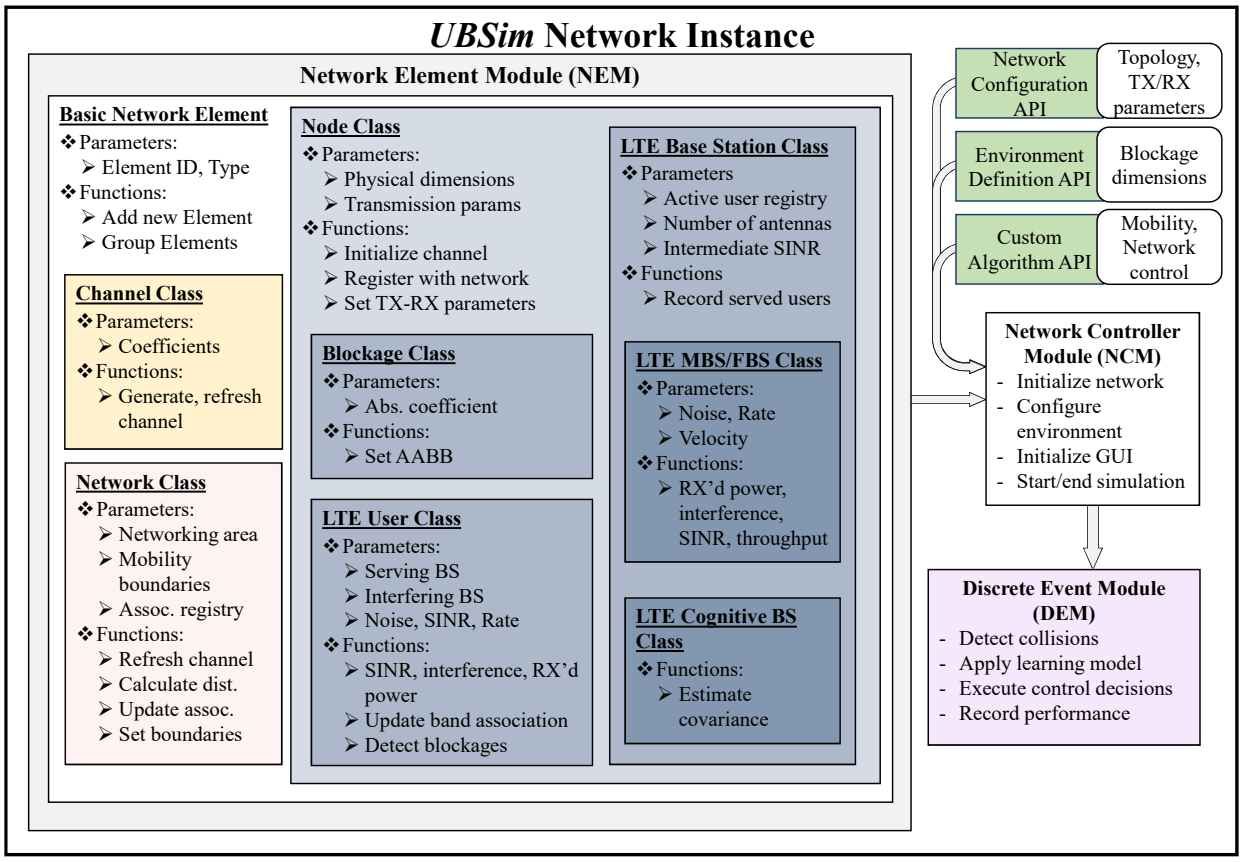


Figure 4.2: Architectural overview of UBSim network simulator.

the behaviors of all types of communication nodes, environmental blockages, channels, and the network as a whole. The network controller module (NCM) organizes the information from the NEM and each user API to define the network topology, environment, and control objective. The discrete event module (DEM) then takes the resulting full scenario definition and starts the discrete event-driven simulation process.

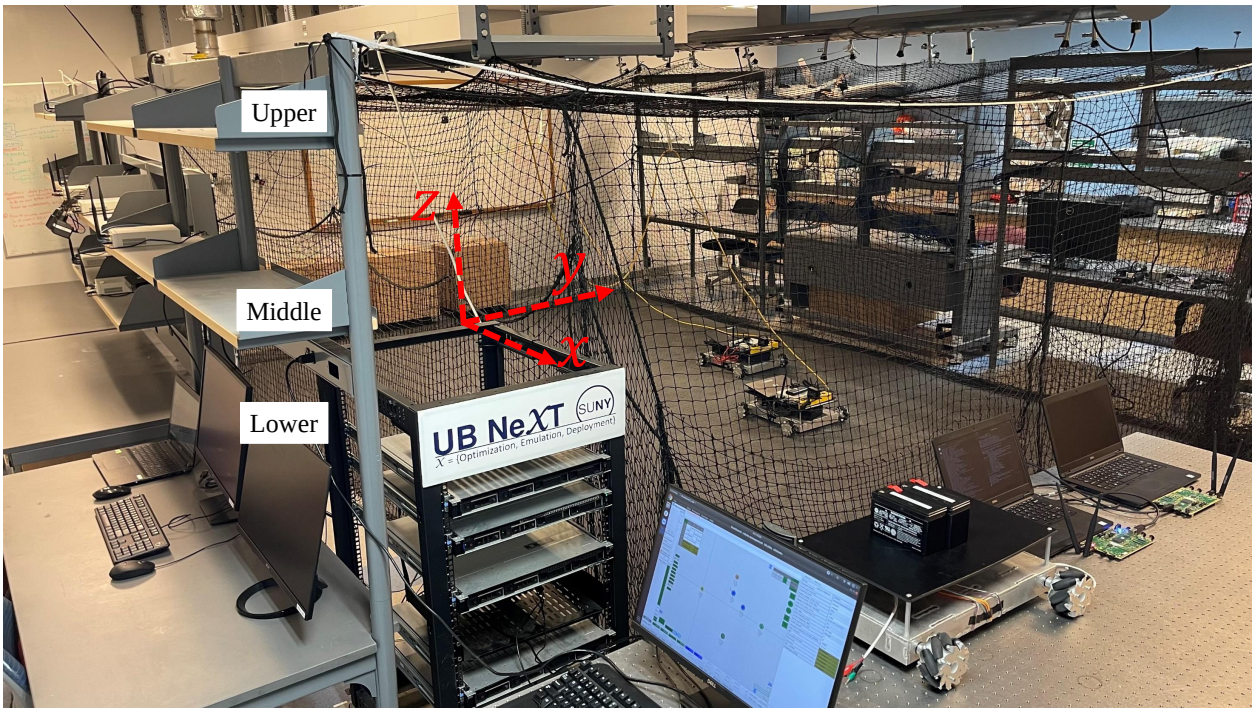
The simulator APIs offer full configuration over network behaviors, environment specification, and control specification. Specifically, the network configuration API provides control over parameters such as frequency, bandwidth, mobility, and location of nodes, as well as networking area and propagation characteristics. The environmental definition API provides control over the locations and sizes of blockages as well as their RF absorption coefficients over different frequency bands. In general, all physical environmental features, including lab benches, server rack, and UAV enclosure as shown in Figure 4.5(a), are modeled as block-

ages within the networking area. Finally, the custom algorithm API provides access to the run time behavior of all the nodes, such as mobility, transmission patterns, band association, among others. Particularly, this API module provides direct support for experimental applications of AI/ML for tasks such as network automation and self-configuration.

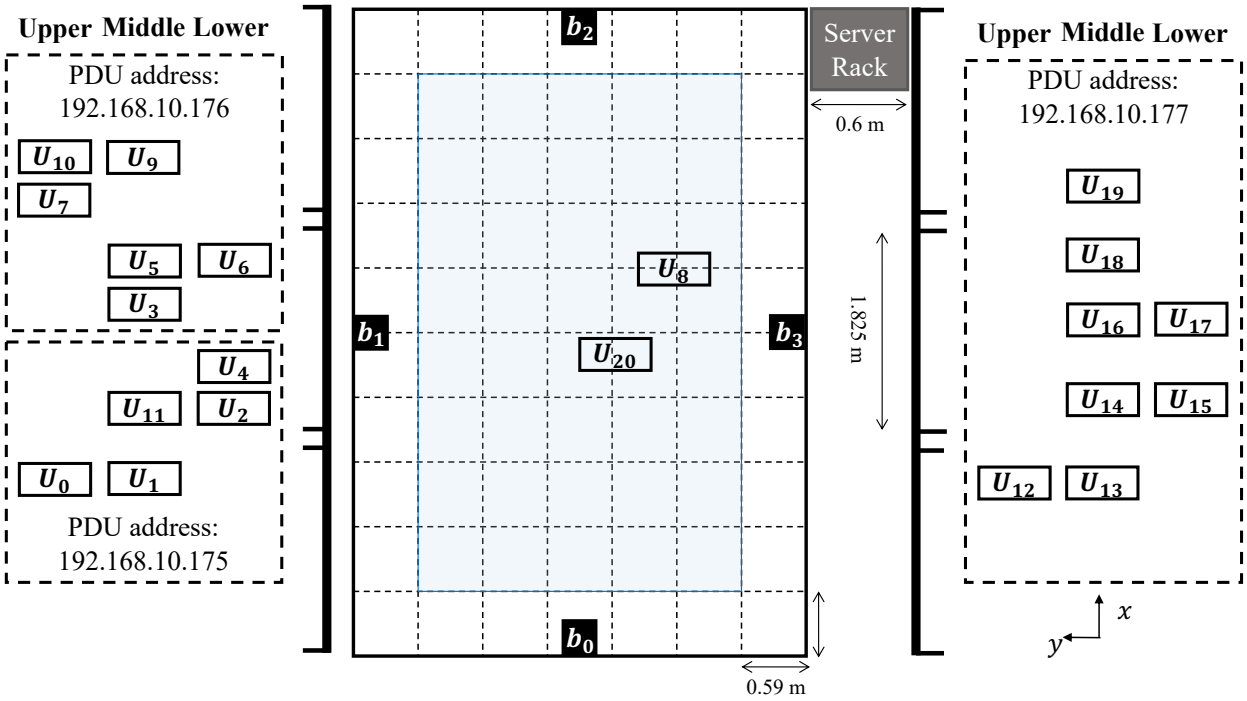
The parallel deployment of UBSim alongside the *NeXT* testbed provides several advantages. The highly configurable nature of UBSim provides a virtual sandbox in which experiments can be designed and evaluated for deployment on the *NeXT* testbed much faster than using SDR hardware alone. Additionally, the speed of simulation design and execution in UBSim enables pre-training or parallel training of AI models prior to deployment on hardware. This is particularly important for models in which significant amounts of environmental data must be available to generate an optimal solution, such as those used for deep learning and reinforcement learning. Furthermore, over-the-air data collected from the *NeXT* testbed can be used to improve the accuracy of data generated by UBSim by means of system identification [94], addressing challenges associated with high-quality data collection for AI/ML algorithms mentioned in Chapter 1.

4.1.2 Software-Defined Forwarding Infrastructure: RoboNet

The design objective of RoboNet is to support experiments in wireless networks with mobile robots, such as mobile hotspots [95] and wireless UAVs [96]. The testbed is located in 238 Davis Hall on the University at Buffalo's North Campus. Figure 4.5 shows a snapshot of RoboNet and the corresponding topology. At the center of RoboNet is a netted enclosure of dimension $6 \times 4 \times 2.1$ m^3, providing a safe space for robot navigation. For mobile nodes, three wireless robots have been designed based on SuperDroid vehicles and universal software radio peripheral (USRP) SDRs. An indoor navigation system is also designed based on Marvelmind beacons to provide indoor localization for the robots. For static nodes, a set of USRP SDRs have been deployed over the shelves on the left and right sides of the netted enclosure. All the static software radios are controlled by a server rack of five Dell workstations. The mobile

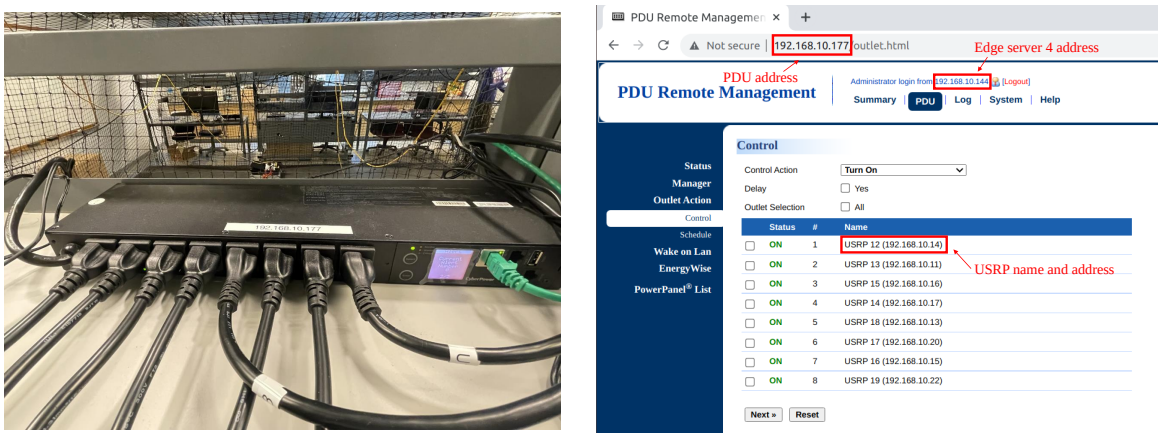


(a)



(b)

Figure 4.5: (a) Snapshot of the RoboNet testbed; (b) RoboNet network topology.



(a) (b)

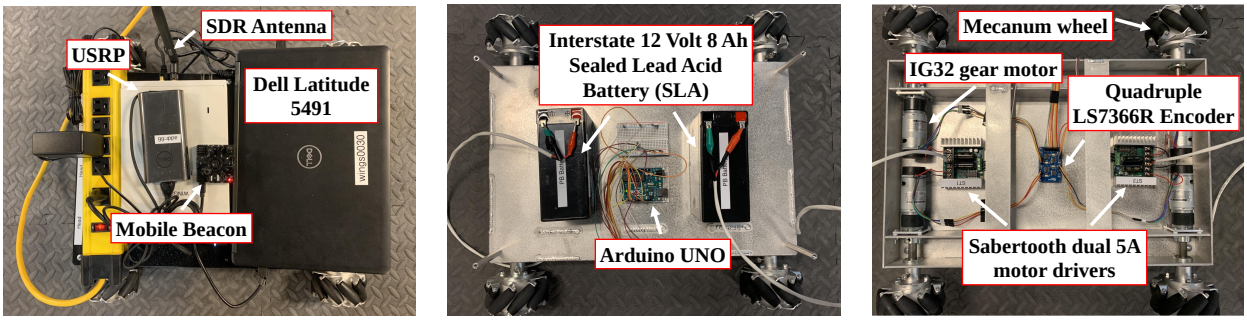
Figure 4.6: (a) Snapshot of PDU setup; (b) PDU remote management interface.

software radios are controlled by the robots' onboard computing hosts.

Static Nodes. The static nodes consist of 19 USRP N210, 5 USRP B210 SDRs and 1 wAP 60G (AP). Each USRP N210 operates at frequencies from DC to 6 GHz and can process up to 50 mega samples per second (MS/s). Each USRP N210 is equipped with a CBX daughterboard and two VERT900/VERT2450 antennas. These USRP SDRs are connected via two switches to a server rack, comprising four Dell EMC R340 PowerEdge workstations for baseband signal processing. Each USRP B210 is designed for low-cost experimentation with continuous frequency coverage from 70 MHz to 6 GHz. Each USRP B210 is also equipped with two VERT2450 antennas. The five USRP B210s provide flexibility because they can be deployed to any place depending on the requirements. The wAP 60G (AP) router is a product from MikroTik [97] and can be used either as a point-to-point primary or a point-to-multi-point primary.

The USRP SDRs are powered via three remotely accessible CyberPower Power-Distribution-Units (PDUs), as shown in Figure 4.6(a). These PDUs are assigned with Ethernet LAN IP addresses 192.168.10.175, 192.168.10.176 and 192.168.10.177 and connected to edge servers via switches. By getting access to the three default IP addresses, experimenters can power on, shut down and make a schedule with all static USRPs remotely. Figure 4.6(b) shows

the PDU remote management interface, via which experimenters can power on/off USRPs in real time or at scheduled times.



(a) (b) (c)

Figure 4.7: Snapshots of mobile node. (a) USRP software radio, control host, laptop, and mobile beacon; (b) Power unit and Arduino controller; and (c) Bottom view: motors, motor drivers and encoder.

Mobile Nodes. Three software-defined robot vehicles have been designed for RoboNet based on a combination of SuperDroid robots and USRP SDRs. Snapshots of the robot vehicles are shown in Figure 4.7. The SuperDroid robot serves as the mobile carrier of the software radios. A programmable Mecanum wheel vectoring robot has been used in the current design of the mobile nodes. Each robot comprises 4 Mecanum wheels, 4 IG32 gear motors, 2 Sabertooth dual 5A motor drivers, 1 Quadruple LS7366R Encoder and 1 Arduino UNO controller. Each robot is powered by two 18V/2.4A PB (lead-acid) batteries. This allows each robot vehicle to carry up to 50 lbs of payload, including the USRP SDRs and their controlling host. Each robot is equipped with USRP SDRs for programmable wireless communications. Currently, both USRP N210 and B210 can be supported by mobile nodes. Each robot can also carry a wAP 60G to enable mmWave communications.

A Dell Latitude 5491 laptop with Intel Core^{TM}i7-8850H CPU@2.6GHz*12 is used for robot control, USRP SDR control and baseband signal processing. The connection between the controlling laptop and the robot vehicle is established by an Arduino via USB port "/dev/ttyACM0". The mobile beacon is connected to the laptop via USB port "/dev/ttyACM1". The two default serial ports provide more flexibility of our testbed. For example, by access-

ing the USB serial port, experimenters can access the raw beacon location information and design their own position algorithms, rather than using algorithms that we provide. Finally, the movement of the robot is controlled and navigated by the Arduino and the beacon via serial communications.

(a) (b)

Figure 4.8: (a) Controller modem; (b) Super beacon.

Indoor Positioning System. Because of the poor reception of GPS signals in indoor environments, an indoor positioning system has been deployed, as shown in Figure 4.8. The system consists of a controller modem (Figure 4.8(a)) and 7 precise (with accuracy of ± 2 cm) Marvelmind Super-Beacons (Figure 4.8(b)). Based on this system, the location of the mobile beacon can be calculated using trilateration based on the propagation delay of ultrasonic signals to a set of stationary beacons.

The 7 super beacons are divided into two groups: 4 static and 3 mobile beacons. As shown in Figure 4.5(b), the 4 static beacons, b_1, b_2, b_3 and b_4, are attached to the four sides of the protective net. For example, Figure 4.8(b) shows the deployment of b_1, which can communicate with the controller modem, its neighbour beacons and the mobile beacon using the selected frequency (19/25/31/37 kHz). According to the exchanged information among the static beacons, the mobile beacon and the modem, the robot locations will be updated in real time. We adopt a Non-Inverse Architecture to set up the navigation system and 31 kHz is used as the communication frequency.

Finally, the controller modem is connected to the edge server via a USB port. Through

the control dashboard at the server, experimenters can define a network map by assigning the origin point of the 3D network, configuring beacon parameters (e.g., beacon address and mode), and monitoring the movements of the mobile beacons mounted on the robots.

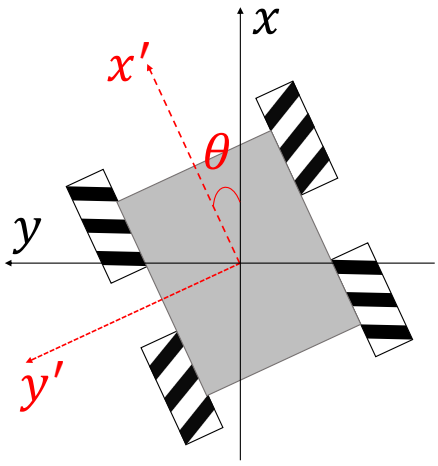


Figure 4.9: Mecanum wheel robot with angular deviation.

Robot Self-Adjustment Scheme. Since we focus on investigating the wireless communication network, we always hope that the robot will move as prescribed and arrive at its target location. However, with inaccurate readings from the encoder and different speeds of the four Mecanum wheels (shown in Figure 4.7(c)), the robot may fail to arrive at the expected position and collisions may happen when multiple robots exist. In order to focus on the wireless network study itself without worrying about the negative impacts induced by the robot, we propose a beacon-based robot self-adjustment scheme to allow the robot to automatically adjust its position and posture during experiments. The overall robot self-adjustment scheme is summarized in Algorithm 2.

There are two phases of the self-adjustment scheme: i) beacon-based robot posture adjustment and ii) beacon-based robot position adjustment. Due to the different speeds of the Mecanum wheels, there is a divergence angle θ between the movement direction of the robot and the network's x-axis, as shown in Figure 4.9, especially when the robot moves left or right. At the beginning of the adjustment, the robot records its beacon-based position (x_1, y_1). Since movement errors are negligible when moving short distances forward or

Algorithm 2: Robot Self-Adjustment Scheme

Beacon-based Robot Posture Adjustment:
- Measure current position (x_1, y_1) via beacon
- Robot moves forward for τ seconds
- Measure new position (x_2, y_2)
- Robot moves back to (x_1, y_1)
- Calculate the angle deviation based on (4.1)
- Determine the rotation direction and calculate rotation distance based on Table 4.2
- Arduino movement control

Beacon-based Robot Position Adjustment:
- Measure current position (x_3, y_3) via beacon
- Look up state position table and obtain target state position (x, y)
- Determine the movement direction and calculate movement distance based on Table 4.2
- Arduino movement control

return

backward, we have the robot move forward for τ seconds ($\tau = 3$ by default), record its new beacon-based position (x_2, y_2), and then move backwards for τ seconds back to its original position (x_1, y_1). With the recorded two positions, the divergence angle θ can be calculated based on

$$\theta = \begin{cases} 90°, & \text{if } x_1 = x_2 \text{ and } y_1 < y_2 \\ -90°, & \text{if } x_1 = x_2 \text{ and } y_1 > y_2 \\ \arctan(\frac{y_2 - y_1}{x_2 - x_1}), & \text{otherwise.} \end{cases} \tag{4.1}$$

With the divergence angle θ, the movement option and the movement distance can be obtained by referring to Table 4.1, in which d_1 is the measured reference distance, which is obtained as follows: When a robot turns left or right, one of its four wheels (left-back wheel by default) does not move and the other three wheels do. By reading the encoder value of one of the non-static wheels (the left-front wheel by default) when the robot rotates 360°, the value of d_1 can be obtained. The two adjustment parameters (θ and d_1) will then be packed in a message and sent to the onboard Arduino controller. With the received message,

the Arduino will control the robot to finish the beacon-based robot posture adjustment.

In the second phase, the robot first measures its new position (x_3, y_3) and compares it with the measurement-based beacon state information (x, y) which can be obtained via a one-time beacon-based measurement. The obtained distance divergence d_x and d_y for the x and y axis will be calculated and transformed to the corresponding movement direction and distance as shown in Table 4.1, in which d_2 and d_3 are the measured reference distance when the robot moves forward and backward for 1 meter, respectively. Similarly, the obtained adjustment parameters will be packed and sent to the Arduino. Once the Arduino receives the movement command, the robot will adjust its position and then finish the second-phase adjustment.

Table 4.1: Robot movement operation

Parameter	Movement Option	Movement Distance	Parameter	Movement Option	Movement Distance
$\theta > 0$	Rotate Left	$\lvert\theta\rvert/360 * d_1$	$\theta < 0$	Rotate Right	$\lvert\theta\rvert/360 * d_1$
$x_3 < x$	Move Forward	$d_x = (x - x_3) * d_2$	$x_3 > x$	Move Backward	$d_x = (x_3 - x) * d_2$
$y_3 < y$	Move Left	$d_y = (y - y_3) * d_3$	$y_3 > y$	Move Right	$d_y = (y_3 - y) * d_3$
Otherwise	Stop	0			

4.2 Control Plane Design

The control plane supports both traditional model-based control, enabled by WNOS, and emerging data-driven control, enabled by the RL repository. A set of APIs are developed for WNOS to enable automatic generation of distributed cross-layer control algorithms. While the RL repository is combined with a set of experiment management APIs and multiple communication protocols to ease the use of *NeXT* testbed and enable broadband wireless communication. The control plane is deployed over the edge servers which are placed in the shelf labeled as "UB NeXT" in Figure 4.5(a).

4.2.1 Network Modeling and Optimization Support

It is typically tedious and error-prone to manually model and optimize forwarding infrastructure in the data plane. To address this challenge, we deployed our previously designed WNOS [40,98–100] over *NeXT*. The primary benefits of WNOS are that it abstracts the data plan forwarding infrastructure, allows experimenters to define control objectives in a centralized manner using high-level APIs, and then automatically generates distributed cross-layer control algorithms that can be deployed on *NeXT*'s data plane, e.g., *UBSim* and *RoboNet*. At a high level, WNOS comprises two key components: network abstraction and network control problem decomposition and control program generation. The network abstraction provides a set of APIs, based on which experimenters can characterize in a centralized manner the desired network behaviors before actual deployment. The network control problem decomposition and control program generation is enabled by *disciplined instantiation (DI)* [40], based on which user-defined abstract centralized network control problems can be decomposed into a set of distributed subproblems. WNOS is designed based on a three-level hierarchical architecture to enable scalable network deployment. Specifically, at the first-level, the WNOS control host is connected to all second-level SDR control hosts via wireless interfaces (Wi-Fi in our current prototype). The generated distributed algorithms are automatically pushed over the wireless interfaces and installed at each of the SDR control hosts which form the third-level. Hence, one only needs to create a single piece of code to control all the SDR devices.

WNOS supports a wide set of network control problems in both static and mobile networks. These include, but are not limited to, *rate maximization, power minimization, end-to-end delay minimization, and movement optimization*. WNOS also provides a rich set of APIs, based on which experimenters are allowed to define more sophisticated control problems in next-generation broadband networks spanning across multiple frequency bands, e.g., microwave, mmWave as well as THz bands. Below are some examples of the APIs.

Table 4.2: Example APIs of WNOS

API	Description
`attach(·)`	Add elements to the network
`connect(·)`	Link one or more network elements
`install_model(·)`	Install an expression model for a network element attribute
`get_expr(·)`	Get the expression of a network element
`mkexpr(·)`	Construct the new expression
`record_expr(·)`	Store the expression in the database
`set_para(·)`	Designate a specific expression as a utility function, constraint, or optimization variable
`set_soln(·)`	Select the solution method to optimize the designated variables
`record_expr(·)`	Store the expression in the database

4.2.2 Data-Driven Network Control Repository

The second part of the control plane is the data-driven network control repository which enables data-driven control on RoboNet and makes it easy to modify advanced AI/ML algorithms to be compatible with our testbed. This repository consists of two classes of APIs for data-driven control, i.e., *Basic Class* and *Advanced Class*. The basic class is responsible for network initialization. Examples include the *Environment Initialization API, Variable Initialization API* and *Feedback List Initialization API*. The *Advanced Class* APIs are designed based on *Basic Class* and are used for policy training, including updating states, actions and a value table. Given the number of states and actions specified using the *Configuration API*, the environment can be initialized using the *Environment Initialization API*. Key variables involved in learning algorithms, such as the current state and next state, can be initialized via the *Variable Initialization API*. One is also allowed to choose the *Reward Type* and *Calculator Mode* through the *Configuration API*. Based on these APIs, four classes of RL algorithms have been implemented in the advanced class and can be called via the *RL Algorithm API*. These are epsilon-greedy search, upper confidence bound (UCB) action selection, Q-learning and State–action–reward–state–action (SARSA). Different reward types and calculator modes have been defined in advance, while experimenters can define custom reward types and calculator modes for their own experiments.

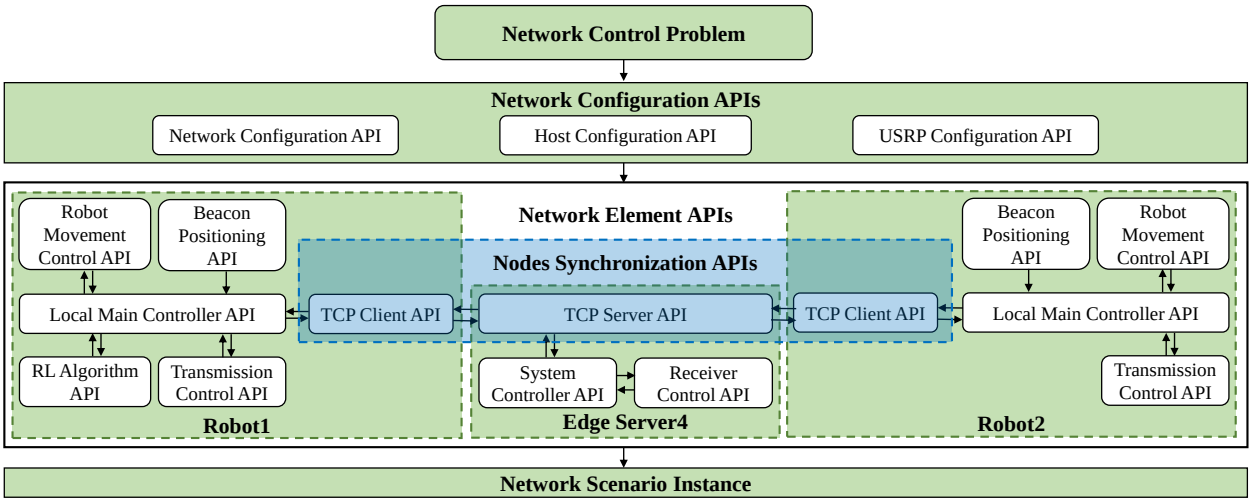


Figure 4.10: Network element control interface and experiment management APIs.

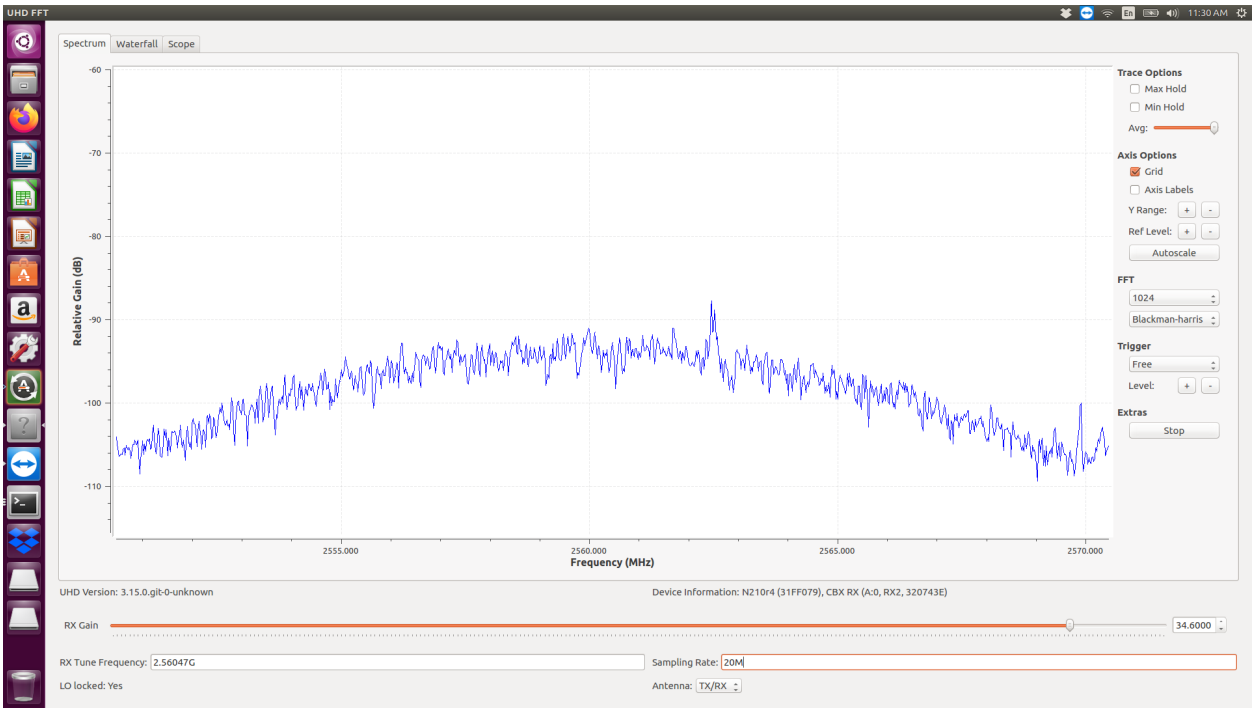
4.2.3 NeXT Experiment Management APIs

Extensive experiments can be conducted over the *NeXT* testbed, especially on *RoboNet* discussed in Chapter 4.1.2. To help experimenters use our testbed efficiently, we design a set of experiment management APIs, by which elements deployed on RoboNet can be coordinated. As shown in Figure 4.10, there are three classes of APIs, as discussed next.

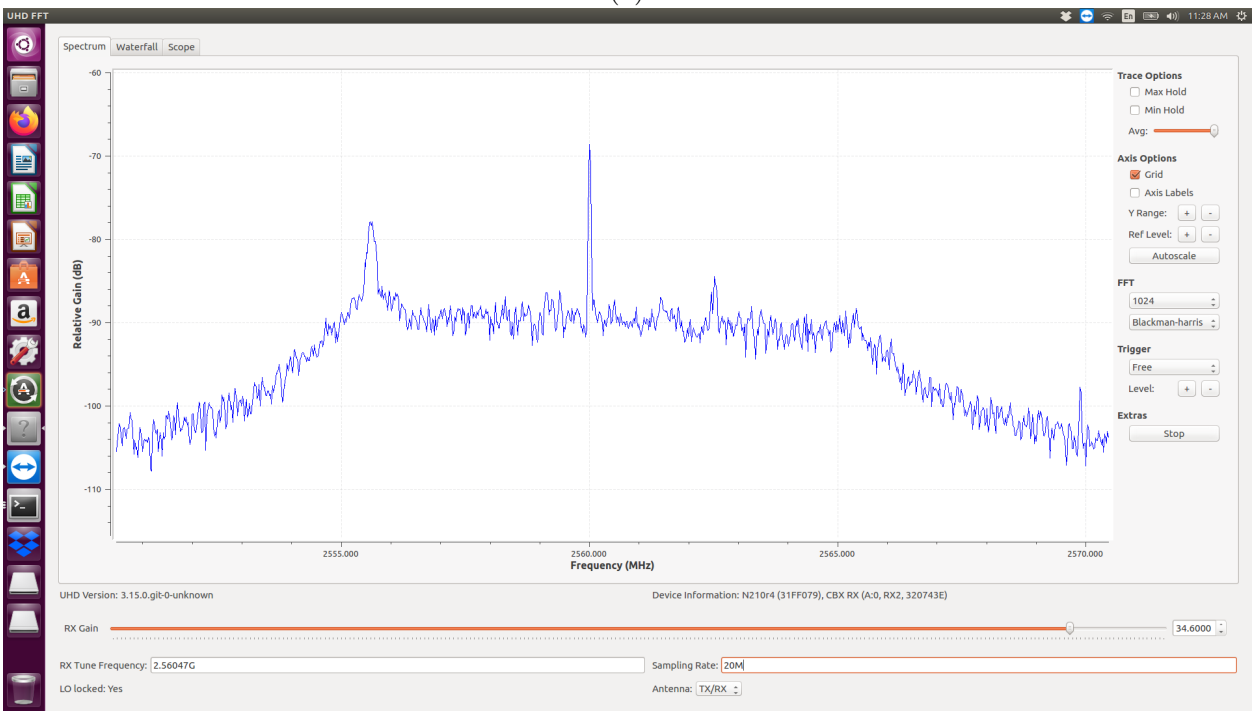
Network Configuration APIs. APIs in this class are used to define various network environments. We provide three different APIs, *Network Configuration API*, *Host Configuration API* and *USRP Configuration API*. Parameters that can be configured via network configuration APIs include network area, center frequency, bandwidth, transmission power, modulation type, slot duration, the number of robots, etc. Through host and USRP configuration APIs, experimenters can manage Ethernet addresses, wireless network addresses, and port numbers for the SDRs and their controlling hosts.

Nodes Synchronization APIs. To easily coordinate the server and mobile hotspot controllers, Nodes Synchronization APIs are provided. With these APIs, for example, experimenters can start the experiments with just one command executed on the edge server.

These APIs are based on a Transmission Control Protocol (TCP) connection established over WiFi to provide communications among different nodes. The WiFi wireless local area



(a)



(b)

Figure 4.13: Screenshot of 2.56 GHz spectrum monitor of network (a) in idle mode with -92 dBFS peak interference; (b) during experiments with -68 dBFS peak signal.

network is enabled by TP-Link Archer A7 AC1750 Wireless Dual Band Gigabit Router, which follows wireless LAN 802.11a/b/g/n/ac standards. The 2.4 GHz and 5 GHz bands are dedicated for node synchronization and we avoid using the two bands for conducting experiments. Thus, the WiFi will not cause any interference to our target experiments. The other potential external interferes are mainly from wireless devices like phones. However, since most wireless devices get access to the internet via University at Buffalo's WiFi network, which also works in the 2.4 GHz and 5 GHz bands, the interference to measurements is limited. In Figure 4.13, we show the spectrum comparison without and with ongoing experiments, respectively. The results show that the possible interference (-92 dBFS) to our experiments is much smaller than our signal strength (-68 dBFS). Thus we can neglect the possible interference.

Network Element APIs. After the experiment profile has been configured, one can further control various network components via a set of system control APIs deployed at the edge server and mobile hotspot controller. These include the *Transmission Control API*, which can be used to control the transmissions of the USRP N210 carried by the robot vehicle; the *Receiver Control API* for controlling data receiving; the *Robot Movement Control API* for controlling robot movement; and finally the *Beacon Positioning API*, based on which experimenters can obtain the robots' real-time positions.

In these network element APIs, logging features are enabled to record system status like transmission process startup, beacon positioning updates, robot movements and so on. Data that will be used for analysing and processing, like throughput for each time slot, are stored and updated in dictionaries/tables during the tests and saved automatically once an experiment finishes.

```
1 import Reinforcement_Learning_API as rli
2 import Robot_Movement_Control_API as rmi
3 import Beacon_Positioning_API as bpi
4
5 while experiment_running:
```

```
6     curt_state = bpi.operation()
7     next_state = rli.operation()
8     updt_rbt_movement_ctrl(curt_state, next_state)
9     updt_usrp_commn_ctrl()
10    updt_fdbk_request()
11    reward = rli.fdbk_processing(fdbk, fdbk_type, rwd_calc)
12    rli.updt_value_table(reward)
13    if rbt_adjustment_status:
14        rmi.rbt_adjustment(next_state)
```

Listing 4.1: Example of Experiment Management APIs

In Listing 4.1, we show an example of using the aforementioned APIs to conduct experiments on the *NeXT* testbed. While an experiment is running (line 5), the user calls *Beacon Positioning API* to get the current state information (line 6) and calls *Reinforcement Learning API* to get the next state information (line 7). The robot updates its location by calling *Robot Movement Control API* (line 8). After the robot arrives at the target location, the communication begins (line 9). After a pre-defined communication time in *Network Configuration API*, the robot requests feedback (line 10) and obtains the current reward (line 11), and the value table is then updated (line 12). Before conducting the next time-slot experiment, the robot adjustment status parameter defined in *Network Configuration API* will be checked (line 13). If the status is *True*, the robot will adjust its posture and position based on Algorithm 2 (line 14).

4.2.4 Communication Protocols Management

We consider three different communication protocols in our testbed: GNU Radio Benchmark, srsRAN and mmWave communication protocols.

GNU Radio Benchmark Protocol. This is developed based on GNU Radio narrow-band benchmark library [101]. Specifically, we extend the original benchmark narrow-band

library by designing three additional APIs. These are *Benchmark Interaction API*, *Benchmark Transmission Control API* and *Benchmark Receiving Monitor API*. For example, the *Benchmark Transmission Control API* is used to control when and what data is transmitted. The transmission duration and transmission information can be configured in *Network Configuration API* in Chapter 4.2.3 and can then be transmitted to the basic benchmark module via *Benchmark Interaction API*. *Benchmark Receiving Monitor API* is used to monitor the status of the receiver. If the receiver detects disconnected links, it will restart the transmitter by sending a request to the edge server via *Benchmark Interaction API*.

```
1  import Network_Configuration_API as ncfg
2  import Host_Configuration_API as hcfg
3  import Benchmark_Interaction_API as bmia
4
5  def Benchmark_Transmission_Control_API():
6
7      if bmia.tmr1 == 0:
8          data = ncfg.cxn_data
9      elif bmia.tmr2 <= ncfg.ts_len:
10         data = ncfg.comm_data
11     else:
12         data = ncfg.cxn_data
13         bmia.socket.sendto(ncfg.cmd_done, (hcfg.wl_host, hcfg.port))
14
15     return data
```

Listing 4.2: Example of *Benchmark Transmission API*

Listing 4.2 shows an example of how *Benchmark Transmission Control API* is used to control the transmission of data at run time. There are two types of data that can be transmitted: regular *transmission data*, which is the actual data that we want to deliver over the network, and dummy *connection data*, which we use to keep the network connection alive. The connection data is needed because GNU Radio does not provide an auto re-connection

scheme if a connection is lost. In Listing 4.2, the experimenter first calls the *Benchmark Transmission Control API* (line 5) to determine the data to be transmitted based on the two timers received from *Benchmark Interaction API*. If timer 1 (*bmia.tmr1*) equals 0 (line 7), the data is set as connection data (*ncfg.cxn_data*, line 8); if timer 2 is (*bmia.tmr2*) smaller than a predefined transmission duration (*ncfg.ts_len*, line 9), the data is set as communication data (*ncfg.comm_data*, line 10); otherwise, the data is set to *ncfg.cxn_data* (line 12) and the *one-time-slot-finished* information will be sent to *Local Main Controller API* via *Benchmark Interaction API* (line 13).

Software-Defined RAN Protocol. This is developed based on srsRAN, an open-source 4G and 5G software radio suite developed by Software Radio Systems (SRS) [102]. It contains three different modules, srsEPC, srsENB and srsUE. We design a set of *srsRAN Configuration APIs* to manage the three modules based on the parameters in *Network Configuration API*. For example, the user dataset information can be generated automatically and stored in "user_db.csv" via *srsEPC configuration API*, and srsENB operation parameters like communication frequency can be generated automatically via *srsENB configuration API*. The *srsUE configuration API* is used to generate "ue.conf" file which contains srsUE operation information, such as IMSI information.

```
1  import os
2  import srsEPC_configuration_API as epca
3  import srsENB_configuration_API as enba
4  import srsUE_configuration_API as suea
5
6  epca.srsepc_operation()
7  os.system("gnome-terminal -- bash -c \"sudo srsepc; exec bash\"")
8  enba.srsenb_operation()
9  os.system("gnome-terminal -- bash -c \"sudo srsenb; exec bash\"")
10 suea.srsue_operation()
11 os.system("gnome-terminal -- bash -c \"sudo srsue; exec bash\"")
```

Listing 4.3: Example of *srsRAN Configuration APIs*

Listing 4.3 shows an example of how to generate srsRAN configure files and run the corresponding programs. Users call *srsEPC_operation* (line 6) to generate "user_db.csv" and start up srsEPC program in line 7. Similarly, "enb.conf" and "ue.conf" are generated by calling *srsENB_operation* (line 8) and *srsUE_operation* (line 11), respectively.

Millimeter-Wave Communication Protocol. The mmWave communication protocol is supported by MikroTik mmWave routers [97]. These mmWave routers can be configured to form a point-to-point network or point-to-multi-point network based on requirements. When integrating the mmWave communication protocol with srsRAN to enable large scale wireless network communication, we connect both the USRP B210 (running srsEPC and srsENB) and the mmWave router (primary) to a single laptop and design a *Gateway Setting API* to navigate data traffic between srsUE and mmWave subordinate.

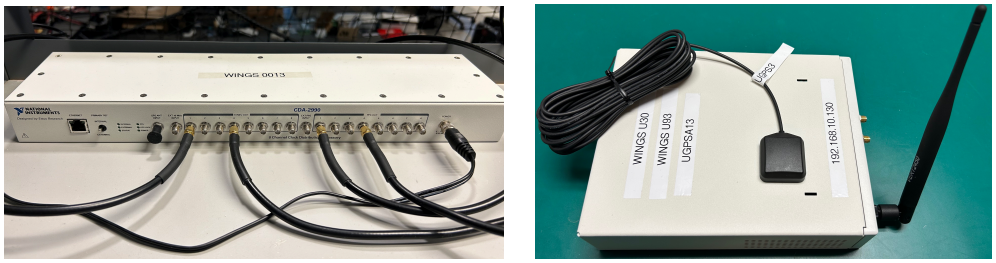
```
1  import os
2  import Network_Configuration_API as ncfg
3  import Host_Configuration_API as hcfg
4
5  def Gateway_Setting_API():
6
7      subnet = hcfg.srsRANsubnet
8      eth_addr = getattr(hcfg, ncfg.laptop_name).get("eth_host")
9      gw_setting_cmd = "gnome-terminal -- bash -c \"sudo ip route add " + str(
       subnet) + " via " + str(eth_addr) + " ; exec bash\""
10     os.system(gw_setting_cmd)
```

Listing 4.4: Example of *Gateway Setting API*

Listing 4.4 shows an example of how to set the gateway to enable communication between srsUE and mmWave subordinate. The key is to get the subnet address of srsRAN (line 8) and Ethernet address of the current laptop (line 9). The command is generated based on the above two information (line 10) and the gateway is set by executing the command (line 11).

Since the three communication protocols have their own logic stacks and the interactions

with controllers are processed in different ways, we implement three different profiles for each communication protocol. These three profiles are stored in laptop controllers and edge servers. Each profile is independently stored in different folders but they share the same components except communication protocol. Experimenters can choose the profile to load, i.e., select the communication protocol they want to use, before conducting experiments. By providing different profiles for different communication protocols, we can easily integrate more communication protocols, like direct sequence spread-spectrum (DSSS) [103], to our testbed in the future.



(a) (b)

Figure 4.14: (a) Snapshot of the Octocolock; (b) GPS module

The three communication protocols that we have implemented all rely on self-synchronization schemes. For example, srsRAN implements the Precision Time Protocol (PTP), which is a standard protocol used for time synchronization in packet-based networks, so no clock/time synchronization features are needed. However, protocols that rely on clock/time synchronization can be implemented with support of additional hardware, such as an Octoclock or GPS module, as shown in Figure 4.14. The accuracy of Octoclock is within a few 100 ns and GPS module is within 50 ns.

4.3 Scaling Out srsRAN

srsRAN has been widely used in experimental research for 5G, 6G and their evolutions. However, in the current implementation of srsRAN, the srsENB and srsEPC are interfaced

through wired connections, which makes it challenging to conduct experiments with mobile srsENBs and dynamic association between User Equipment (srsUE) and srsENB in future wireless networks. To address this challenge, we propose to interface srsENB and srsEPC by allowing srsENB to interface with srsEPC through wireless links and hence enabling easy integration of multiple possibly mobile srsENBs in experimental research [104]. We show the effectiveness and scalability of the new srsRAN architecture through two demonstrations: (i) srsUE-srsENB connection establishment; (ii) srsUE handover between two srsENBs wirelessly interfaced with the same srsEPC.

In this research, we focus on enhancing the scalability of srsRAN, which is an open-source 4G LTE/5G NR commercial-grade software radio suite developed by Software Radio Systems and has been widely used in experimental research for 5G, 6G and their future evolutions.

4.3.1 srsRAN: A Primer and Challenges.

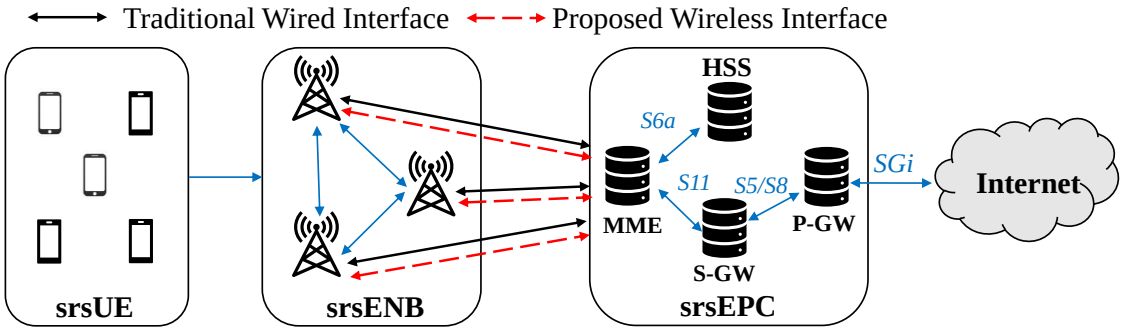


Figure 4.15: Diagram of the traditional srsRAN architecture with wired interface (black solid lines) between srsENB and srsEPC and the architecture with wireless interface (red dashed lines).

As illustrated in Figure 4.15, the srsRAN framework consists of three major components, namely the User Equipment (srsUE), Evolved NodeB (srsENB) and Evolved Packet Core (srsEPC). The srsUE is a software based 4G LTE and 5G NR UE modem capable of connecting to any LTE or 5G NR network and providing high-speed mobile connectivity. The srsENB is the software based LTE eNodeB basestation that connects to srsEPC. The srsEPC is a lightweight implementation of the LTE core network that consists of Home Subscriber

Service (HSS), Mobility Management Entity (MME), Service Gateway (S-GW) and Packet Data Network Gateway (P-GW) modules. The HSS module is the user database that stores user information such as user id, key and usage limits. It is also responsible for authorizing the user to connect to the network. The MME module is the main control element in the network that handles mobility. The S-GW module is responsible for setting up sessions between the srsENB and P-GW. Finally, the P-GW module acts as the point of contact with external networks.

In the current deployment of srsRAN, the srsENB and srsEPC are interfaced with each other through wired connections. As a result, it is hard to support experiments for wireless networks with mobile hotspots, where base stations can be carried on ground or flying vehicles [6, 28]. A natural question to ask is: *Can we interface srsENB with srsEPS through wireless links and hence enhance the scalability of srsRAN in experimental research for NextG networks with mobile hotspots?*

4.3.2 Interface Analysis and Adaptation

We first identify the modules and functionalities of srsRAN that will be affected by the adoption of a wireless interface between srsENB and srsEPC. In srsENB there are two layers connected to srsEPC. The S1 Application Protocol Layer (S1-AP) in srsENB is connected to MME in srsEPC to provide the control plane connection, and the GPRS Tunneling Protocol User Plane Layer (GTP-U) in srsENB is connected to S-GW to provide the data plane connection. The interface for these two connections are S1-AP and S1-U, respectively. To enable connecting srsEPC and srsENB wirelessly, the interfaces for both S1-AP and S1-U in srsEPC and srsENB need to be adapted properly. To this end, both hosts running srsEPC and srsENB need to establish a wireless link. In our prototyping, we consider commercial off-the-shelf Wi-Fi links as an example, while other wireless links can also be adopted, e.g., mmWave- and terahertz-band links.

Denote `wl_epc_addr` and `wl_enb_add` as the wireless IP address of hosts running srsEPC

and srsENB, respectively. Then, we need to further modify the two interfaces based on the allocated wireless IP addresses in srsEPC configuration file. Specifically, we need to update MME Bind Address (`mme_addr`) and GTP-U bind address (`gtp_bind_addr`). The former specifies where the MME will listen for upcoming srsENB S1-AP connection and the latter specifies the tunnel address of S-GW for transmitting and receiving information to and from GTP-U at srsENB. Both of the two parameters need to be configured as `wl_epc_addr`. Furthermore, three bind addresses need to be modified for the two interfaces, that is GTP-U bind address (`gtp_bind_addr`), S1-C Bind Address (`s1c_bind_addr`) and MME Address (`mme_addr`), where `s1c_bind_addr` is used to for S1-AP connection. Both `gtp_bind_addr` and `s1c_bind_addr` need to be assigned with `wl_enb_add`, and `mme_addr` needs to be assigned with `wl_epc_addr`.

After establishing the wireless link between srsEPC and srsENB, the parameters stored in UE's USIM card should be added in the `user_db.csv`. This is a separate configuration file used by srsEPC to store the details of the users in HSS. The following parameters need to be updated in the `user_db.csv` file for each UE: ue_name (Human readable name to help distinguish UE's), algo (Authentication algorithm used by the UE like XOR and MILENAGE), imsi (UE's IMSI value), K (UE's key stored in hexadecimal), `OP_type` (Operator's code type, either OP or OPc), `OP_value` (Operator Code/Cyphered Operator Code), AMF (Authentication management field), SQN (UE's Sequence number for freshness of the authentication), QCI (QoS Class Identifier for the UE's default bearer), `IP_alloc` (IP allocation strategy for the SPGW). When `IP_alloc` parameter is set as "dynamic", SPGW automatically allocates IP address to UE. When `IP_alloc` parameter is set to a valid IPv4 (IP address must be in the same subnet as that of srsENB and srsEPC), SPGW statically assigns an IP address to UE.

4.4 Example Experiments Over NeXT

We now test *NeXT* and showcase its capabilities of optimization, simulation and experimentation considering different network control problems. These include user scheduling in a cellular network, trajectory optimization for a mobile hotspot, and joint rate and power control in multi-hop networks. A comprehensive overview of these experiments is summarized in Table 4.3.

Table 4.3: Experiments Overview

Experiment No.	Name	Type
1	User Scheduling	simulation & experimentation
2	Overflow Control	experimentation
3	Mobile Hotspot Navigation	experimentation
4	Multi-Mobile Hotspots Navigation	experimentation
5	Mobile Hotspot Navigation with srsRAN	experimentation
6	Mobile Hotspot Navigation in IAB	experimentation
7	Multi-hop Network Optimization	optimization & simulation & experimentation

4.4.1 Experiment 1: User Scheduling

In the first experiment, we consider a wireless network with a hotspot serving a set of users. The transmission time is divided into a set of consecutive time slots. In each time slot, we consider that the hotspot can serve at most one user. The objective of the hotspot is to maximize the aggregate throughput by selecting a user to serve in each time slot. We design control algorithms for the hotspot based on the data-driven network control repository as discussed in Chapter 4.2.2. Specifically, we consider the upper confidence bound (UCB) action selection algorithm and test it over both UBSim and RoboNet developed in Chapter 4.1. First, we test the effectiveness of the UCB algorithm in UBSim. Figure 4.16(a) plots the achievable capacity averaged over 20 episodes each with 100 time slots. It can be seen that the average capacity improves over time, and this validates the effectiveness of the data-driven network control repository.

Then we further test the data-driven network control repository over RoboNet considering SDRs and real-world wireless channels. USRP20 is selected as the transmitter and five

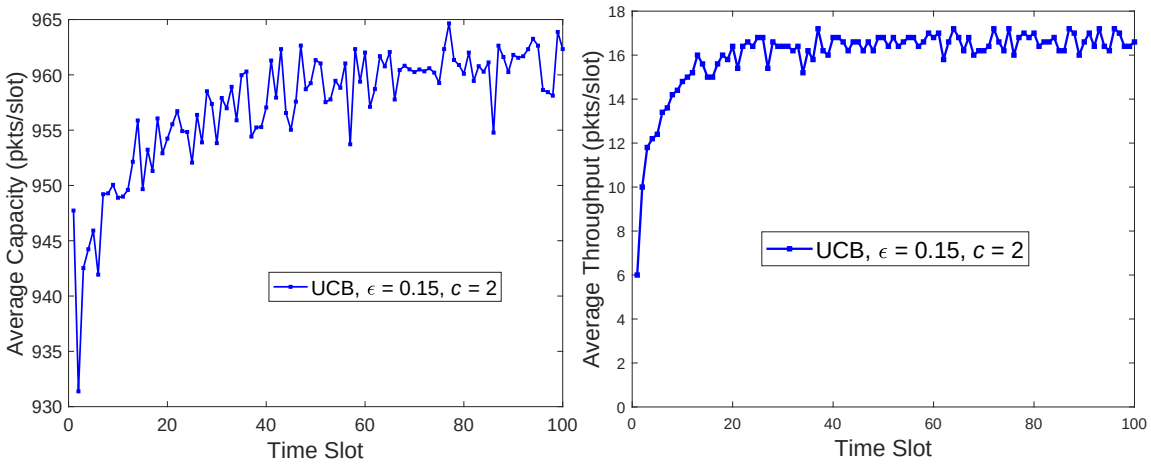


Figure 4.16: User scheduling scenario: (a) Average capacity obtained over UBSim; and (b) average throughput using RoboNet.

USRPs (USRP2, USRP5, USRP9, USRP11 and USRP19) are selected as receivers (see Figure 4.5). The time slot duration is set to 3 seconds. The exploration parameter ϵ and UCB control parameter c are set to 0.15 and 2, respectively. We run 10 episodes of robot navigation, with each episode consisting of 100 time slots. We calculate the average number of received packets in each time slot and the results are shown in Figure 4.16(b). It can be seen that the highest throughput can be achieved in around 20 time slots. This further validates the effectiveness of the data-driven network control repository. Comparing Figures. 4.16(a) and (b), we found the average capacity in UBSim is much larger than the average throughput on RoboNet. This is because we use different protocols in each system. In UBSim, the network capacity is calculated based on the Shannon capacity formula while on RoboNet, the throughput is obtained based on GNU Radio's narrowband communication protocol. Besides, the transmission power, bandwidth and so on are different in UBSim and RoboNet. For these reasons, the gap between UBSim and RoboNet is large. Since we focus on the verification of algorithms effectiveness, we neglect the gap between the simulator and reality. However, it would be interesting to investigate how to mitigate the reality gap, which is also a potential function provided by the *NeXT* system. We discuss this further

in Chapter 4.5. Recall that the primary objective of *NeXT* is to provide an integrated environment for optimization, simulation, and experimentation in software-defined wireless networks. This experiment illustrates the benefits of using the *NeXT* testbed. Conducting experiments in the real-world is time consuming while simulation-based experiments can be done much quicker. With the *NeXT* testbed, users can test their algorithm in UBSim first and check the performance of the proposed algorithm. If the results show that the algorithm needs improvement, they do not need to do the tests in the real-world, which saves time. By conducting experiments on our testbed, users also avoid directly collecting data in the real-world, which can sometimes be unsafe.

```
1 usrp_rx_list = ["usrp2","usrp5","usrp9","usrp11","usrp19"]
2 ts_len = 3                    # time slot length; unit: second
3 ts_nim = 100                  # time slot number
4
5 Q_Epsilon = 0.15
6 Q_StepSize = 0.2
7 Q_DiscountRate = 0.95
```

Listing 4.5: User scheduling configuration parameters in *Network Configuration API*

Listing 4.5 shows user scheduling configure parameters in *Network Configuration API*, which can be used to configure the parameters involved in this experiment. Experimenters can specify the USRPs they want to use as the receiver in line 1. The time-slot length and time-slot number for each episode can be set via line 2 and line 3, respectively. The parameters used for the UCB action selection algorithm configuration can be configured from line 5 to line 7.

4.4.2 Experiment 2: Overflow Control

In this scenario, Robot1 is adopted as the transmitter and three USRPs (USRP1, USRP9 and USRP11) are adopted as receivers. Each time slot is set to 5 seconds and the transmitter transmits data every 0.15 seconds. Assume that the arrival rate follows a Poisson distribution

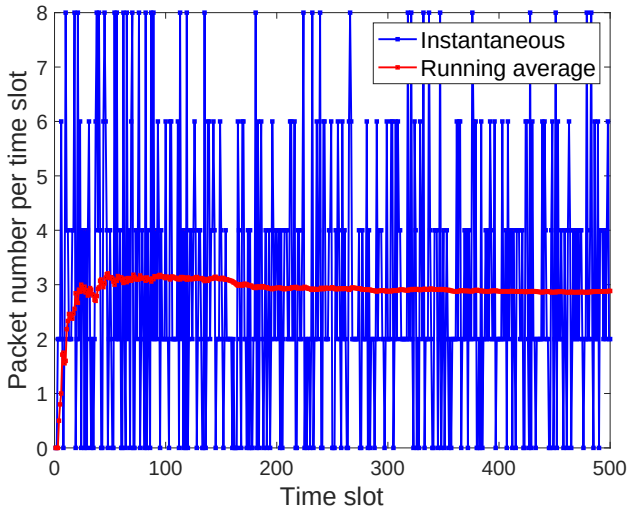


Figure 4.17: Overflow control scenario: Transmitted packet number vs. time slot

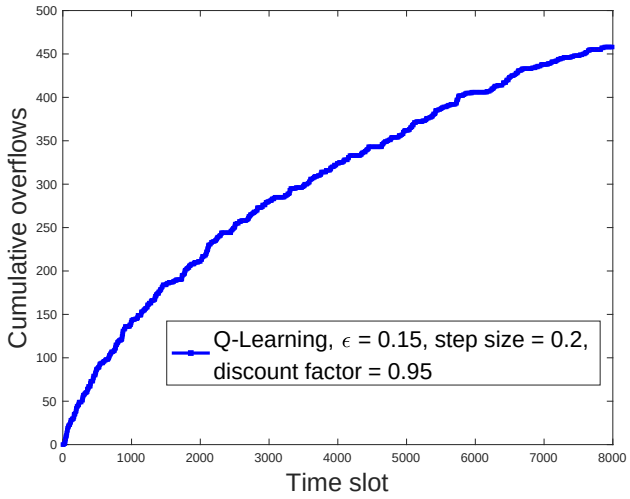


Figure 4.18: Overflow control scenario: Cumulative overflows vs. time slot

and the average arrival rate is set to 1 packet per time slot. The maximum data buffer length is set to 8 packets for each receiver. Q-learning is adopted in this case to control data buffer overflows. As shown in Figure 4.17, we run one episode with 500 time slots and calculate the number of transmitted packets and the corresponding running average for each time slot. It can be seen that, in some time slots the number of transmitted packets is 0. This happens when there are no packets available in the buffer to transmit or when the channel conditions are bad. For Poisson distributed packet arrivals with average arrival rate 1, the expected number of packets arriving at each user in each time slot is 1. Thus, the total

expected arrivals for three users in each time slot is 3. From Figure 4.17, the running average is around 3 packets per time slot which matches the above mathematical analysis. The cumulative overflows are shown in Figure 4.18, in which the slope converges gradually over time towards the optimal achievable packet overflow rate.

```
1 pkts_arrival_rate = 1                # unit: packet/slot
2 queue_max_pkts_num = 8               # maximum buffer size
3 queue_overflow_reward = -20
```

Listing 4.6: Overflow control configuration parameters in *Network Configuration API*

As shown in Listing 4.6 experimenters can modify the parameters in *Network Configuration API* to meet their needs. Experimenters can modify the packet arrival rate and maximum buffer length for each user via line 1 and line 2, respectively. Line 3 can be configured to set an overflow reward (such that a negative value corresponds to a penalty).

4.4.3 Experiment 3: Mobile Hotspot Navigation.

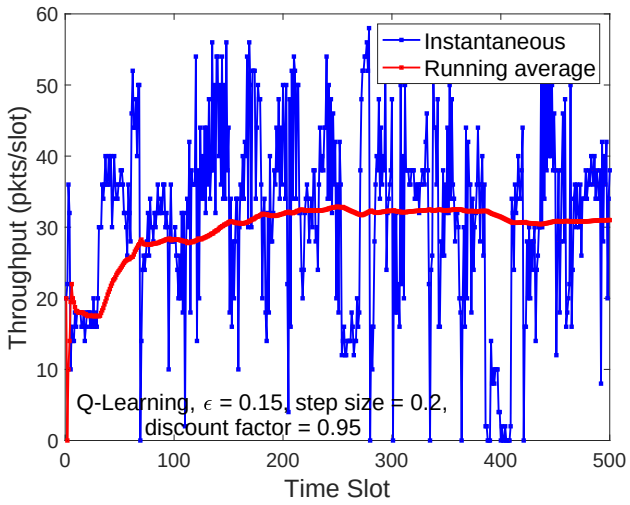


Figure 4.19: Single mobile hotspot scenario: instantaneous and running average of throughput.

In the third experiment, we consider a wireless network where a robot carrying a mobile hotspot moves around to serve a set of users. The objective is to maximize the users' aggregate throughput by controlling the robot's trajectory. The network is divided into a

set of grid cells, each corresponding to a state of the environment. In each grid cell, the robot has five action options, i.e., move forward, move backward, move left, move right and stay. The reward for each state-action pair is defined as the sum throughput of users. Q-learning is considered in this experiment with exploration probability ϵ set to 0.15, step size of 0.2 and discount factor 0.95. Each episode consists of 500 time slots, corresponding to 3 hours. We measure the number of received packets and calculate the corresponding running average in each time slot. The experimental results are reported in Figure 4.19. It can be seen that the running average converges to around 30 packets/slot. The drop of instantaneous throughput around time slot 400 is caused by the imperfection of the wireless link, which got disconnected as the robot moved.

4.4.4 Experiment 4: Multi-Mobile Hotspots Navigation.

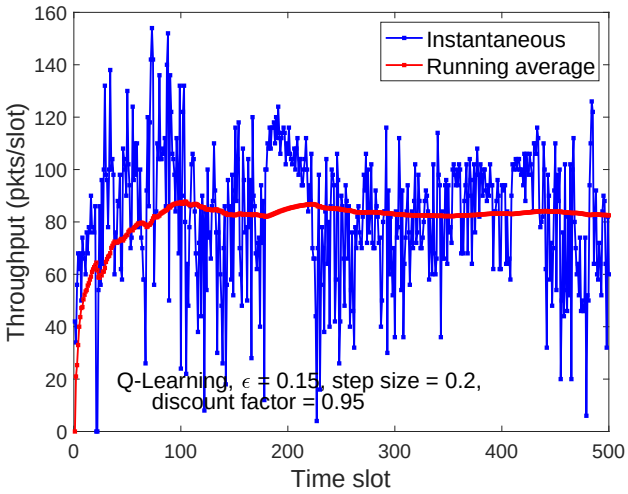


Figure 4.20: Two mobile hotspots scenario: instantaneous and running average of throughput.

In the fourth experiment, we consider the same wireless network scenario as the third except that we adopt two mobile hotspots. To avoid collisions, the network is divided into two regions and each robot can only move within one region. Five USRPs (USRP0, USRP1, USRP2, USRP3 and USRP19) are configured as users to receive service from the two robots. In each time slot, a user is only allowed to connect to the robot with the shortest distance to

it. Q-learning with the same parameters as in the second experiment is adopted. Similarly, we measure the number of correctly received packets and calculate the corresponding running average in each time slot. The experimental results are reported in Figure 4.20. It can be seen that the running average converges to around 80 packets/slot.

```
1  import struct
2  import Network_Configuration_API as ncfg
3  import Beacon_Positioning_API as bpi
4  import Host_Configuration_API as hcfg
5
6  def update_rbt2_state():
7
8      curt_state = bpi.operation()
9      s_addr = hcfg.rbt2.get("code")
10     d_addr = hcfg.rbt1.get("code")
11
12     data = struct.pack('!H', s_addr & 0xffff) + struct.pack('!H', d_addr & 0
       xffff) + struct.pack('!H', curt_state & 0xffff) + ncfg.rbt2_state_info
13     rb2_tcp_skt.sendto(data, ((hcfg.rbt1.get('host'), hcfg.rbt1.get('port'))))
```

Listing 4.7: Example of multi-robot interaction

In Listing 4.7 we give an example showing how Robot2 sends its state information to Robot1 during the experiments. Robot2 first gets its current state information via interaction with *Beacon Positioning API* (line 8). By calling *Host Configuration API*, the message source code (line 9) and message destination code (line 10) are obtained for message routing. Then the data is constructed (line 12) and sent to Robot1 via *TCP Client API* (line 13).

In the above four experiments, we adopt the GNU Radio Benchmark communication protocol. In the following two experiments, we adopt srsRAN (and mmWave) as the communication protocol.

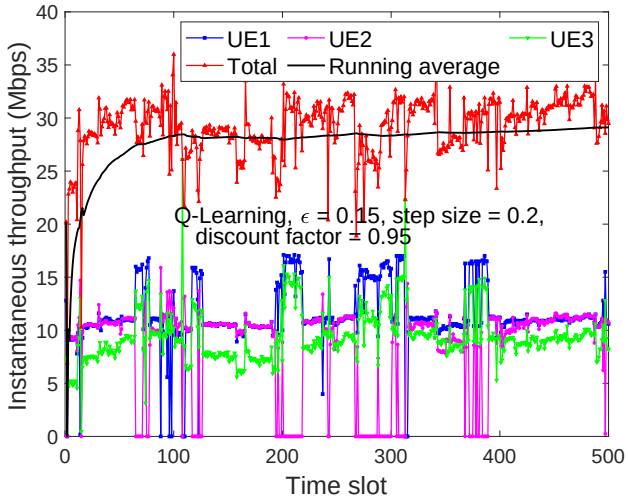


Figure 4.21: Single mobile hotspot scenario: instantaneous and running average of throughput.

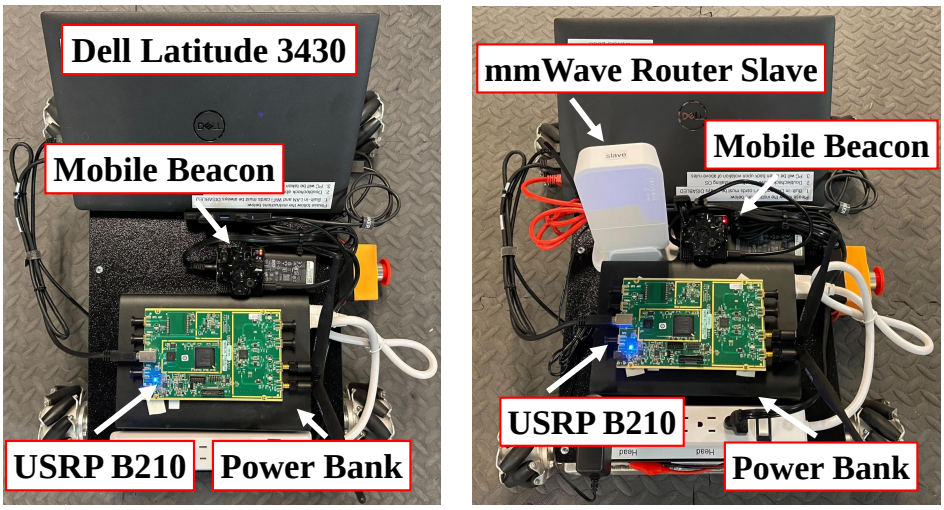
4.4.5 Experiment 5: Mobile Hotspot Navigation with srsRAN

Similar to *Mobile Hotspot Navigation*, we want to maximize the users' aggregate throughput by controlling a robot's trajectory but with a different communication protocol, namely, srsRAN. The robot carries a USRP B210 which works as a base station to serve three users (each being a USRP B210). The three USRP B210s are located at the position of USRP0, USRP5 and USRP14 as shown in Figure 4.5(b), respectively. In each time slot, we use *iperf3* to measure instantaneous throughput for 3 seconds and calculate the corresponding average throughput. The results are shown in Figure 4.21. It can be seen that the total running average converges to around 28 Mbps. The drop of instantaneous throughput of UE2 near the 200th time slot results from it losing its connection and thus leads to increased throughputs of UE1 and UE3. We can also find that the three UEs can achieve similar throughput if no connections are lost. This is because we set the srsRAN MAC layer scheduling mechanism to proportional fair (PF), which aims to balance system throughput and fairness. Based on the PF scheduling mechanism, UE's with relatively better instantaneous channel quality indicator (CQI) compared to their historic average rates will be allocated with more resources. Experimenters can also choose a round-robin scheduling method by specifying it in *Network Configuration API* at the MAC layer.

```
1 import os
2 import Host_Configuration_API as hcfg
3 import srsUE_configuration_API as suea
4
5 ue_name = suea.get_srsRANue_name()
6 ue_port = getattr(hcfg, ue_name).get("port")
7 iperf3_server_cmd = suea.cmd_gen(ue_port)
8 os.system(iperf3_server_cmd)
```

Listing 4.8: Example of starting *iperf3* server

Listing 4.8 shows how to generate an *iperf3* server on srsUE side. Firstly, users need to obtain the UE name (line 5) and the predefined port number (line 6). Then *iperf3* server command is generated (line 7) and executed (line 8) to run the *iperf3* server, which is waiting for *iperf3* client connection from the client side (i.e., the robot).



(a) (b)

Figure 4.22: (a) Snapshot of srsRAN based robot; and (b) snapshot of IAB based robot.

4.4.6 Experiment 6: Mobile Hotspot Navigation in IAB

The sixth experiment is mobile hotspot navigation in an integrated access and backhaul (IAB) network setting. As shown in the Figure 4.22(b), a robot carries a USRP B210 and a

mmWave router slave as a relay to bridge three users and the base station (mmWave router primary). The three users are located at the positions of USRP0, USRP5 and USRP19 and the mmWave primary is located at (7.1 m, 0 m, 1.5 m) in the RoboNet network. Q-learning is adopted to optimize the robot trajectory. The results of the experiments are shown in Figure 4.23. Similarly, the running average throughput converges to 22 Mbps.

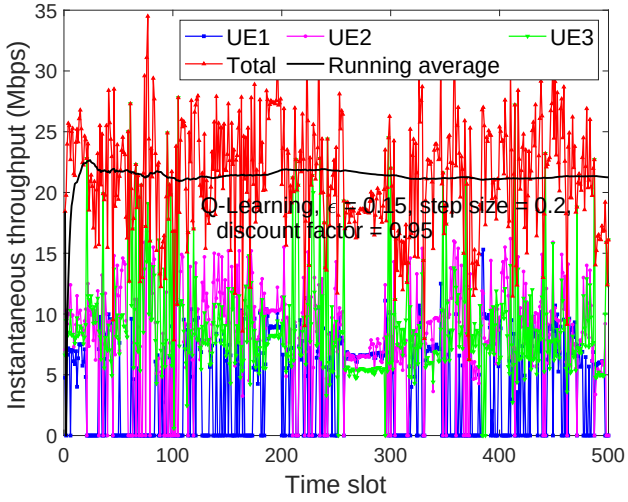


Figure 4.23: Single mobile hotspot scenario in IAB setting: instantaneous and running average of throughput.

In the above experiments, the RL algorithms are adopted to improve network performance. However, we found that sometimes RL does not work well in practice. Randomness (like the user disconnection, time-varying channel, movement of robot and so on) could affect the RL performance. For example, during the mobile hotspot navigation in IAB experiments, the robot spends most time in a state (state 6) after 200 time-slots but the running average throughput is lower as shown in Figure 4.23. This could be due to the laptop's limited computation capabilities, which can degrade the USRP B210s performance. In this case, if we want to apply RL in a wireless network, we need to take randomness into consideration and design a more sophisticated reward (not simply taking throughput as the only one criteria).

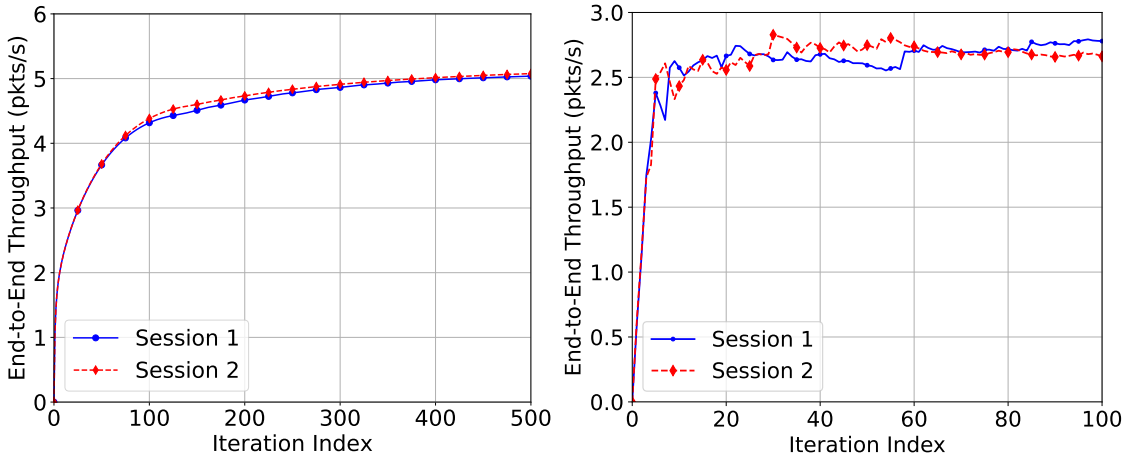


Figure 4.24: Multi-hop network optimization scenario: Average end-to-end throughput with (a) *UBSim* simulator and (b) *NeXT* testbed.

4.4.7 Experiment 7: Multi-hop Network Optimization

In the seventh and final experiment, we consider a multi-session multi-hop network with two sessions and eight nodes. Each session consists of four nodes, namely one source node, two relay nodes and one destination node. The objective is to maximize the network throughput while minimizing the interference between the two sessions by jointly optimizing the physical and transport layers. The optimization algorithms are generated automatically by WNOS, which has been deployed over the control plane of *NeXT*, as described in Chapter 4.2. The resulting algorithms are deployed over the data plane. Similar to the *User Scheduling* experiment discussed above, we conduct this experiment over both UBSim and RoboNet. The results are reported in Figure 4.24. We can see that the control algorithms converge over both UBSim and RoboNet. It is worth pointing out that different link models have been considered in UBSim and RoboNet in their current implementations. In future research, we will create a digital twin of RoboNet based on UBSim and test the gap between simulated and real-world performance.

4.4.8 Scaling Out srsRAN

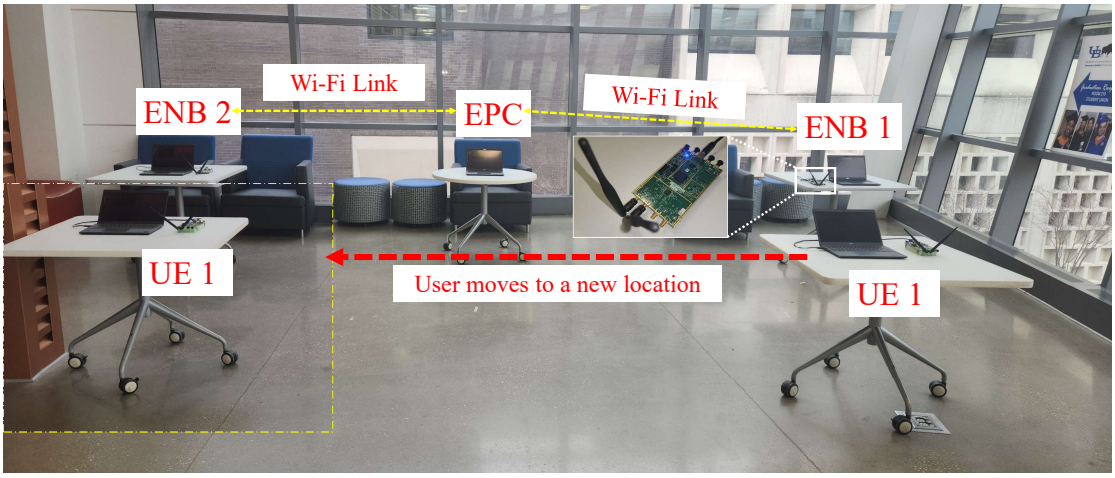
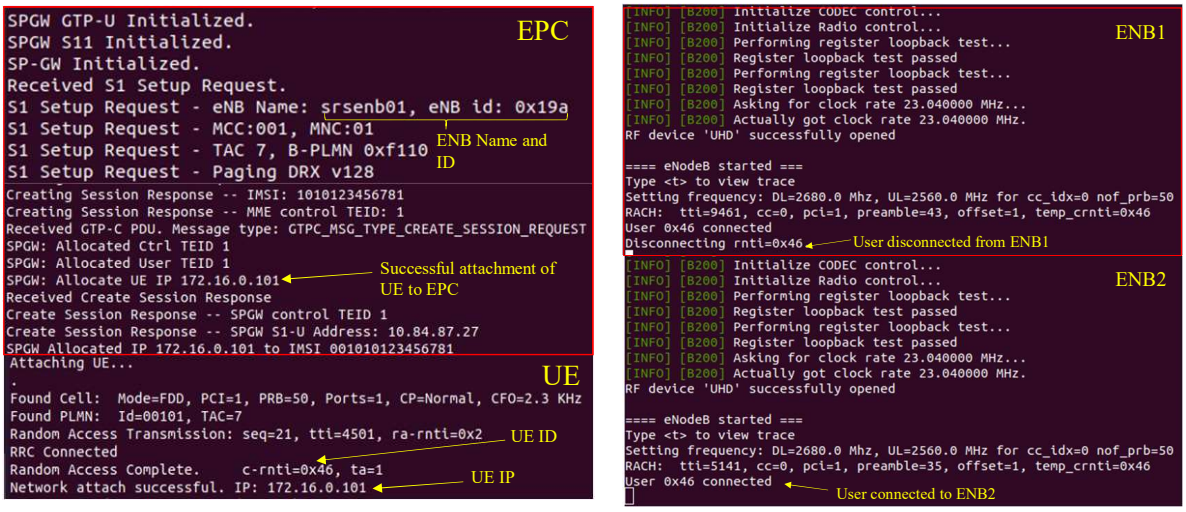


Figure 4.25: Snapshot of the testbed deployed in *Salvador Lounge* in Davis Hall at University at Buffalo.

To verify the effectiveness of the wireless interface between srsENB and srsEPC, we develop a software-defined radio based testbed. The testbed consists of three Universal Software Radio Peripheral (USRP) B210 for 2 srsENBs and 1 srsUE. Each USRP B210 is controlled by a dedicated Dell Latitude 3430 laptop. Another Dell laptop is used for srsEPC. All the hosts run Ubuntu 20.04 LTS. The Downlink E-UTRA Absolute Radio Frequency Channel Number (`dl_earfcn`) is set to 3350, corresponding to 2680 MHz for downlink and 2560 MHz for uplink. A snapshot of the SDR testbed is shown in Figure 4.25. In the demonstration we will show the effectiveness and scalability of the new srsRAN architecture considering two scenarios: (i) srsUE-srsENB connection establishment; (ii) srsUE handover between two srsENBs wirelessly interfaced with the same srsEPC.

Connection Establishment: We first demonstrate the establishment of the wireless link between srsUE, srsENB and srsEPC. After the initial configuration, srsEPC and srsENB are started first by executing `sudo srsepc` and `sudo srsenb`, respectively. Then, srsUE is started with command `sudo srsue`. As shown in Figure 4.26(a), the srsUE can be successfully attached to the network, verifying the effectiveness of the wireless interface between srsENB and srsEPC. *It is worth noting that in this experiment the srsENB first establishes*



(a) (b)

Figure 4.26: Terminal screenshots for (a) link establishment and (b) scalability and handover testing with a wireless interface between srsENB and srsEPC.

a wireless link to the srsEPC (rather than wired connections in traditional srsRAN deployment).

srsUE Handover: We further demonstrate the scalability of the new srsRAN architecture considering two srsENBs wirelessly connected to the same srsEPC. The second srsENB is configured similarly to the first. Additionally, the two srsENBs are deployed sufficiently far away from each other thereby creating no overlap in their coverage areas. In this scenario, we first let the srsUE connect to the first srsENB, i.e., ENB 1 in Figure 4.5. The corresponding terminal output is shown in Figure 4.26(b). We then move the srsUE from the coverage area of ENB 1 to ENB 2, as shown in Figure 4.5 with a dashed red line. The corresponding terminal output is also shown in Figure 4.26(b). It can be seen that the user with ID 0x46 is connected first to ENB 1. When it moves outside the coverage area of ENB 1, it first gets disconnected from ENB 1 and then connects to srsENB 2 automatically. *Notice again that the two srsENBs are wirelessly interfaced with the srsEPC, which makes it easier to deploy a large number of ENBs over possibly mobile hotspots.*

4.5 Enabled New Research

NeXT can enable a wide set of experiments, including *sim-to-real transfer learning*, *robust wireless network control*, *online digital twin construction and optimization*, and *multi-agent reinforcement learning*.

Sim-to-real transfer learning: Towards zero-touch wireless network self-configuration, the proposed framework will connect accelerated learning in the virtual domain with performance evaluation in the real domain. With the proposed framework, novel machine learning algorithms can be designed and tested rapidly in the virtual domain in a variety of configurable networking scenarios, and the converged algorithms can be deployed on SDR hardware for practical evaluation. Making use of a digital twin for initial policy iteration can significantly reduce the time required to generate an optimal control policy, especially in the case of deep learning or deep reinforcement learning. These transfer learning experiments will be used to understand the performance discrepancy between simulation and hardware evaluation, which will be necessary for designing repeatable experiments towards accelerated learning for wireless network self-configuration. This investigation into efficient transfer learning will start with experimental benchmarks to quantify the reality gap between UBSim and RoboNet and then designing methods to minimize the impact of this gap through an experimental campaign of domain adaptation and novel twin-domain learning algorithms.

Robust wireless network control: The use of robust learning for domain adaptation in the wireless domain has been introduced in [93]. By introducing noise to the training data or training environment during policy iteration, it has been shown that the resulting control policy will provide improved performance when faced with unexpected observations or perturbations compared to a non-robust policy. In this line of research, this uncertainty can be interpreted as the set of all physical phenomena which contribute to the performance gap between simulation and hardware scenarios, such as unpredictable RF interference or hardware nonlinearities. The programmable SDR hardware provided by the RoboNet testbed coupled with the virtualization of the RoboNet environment in UBSim enables investigation

into robust learning to improve sim-to-real transfer learning performance in a wide variety of networking scenarios, with or without knowledge of the reality gap. We plan to build on findings in [93] by applying the experimental robust learning framework to the sim-to-real capabilities presented in this work, exploring robust learning as a method of mitigating performance degradation due to sim-to-real policy transfer.

Online digital twin construction and optimization: Existing methods for generating a virtual model for digital twin applications typically rely on human expertise, and can be tedious and error-prone [105, 106]. This motivates autonomous virtual environment construction based on mobile sensing techniques such as simultaneous localization and mapping (SLAM). Using SLAM with remote-control hardware such as the robots introduced in Section 4.1, it is possible to record observations and generate a 3D environment map with configurable fidelity in real time without significant human intervention. This capability can significantly accelerate the digital twin construction process by automating the collection and import of environmental data into the desired simulation environment, such as UBSim. With integrated simulation and experimentation capabilities, the *NeXT* testbed can enable research of online digital twin construction by providing configurable network simulation environments in UBSim, and verifying the accuracy of the autonomously generated digital twin with ground truth obtained through testbed experiments.

Multi-agent Reinforcement Learning (MARL): The *NeXT* testbed can support MARL research for development and evaluation of algorithms such as REINFORCE policy gradient (PG) [107], gradient-based partially observable MDP (G(PO)MDP) [108], actor-critic (A2C) [109], or asynchronous actor-critic (A3C) [110]. In general, these algorithms require significantly more time to converge to an optimal policy than their single-agent counterparts. Additionally, debugging MARL algorithms can be complicated due to the distributed nature of data collection and processing [111]. The architecture of UBSim and its supporting APIs can significantly simplify the simulation design process by streamlining user-configurable parameters such as the number of nodes, distributed or centralized control algorithms, and

reward function related to the environment. This can save time, provide configurable online feedback to display only target data points, and limit redundancy in coding for large-scale MARL problems. Finally, the configurable SDR topology and the hardware available in the RoboNet testbed can provide a framework through which simulation results obtained in the virtual digital twin environment can be verified through real-world experiments.

Chapter 5

Conclusions

In this dissertation, we explored the diverse applications of UAVs and examined spectrum sharing technologies aiming at alleviating the strain on spectrum resources. We presented a comprehensive review of existing experimental platforms designed for conducting experiments in this field. Moreover, we delved into the specific challenges associated with spectrum sharing in the 6 GHz band and the development of testing frameworks. A particular emphasis was placed on enabling UAV operations within the 6 GHz band, alongside the development of a robust testing framework to support these endeavors. To this end, we proposed a new framework called *SwarmShare* to enable spectrum sharing between the incumbent systems and the coexisting UAV networks in the 6 GHz band. We validated the effectiveness of the framework through an extensive simulation campaign. *SwarmShare* is shown to be mobility-resilient and hence is suitable for the operations of moving vehicles such as cars and UAVs on this newly opened spectrum band *without* requiring pre-defined exclusion zones. It is also found that the aggregate interference of the UAVs does not follow any existing distributions. We also introduced the software-defined testbed *NeXT*, which enables integrated simulation, experimentation and optimization for wireless research. We designed the data plane with both the simulator *UBSim* and the testing facility *RoboNet*. We designed the control plane in which a software toolchain is developed to support both traditional model-based and new

data-driven control techniques. We presented the communication protocols deployed on our testbed. We verified the effectiveness and flexibility of *NeXT* considering both simulation and testbed experiments.

Bibliography

[1] Z. Chu, W. Hao, P. Xiao, and J. Shi, "UAV Assisted Spectrum Sharing Ultra-Reliable and Low-Latency Communications," in *Proc. of IEEE GLOBECOM*, Waikoloa, USA, Dec. 2019.

[2] B. Shang, L. Liu, R. M. Rao, V. Marojevic, and J. H. Reed, "3D Spectrum Sharing for Hybrid D2D and UAV Networks," *IEEE Transactions on Communications*, vol. 68, no. 9, pp. 5375–5389, May 2020.

[3] S. K. Moorthy, N. Mastronarde, S. Pudlewski, E. S. Bentley, and Z. Guan, "Swarm UAV Networking With Collaborative Beamforming and Automated ESN Learning in the Presence of Unknown Blockages," *Computer Networks (Elsevier)*, vol. 231, 2023.

[4] S. K. Moorthy and Z. Guan, "Beam Learning in MmWave/THz-band Drone Networks Under In-Flight Mobility Uncertainties," *IEEE Transactions on Mobile Computing*, vol. 21, no. 6, pp. 1945–1957, June 2022.

[5] S. K. Moorthy, M. McManus, and Z. Guan, "ESN Reinforcement Learning for Spectrum and Flight Control in THz-Enabled Drone Networks," *IEEE/ACM Transactions on Networking*, vol. 30, no. 2, pp. 782–795, April 2022.

[6] Z. Guan, N. Cen, T. Melodia, and S. Pudlewski, "Joint Power, Association and Flight Control for Massive-MIMO Self-Organizing Flying Drones," *IEEE/ACM Trans. on Netw.*, vol. 28, no. 4, pp. 1491–1505, August 2020.

[7] S. P. E. S. B. S. K. Moorthy, Z. Guan, "FlyBeam: Echo State Learning for Joint Flight and Beamforming Control in Wireless UAV Networks," in *Proc. of IEEE International Conference on Communications (ICC)*, Virtual/Montreal, Canada, June 2021.

[8] S. K. Moorthy and Z. Guan, "LeTera: Stochastic Beam Control Through ESN Learning in Terahertz-Band Wireless UAV Networks," in *Proc. of IEEE INFOCOM Workshop on Wireless Communications and Networking in Extreme Environments (WCNEE)*, Toronto, Canada, July 2020.

[9] Z. Liu, G. Huang, Q. Zhong, H. Zheng, and S. Zhao, "UAV-Aided Vehicular Communication Design With Vehicle Trajectory's Prediction," *IEEE Wireless Communications Letters*, vol. 10, no. 6, pp. 1212–1216, June 2021.

[10] R. Bautista-Montesano, R. Galluzzi, K. Ruan, Y. Fu, and X. Di, "Autonomous navigation at unsignalized intersections: A coupled reinforcement learning and model predictive control approach," *Transportation research part C: emerging technologies*, vol. 139, p. 103662, June 2022.

[11] Z. Wang, L. Duan, and R. Zhang, "Adaptive Deployment for UAV-Aided Communication Networks," *IEEE Transactions on Wireless Communications*, vol. 18, no. 9, pp. 4531–4543, Sept. 2019.

[12] T. Do-Duy, L. D. Nguyen, T. Q. Duong, S. R. Khosravirad, and H. Claussen, "Joint Optimisation of Real-Time Deployment and Resource Allocation for UAV-Aided Disaster Emergency Communications," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 11, pp. 3411–3424, Nov. 2021.

[13] "Unlicensed Use of 6 GHz Band," in *Notice of Proposed Rulemaking ET Docket No.18-295; GN Docket No. 17-183*, Oct. 2018.

[14] C. Zhan, H. Gupta, A. Bhattacharya, and M. Ghaderibaneh, "Efficient Localization of Multiple Intruders in Shared Spectrum System," in *Proc. of ACM/IEEE International*

Conference on Information Processing in Sensor Networks (IPSN), Sydney, Australia, 2020.

[15] T. Zhao, N. Wang, G. Cao, S. Mao, and X. Wang, "Functional Data Analysis Assisted Cross-Domain Wi-Fi Sensing Using Few-Shot Learning," in *Proc. of IEEE International Conference on Communications*, Denver, USA, June 2024.

[16] M. Ghaderibaneh, C. Zhan, and H. Gupta, "DeepAlloc: Deep Learning Approach to Spectrum Allocation in Shared Spectrum Systems," *IEEE Access*, vol. 12, pp. 8432–8448, January 2024.

[17] Z. Guan, G. E. Santagati, and T. Melodia, "Distributed Algorithms for Joint Channel Access and Rate Control in Ultrasonic Intra-body Networks," *IEEE/ACM Transactions on Networking*, vol. 24, no. 5, pp. 3109–3122, Oct. 2016.

[18] Z. Guan, D. Yuan, H. Zhang, and L. Ding, "Cooperative bargaining solution for efficient and fair spectrum management in cognitive wireless networks," *International Journal of Communication Systems*, vol. 27, no. 11, p. 3441–3459, Nov. 2014.

[19] Z. Guan and T. Melodia, "U-LTE: Spectrally-Efficient and Fair Coexistence Between LTE and Wi-Fi in Unlicensed Bands," in *Proc. of IEEE Conference on Computer Communications (INFOCOM)*, San Francisco, CA, USA, April 2016.

[20] Z. Guan, G. E. Santagati, and T. Melodia, "Ultrasonic Intra-body Networking: Interference Modeling, Stochastic Channel Access and Rate Control," in *Proc. of IEEE Conference on Computer Communications (INFOCOM)*, Hong Kong S.A.R., PRC, April 2015.

[21] V. Sathya, M. I. Rochman, M. Ghosh, and S. Roy, "Standardization Advances for Cellular and Wi-Fi Coexistence in the Unlicensed 5 and 6 GHz Bands," *GetMobile*, vol. 24, no. 1, pp. 5–15, March 2020.

[22] A. Gosain, "Platforms for Advanced Wireless Research: Helping Define a New Edge Computing Paradigm," in *Proceedings of the 2018 on Technologies for the Wireless Edge Workshop*, New Delhi, India, November 2018.

[23] J. Breen, A. Buffmire, J. Duerig, K. Dutt, E. Eide, M. Hibler, D. Johnson, S. K. Kasera, E. Lewis, D. Maas, A. Orange, N. Patwari, D. Reading, R. Ricci, D. Schurig, L. B. Stoller, J. Van der Merwe, K. Webb, and G. Wong, "POWDER: Platform for Open Wireless Data-Driven Experimental Research," in *Proceedings of the 14th International Workshop on Wireless Network Testbeds, Experimental Evaluation Characterization*, London, United Kingdom, Sept. 2020.

[24] D. Raychaudhuri, I. Seskar, G. Zussman, T. Korakis, D. Kilper, T. Chen, J. Kolodziejski, M. Sherman, Z. Kostic, X. Gu, H. Krishnaswamy, S. Maheshwari, P. Skrimponis, and C. Gutterman, "Challenge: COSMOS: A City-Scale Programmable Testbed for Experimentation with Advanced Wireless," in *Proc. of the 26th Annual International Conference on Mobile Computing and Networking*, London, United Kingdom, Sept. 2020.

[25] M. L. Sichitiu, I. Guvenc, R. Dutta, V. Marojevic, and B. Floyd, "AERPAW Emulation Overview," in *Proc. of the 14th International Workshop on Wireless Network Testbeds, Experimental Evaluation Characterization*, London, United Kingdom, September 2020.

[26] H. Zhang, Y. Guan, A. Kamal, D. Qiao, M. Zheng, A. Arora, O. Boyraz, B. Cox, T. Daniels, M. Darr, D. Jacobson, A. Khokhar, S. Kim, J. Koltes, J. Liu, M. Luby, L. Nadolny, J. Peschel, P. Schnable, A. Sharma, A. Somani, and L. Tang, "ARA: A Wireless Living Lab Vision for Smart and Connected Rural Communities," in *Proceedings of the 15th ACM Workshop on Wireless Network Testbeds, Experimental Evaluation & CHaracterization*, New Orleans, LA, USA, January 2021, p. 9–16.

[27] L. Bonati, S. D'Oro, M. Polese, S. Basagni, and T. Melodia, "Intelligence and Learning in O-RAN for Data-Driven NextG Cellular Networks," *IEEE Communications Magazine*, vol. 59, no. 10, pp. 21–27, Oct. 2021.

[28] J. Hu, S. K. Moorthy, A. Harindranath, Z. Guan, N. Mastronarde, E. S. Bentley, and S. Pudlewski, "SwarmShare: Mobility-Resilient Spectrum Sharing for Swarm UAV Networking in the 6 GHz Band," in *Proc. of IEEE SECON*, Virtual Conference, July 2021.

[29] J. Yuan, H. Nicolai, and V. Isler, "Multi-step Recurrent Q-learning for Robotic Velcro Peeling," in *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, Xi'an, China, May 2021.

[30] K. Ruan, J. Zhang, X. Di, and E. Bareinboim, "Causal Imitation Learning via Inverse Reinforcement Learning," in *Proc. of International Conference on Learning Representations*, Kigali, Rwanda, May 2023.

[31] Z. Shi, S. He, J. Sun, T. Chen, J. Chen, and H. Dong, "An Efficient Multi-Task Network for Pedestrian Intrusion Detection," *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 1, pp. 649–660, April 2023.

[32] Y. Mi, D. Mohaisen, and A. Wang, "AutoDefense: Reinforcement Learning Based Autoreactive Defense Against Network Attacks," in *2022 IEEE Conference on Communications and Network Security (CNS)*, Austin, TX, USA, November 2022.

[33] N. Pang, L. Qian, W. Lyu, and J.-D. Yang, "Transfer Learning for Scientific Data Chain Extraction in Small Chemical Corpus with BERT-CRF Model," *arXiv preprint arXiv:1905.05615*, 2019.

[34] K. Ruan and X. Di, "Learning Human Driving Behaviors with Sequential Causal Imitation Learning," in *Proc. of the AAAI Conference on Artificial Intelligence*, Vancouver, Canada, June 2022.

[35] F. Wen, M. Qin, P. Gratz, and N. Reddy, "Software Hint-Driven Data Management for Hybrid Memory in Mobile Systems," *ACM Transactions on Embedded Computing Systems*, vol. 21, no. 1, pp. 1–18, January 2022.

[36] F. Tian, Y. Zhang, W. Ye, C. Jin, Z. Wu, and Z.-L. Zhang, "Accelerating Distributed Deep Learning Using Multi-Path RDMA in Data Center Networks," in *Proceedings of the ACM SIGCOMM Symposium on SDN Research (SOSR)*, Virtual Event, USA, October 2021.

[37] Y. Wei, M. Gao, J. Xiao, C. Liu, Y. Tian, and Y. He, "Research and Implementation of Traffic Sign Recognition Algorithm Model Based on Machine Learning," *Journal of Software Engineering and Applications*, vol. 16, no. 6, pp. 193–210, June 2023.

[38] J. Hu, M. McManus, S. K. Moorthy, Y. Cui, Z. Guan, N. Mastronarde, E. S. Bentley, and M. Medley, "NeXT: A Software-Defined Testbed with Integrated Optimization, Simulation and Experimentation," in *Proc. of IEEE Future Networks World Forum (FNWF): WS5: Federated Testbed as a Service for Future Networks: Challenges the State of the Art*, Montreal, Canada, October 2022.

[39] J. Hu, Z. Zhao, M. McManus, S. K. Moorthy, Y. Cui, N. Mastronarde, E. S. Bentley, M. Medley, and Z. Guan, "NeXT: Architecture, Prototyping and Measurement of a Software-Defined Testing Framework for Integrated RF Network Simulation, Experimentation and Optimization," *Elsevier Journal of Computer Communications*, October 2023.

[40] Z. Guan, L. Bertizzolo, E. Demirors, and T. Melodia, "WNOS: Enabling Principled Software-Defined Wireless Networking," *IEEE/ACM Transactions on Networking*, vol. 29, no. 3, pp. 1391–1407, June 2021.

[41] B. Wang, Y. Sun, Z. Sun, L. D. Nguyen, and T. Q. Duong, "UAV-Assisted Emergency Communications in Social IoT: A Dynamic Hypergraph Coloring Approach," *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 7663–7677, April 2020.

[42] J. Lyu, Y. Zeng, and R. Zhang, "Spectrum Sharing and Cyclical Multiple Access in UAV-Aided Cellular Offloading," in *Proc. of IEEE GLOBECOM*, Singapore, January 2018.

[43] H. Wang, J. Wang, G. Ding, J. Chen, Y. Li, and Z. Han, "Spectrum Sharing Planning for Full-Duplex UAV Relaying Systems With Underlaid D2D Communications," *IEEE JSAC*, vol. 36, no. 9, pp. 1986–1999, August 2018.

[44] T. Q. Duong, L. D. Nguyen, H. D. Tuan, and L. Hanzo, "Learning-Aided Realtime Performance Optimisation of Cognitive UAV-Assisted Disaster Communication," in *Proc. of IEEE GLOBECOM*, Waikoloa, USA, February 2019.

[45] B. Ji, Y. Li, D. Cao, C. Li, S. Mumtaz, and D. Wang, "Secrecy Performance Analysis of UAV Assisted Relay Transmission for Cognitive Network with Energy Harvesting," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 7, pp. 7404–7415, July 2020.

[46] A. Fotouhi, H. Qiang, M. Ding, M. Hassan, L. G. Giordano, A. Garcia-Rodriguez, and J. Yuan, "Survey on UAV Cellular Communications: Practical Aspects, Standardization Advancements, Regulation, and Security Challenges," *IEEE Communications Surveys Tutorials*, vol. 21, no. 4, pp. 3417–3442, March 2019.

[47] Z. Ullah, F. Al-Turjman, and L. Mostarda, "Cognition in UAV-Aided 5G and Beyond Communications: A Survey," *IEEE Transactions on Cognitive Communications and Networking*, vol. 6, no. 3, pp. 872–891, September 2020.

[48] Y. Zhou, F. Zhou, H. Zhou, D. W. K. Ng, and R. Q. Hu, "Robust Trajectory and Transmit Power Optimization for Secure UAV-Enabled Cognitive Radio Networks," *IEEE Transactions on Communications*, vol. 68, no. 7, pp. 4022–4034, July 2020.

[49] Jie Xiang, Yan Zhang, and Tor Skeie, "Joint Admission and Power Control for Cognitive Radio Cellular Networks," in *Proc. of IEEE Singapore Int'l. Conference on Communication Systems*, Guangzhou, China, Jan. 2008.

[50] H. Guo, R. Long, and Y. Liang, "Cognitive Backscatter Network: A Spectrum Sharing Paradigm for Passive IoT," *IEEE Wireless Communications Letters*, vol. 8, no. 5, pp. 1423–1426, Oct. 2019.

[51] P. K. Sangdeh, H. Pirayesh, A. Quadri, and H. Zeng, "A Practical Spectrum Sharing Scheme for Cognitive Radio Networks: Design and Experiments," *IEEE/ACM Transactions on Networking*, vol. 28, no. 4, pp. 1818–1831, August 2020.

[52] H. Zhang, N. Yang, W. Huangfu, K. Long, and V. C. Leung, "Power Control Based on Deep Reinforcement Learning for Spectrum Sharing," *IEEE Transactions on Wireless Communications*, vol. 19, no. 6, pp. 4209–4219, June 2020.

[53] L. Li, L. Liu, J. Bai, H.-H. Chang, H. Chen, J. D. Ashdown, J. Zhang, and Y. Yi, "Accelerating Model-Free Reinforcement Learning With Imperfect Model Knowledge in Dynamic Spectrum Access," *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 7517–7528, August 2020.

[54] M. Labib, A. F. Martone, V. Marojevic, J. H. Reed, and A. I. Zaghloul, "A Stochastic Optimization Approach for Spectrum Sharing of Radar and LTE Systems," *IEEE Access*, vol. 7, pp. 60 814–60 826, April 2019.

[55] C. Aydogdu *et al.*, "RadChat: Spectrum Sharing for Automotive Radar Interference Mitigation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 1, pp. 416–429, December 2021.

[56] A. F. Martone, K. A. Gallagher, and K. D. Sherbondy, "Joint Radar and Communication System Optimization for Spectrum Sharing," in *Proc. of IEEE Radar Conference (RadarConf)*, Boston, USA, April 2019.

[57] C. Zhang, C. Jiang, L. Kuang, J. Jin, Y. He, and Z. Han, "Spatial Spectrum Sharing for Satellite and Terrestrial Communication Networks," *IEEE Trans. on Aerospace and Electronic Systems*, vol. 55, no. 3, pp. 1075–1089, June 2019.

[58] M. L. Attiah, A. A. M. Isa, Z. Zakaria, M. Abdulhameed, M. K. Mohsen, and I. Ali, "A Survey of mmWave User Association Mechanisms and Spectrum Sharing Approaches: An Overview, Open Issues and Challenges, Future Research Trends," *Wireless Networks*, vol. 26, no. 4, pp. 2487–2514, 2020.

[59] D. Tarek, A. Benslimane, M. Darwish, and A. M. Kotb, "Survey on Spectrum Sharing/Allocation for Cognitive Radio Networks Internet of Things," *Egyptian Informatics Journal*, vol. 21, no. 4, pp. 231–239, December 2020.

[60] L. Bonati, S. D'Oro, S. Basagni, and T. Melodia, "SCOPE: An Open and Softwarized Prototyping Platform for NextG Systems," in *Proceedings of the 19th Annual International Conference on Mobile Systems, Applications, and Services*, Wisconsin, USA, June 2021.

[61] R. Doost-Mohammady, O. Bejarano, L. Zhong, J. R. Cavallaro, E. Knightly, Z. M. Mao, W. W. Li, X. Chen, and A. Sabharwal, "RENEW: Programmable and Observable Massive MIMO Networks," in *2018 52nd Asilomar Conference on Signals, Systems, and Computers*, Pacific Grove, USA, October 2018, pp. 1654–1658.

[62] I. Baldin, A. Nikolich, J. Griffioen, I. I. S. Monga, K.-C. Wang, T. Lehman, and P. Ruth, "FABRIC: A National-Scale Programmable Experimental Network Infrastructure," *IEEE Internet Computing*, vol. 23, no. 6, pp. 38–47, November 2019.

[63] J. Mirkovic and T. Benzel, "Deterlab Testbed for Cybersecurity Research and Education," *Journal of Computing Sciences in Colleges*, vol. 28, no. 4, April 2013.

[64] R. Zhao, T. Woodford, T. Wei, K. Qian, and X. Zhang, "M-cube: A millimeter-wave massive MIMO software radio," in *Proceedings of the 26th Annual International*

Conference on Mobile Computing and Networking, London, United Kingdom, April 2020, pp. 1–14.

[65] S. Wang, J. Huang, and X. Zhang, "Demystifying millimeter-wave V2X: Towards robust and efficient directional connectivity under high mobility," in *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*, London, United Kingdom, April 2020, p. 677–690.

[66] A. Song, X. Hong, F. Zhang, Z. Peng, and Z. Wang, "Mu-Net: Community-shared infrastructure for mobile underwater acoustic networks," *The Journal of the Acoustical Society of America*, vol. 150, no. 4, pp. A197–A198, November 2021.

[67] "NSF Testbeds," https://nets-vo.org/resoures/nsf-testbeds/.

[68] "CISE Community Research Infrastructure," https://www.ccrivo.org/projects/.

[69] L. Bertizzolo, L. Bonati, E. Demirors, and T. Melodia, "Arena: A 64-Antenna SDR-Based Ceiling Grid Testbed for Sub-6 GHz Radio Spectrum Research," in *Proceedings of the 13th International Workshop on Wireless Network Testbeds, Experimental Evaluation Characterization*. Los Cabos, Mexico: Association for Computing Machinery, October 2019, p. 5–12.

[70] R. K. Sheshadri, E. Chai, K. Sundaresan, and S. Rangarajan, "SkyHaul: An Autonomous Gigabit Network Fabric in the Sky," *ArXiv*, vol. abs/2006.11307, 2020.

[71] P. Sen, D. A. Pados, S. N. Batalama, E. Einarsson, J. P. Bird, and J. M. Jornet, "The TeraNova Platform: An Integrated Testbed for Ultra-Broadband Wireless Communications at True Terahertz Frequencies," *Computer Networks*, vol. 179, p. 107370, October 2020.

[72] R. Lim, F. Ferrari, M. Zimmerling, C. Walser, P. Sommer, and J. Beutel, "FlockLab: A testbed for distributed, synchronized tracing and profiling of wireless embedded

systems," in *2013 ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, Philadelphia, USA, April 2013, pp. 153–165.

[73] R. Trüb, R. D. Forno, T. Gsell, J. Beutel, and L. Thiele, "A Testbed for Long-Range LoRa Communication: Demo Abstract," in *Proceedings of the 18th International Conference on Information Processing in Sensor Networks*, Montreal, Canada, April 2019, pp. 342–343.

[74] J. Ren, Y. He, G. Huang, G. Yu, Y. Cai, and Z. Zhang, "An Edge-Computing Based Architecture for Mobile Augmented Reality," *IEEE Network*, vol. 33, no. 4, pp. 162–169, January 2019.

[75] J. Cui, Y. Liu, and A. Nallanathan, "Multi-Agent Reinforcement Learning-Based Resource Allocation for UAV Networks," *IEEE Trans. on Wireless Communications*, vol. 19, no. 2, pp. 729–743, August 2020.

[76] S. K. Moorthy and Z. Guan, "FlyTera: Echo State Learning for Joint Access and Flight Control in THz-enabled Drone Networks," in *Proc. of IEEE International Conference on Sensing, Communication and Networking (SECON)*, Como, Italy, June 2020.

[77] L. Bertizzolo, E. Demirors, Z. Guan, and T. Melodia, "CoBeam: Beamforming-based Spectrum Sharing with Zero Cross-Technology Signaling for 5G Wireless Networks," in *Proc. of IEEE INFOCOM*, Toronto, Canada, July 2020.

[78] Y. Li, H. Zhang, K. Long, S. Choi, and A. Nallanathan, "Resource Allocation for Optimizing Energy Efficiency in NOMA-based Fog UAV Wireless Networks," *IEEE Network*, vol. 34, no. 2, pp. 158–163, September 2020.

[79] N. Vervliet, O. Debals, L. Sorber, and L. De Lathauwer, "Breaking the Curse of Dimensionality Using Decompositions of Incomplete Tensors: Tensor-based Scientific Computing in Big Data Analysis," *IEEE Signal Processing Magazine*, vol. 31, no. 5, pp. 71–79, August 2014.

[80] K. T. Hemachandra and N. C. Beaulieu, "Novel Representations for the Equicorrelated Multivariate Non-Central Chi-Square Distribution and Applications to MIMO Systems in Correlated Rician Fading," *IEEE Trans. on Commun.*, vol. 59, no. 9, pp. 2349–2354, March 2011.

[81] S. Kusaladharma and C. Tellambura, "Aggregate Interference Analysis for Underlay Cognitive Radio Networks," *IEEE Wireless Communications Letters*, vol. 1, no. 6, pp. 641–644, December 2012.

[82] A. A. Khuwaja, Y. Chen, and G. Zheng, "Effect of User Mobility and Channel Fading on the Outage Performance of UAV Communications," *IEEE Wireless Commun. Letters*, vol. 9, no. 3, pp. 367–370, March 2020.

[83] A. F. Schmidt and C. Finan, "Linear Regression and the Normality Assumption," *Journal of clinical epidemiology*, vol. 98, pp. 146–151, June 2018.

[84] Z. Xie, H. Wang, S. Han, E. Schoenfeld, and F. Ye, "DeepVS: a deep learning approach for RF-based vital signs sensing," in *Proc. of the 13th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics*, Northbrook, Illinois, August 2022.

[85] D. Zhang, F. Zhou, Y. Jiang, and Z. Fu, "Mm-bsn: Self-Supervised Image Denoising for Real-World with Multi-Mask based on Blind-Spot Network," in *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Vancouver, Canada, June 2023.

[86] S. A. Al-Ahmed, M. Z. Shakir, and S. A. R. Zaidi, "Optimal 3D UAV Base Station Placement by Considering Autonomous Coverage Hole Detection, Wireless Backhaul and User Demand," *Journal of Commun. and Networks*, vol. 22, no. 6, pp. 467–475, December 2020.

[87] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. Cambridge, MA, USA: MIT Press, 2018.

[88] D. W. Hosmer Jr, S. Lemeshow, and R. X. Sturdivant, *Applied Logistic Regression*. USA: John Wiley & Sons, 2013.

[89] "SimPy." [Online]. Available: https://pypi.org/project/simpy/

[90] Z. Guan, T. Melodia, and G. Scutari, "To Transmit or Not to Transmit? Distributed Queueing Games for Infrastructureless Wireless Networks," *IEEE/ACM Trans. on Netw.*, vol. 24, no. 2, pp. 1153–1166, April 2016.

[91] R. K. Ganti and M. Haenggi, "Interference in Ad Hoc Networks with General Motion-Invariant Node Distributions," in *Proc. of IEEE Int'l. Symposium on Information Theory*, Toronto, Canada, July 2008.

[92] J. Hu, S. K. Moorthy, A. Harindranath, J. Zhang, Z. Zhao, N. Mastronarde, E. S. Bentley, S. Pudlewski, and Z. Guan, "A Mobility-Resilient Spectrum Sharing Framework for Operating Wireless UAVs in the 6 GHz Band," *IEEE/ACM Transactions on Networking*, vol. 31, no. 6, pp. 3128–3142, December 2023.

[93] M. McManus, Z. Guan, N. Mastronarde, and S. Zou, "On the Source-to-Target Gap of Robust Double Deep Q-Learning in Digital Twin-Enabled Wireless Networks," in *Proc. of SPIE Conference Big Data IV: Learning, Analytics, and Applications*, Orlando, Florida, April 2022.

[94] Y. Jiang, T. Zhang, D. Ho, Y. Bai, C. K. Liu, S. Levine, and J. Tan, "SimGAN: Hybrid Simulator Identification for Domain Adaptation via Adversarial Reinforcement Learning," in *Proceedings of the 2021 IEEE International Conference on Robotics and Automation*, Xi'an, China, May 2021, pp. 2884–2890.

[95] S. Barrachina-Muñoz, B. Bellalta, and E. W. Knightly, "Wi-Fi Channel Bonding: An All-Channel System and Experimental Study From Urban Hotspots to a Sold-Out Stadium," *IEEE/ACM Transactions on Networking*, vol. 29, no. 5, pp. 2101–2114, Oct. 2021.

[96] J. Buczek, L. Bertizzolo, S. Basagni, and T. Melodia, "What is A Wireless UAV? A Design Blueprint for 6G Flying Wireless Nodes," in *Proc. of the 15th ACM Workshop on Wireless Network Testbeds, Experimental evaluation CHaracterization (WiNTECH'21)*, New Orleans, LA, USA, January 2022.

[97] "mmWave Router." [Online]. Available: https://mikrotik.com/product/wap$_$60g

[98] S. K. Moorthy, N. Mastronarde, E. S. Bentley, M. Medley, and Z. Guan, "OSWireless: Hiding the Specification Complexity for Zero-Touch Software-Defined Wireless Networks," *Computer Networks (Elsevier)*, vol. 237, December 2023.

[99] S. K. Moorthy, Z. Guan, N. Mastronarde, E. S. Bentley, and M. Medley, "OSWireless: Enhancing Automation for Optimizing Intent-Driven Software-Defined Wireless Networks," in *Proc. of IEEE International Conference on Mobile Ad-Hoc and Smart Systems (MASS)*, Denver, Colorado, October 2022.

[100] Z. Guan, L. Bertizzolo, E. Demirors, and T. Melodia, "WNOS: An Optimization-based Wireless Network Operating System," in *Proc. of ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, Los Angeles, USA, June 2018.

[101] "GNURadio Benchmark." [Online]. Available: https://github.com/n4hy/gnuradio/tree/master/gr-digital/examples/narrowband

[102] "srsRAN." [Online]. Available: https://www.srslte.com/

[103] G. Sklivanitis, A. Gannon, K. Tountas, D. A. Pados, S. N. Batalama, S. Reichhart, M. Medley, N. Thawdar, U. Lee, J. D. Matyjas, S. Pudlewski, A. Drozd, A. Amanna,

F. Latus, Z. Goldsmith, and D. Diaz, "Airborne Cognitive Networking: Design, Development, and Deployment," *IEEE Access*, vol. 6, pp. 47 217–47 239, July 2018.

[104] N. Mishra, Y. V. Iyengar, A. C. Raikar, N. Thomas, S. K. Moorthy, J. Hu, Z. Zhao, N. Mastronarde, E. S. Bentley, M. Medley, and Z. Guan, "Demo Abstract: Scaling Out srsRAN Through Interfacing Wirelessly srsENB With srsEPC," in *Proc. of IEEE International Conference on Computer Communications (INFOCOM)*, New York area, USA, May 2023.

[105] S. K. Moorthy, A. Harindranath, M. McManus, Z. Guan, N. Mastronarde, E. S. Bentley, and M. Medley, "A Middleware for Digital Twin-Enabled Flying Network Simulations Using UBSim and UB-ANC," in *Proc. of IEEE DCOSS Workshop on Wireless Communications and Networking in Extreme Environments (WCNEE)*, LA, California, June 2022.

[106] M. McManus, Y. Cui, J. Zhang, J. Hu, S. K. Moorthy, N. Mastronarde, E. S. Bentley, M. Medley, and Z. Guan, "Digital Twin-Enabled Domain Adaptation for Zero-Touch UAV Networks: Survey and Challenges," *Elsevier Journal of Computer Networks*, November 2023.

[107] T. Zhang, H. Wen, Y. Jiang, and J. Tang, "Deep Reinforcement Learning Based IRS for Cooperative Jamming Networks under Edge Computing," *IEEE Internet of Things Journal*, January 2023.

[108] R.-T. Ma, Y.-P. Hsu, and K.-T. Feng, "A POMDP-Based Spectrum Handoff Protocol for Partially Observable Cognitive Radio Networks," in *2009 IEEE Wireless Communications and Networking Conference*, Budapest, Hungary, May 2009.

[109] C. Jingdi, W. Yimeng, and L. Tian, "Bringing Fairness to Actor-Critic Reinforcement Learning for Network Utility Optimization," in *Proc. of IEEE Conference on Computer Communications*, Vancouver, Canada, May 2021.

[110] H. Zhou, Z. Wang, H. Zheng, S. He, and M. Dong, "Cost Minimization-Oriented Computation Offloading and Service Caching in Mobile Cloud-Edge Computing: An A3C-based Approach," *IEEE Transactions on Network Science and Engineering*, March 2023.

[111] J. Zhi, Y. Hao, C. Vo, M. Morales, and J.-M. Lien, "Computing 3-D From-Region Visibility Using Visibility Integrity," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4286–4291, IEEE 2019.