Università degli Studi di Padova

Department of Information Engineering

Ph.D. Course in Information Science and Technology

XXXIV Series

Evaluation and Improvements of mmWave 802.11 Models, Protocols, and Systems

Coordinator
Andrea Neviani
University of Padova

Supervisor
Michele Zorzi
University of Padova

Ph.D. Candidate
Mattia Lecci

Academic Year 2021/2022

Abstract

The 5$^{\text{th}}$ Generation (5G) of communication networks is currently being deployed, promising better than ever capacity, responsiveness, and coverage. Many new technologies, as well as evolutions of old technologies, have been harvested to improve over the previous generation, such as the usage of high frequencies commonly known as the Millimeter Wave (mmW) band. These new bands, typically ranging between 6 and 100 GHz, have long been studied, trying to overcome many of their peculiarities such as (i) low range due to high free-space propagation loss, (ii) high susceptibility to blockage, and (iii) sparse directionality, among others.

In this thesis, we analyze and propose models that allow more in-depth studies on next-generation networks on different levels. Aiming to improve the next-generation IEEE 802.11 standards, also known as Wireless Gigabit (WiGig), we focused mainly on full-stack network simulations, given the higher degree of realism with respect to mathematical models, and the much lower cost and flexibility with respect to hardware testbeds. We were able to improve and create models ranging across almost all levels of the communication stack, from the Physical (PHY) up to the Application (APP) layers. This allowed us to obtain a holistic view of the mmW-based network, making us able to design and characterize better models.

Starting from the mmW channel itself, we will describe our proposals to modify well-known channel models to improve simulation performance and extend network analysis to scenarios that were never explored before, due to a lack of available tools. Antenna models were studied, and, with the help of machine learning techniques, optimal configurations specific for the mmW band were obtained. Moving towards the WiGig protocol stack, works have been done on the optimization of Medium Access Control (MAC)-layer scheduling algorithms, specifically tailored for quasi-periodic applications. Finally, we analyzed, characterized, and modeled eXtended Reality (XR) traffic, one of the most prominent types of quasi-periodic applications that are forseen to be largely used in 5G networks.

Sommario

La prossima generazione di reti di comunicationi, conosciuta come 5G, promette di avere velocità e coperture nettamente superiori alle reti 4G. Per ottenere queste prestazioni è stato proposto di utilizzare sia nuove tecnologie che evoluzioni di tecniche già conosciute e studiate in passato. La novità più acclamata è l'uso di frequenze superiori al passato che vanno dai 6 ai 100 GHz, anche chiamate onde millimetriche, o Millimeter Wave (mmW) in inglese. Molti studi degli anni passati si sono focalizzati su questo intervallo di frequenze, cercando di renderle utilizzabili e superando alcune loro peculiarità, tra cui (i) la poca copertura dovuta a perdite di potenza maggiori sulle lunghe distanze rispetto a frequenze più basse, (ii) la loro estrema facilità ad essere bloccate da persone ed oggetti, e (iii) una forte tendenza a concentrare la potenza in poche direzioni circoscritte.

In questa tesi analizziamo e proponiamo modelli che consentono studi più approfonditi sulle reti di nuova generazione a diversi livelli. Con l'obiettivo di migliorare gli standard Wi-Fi di prossima generazione, noti anche come Wireless Gigabit (WiGig), ci siamo concentrati principalmente su simulazioni di rete full-stack, dato il maggior grado di realismo rispetto ai modelli matematici da una parte, e il minor costo unito a una maggiore flessibilità rispetto a soluzioni hardware. Siamo stati in grado di migliorare e creare modelli che spaziano su quasi tutti i livelli dello stack di comunicazione, dal livello fisico (PHY) fino al livello applicazione (APP). Questo ci ha permesso di ottenere una visione completa della rete mmW, rendendoci in grado di progettare e caratterizzare modelli migliori.

Partendo dal canale mmW stesso, descriveremo le nostre proposte per modificare modelli di canale noti per migliorare le prestazioni di simulazione ed estendere l'analisi di rete a scenari mai esplorati prima, a causa della mancanza di strumenti disponibili. Sono stati studiati modelli di antenne e, con l'ausilio di tecniche di machine learning, sono state ottenute configurazioni ottimali specifiche per la banda mmW. Focalizzandosi sugli standard WiGig, sono stati fatti lavori sull'ottimizzazione degli algoritmi di scheduling a livello MAC, pensati appositamente per applicazioni con carattere quasi periodico. Infine, abbiamo analizzato, caratterizzato e modellato il traffico di realtà aumentata e virtuale (conosciuto anche come realtà estesa (XR)), una delle applicazioni quasi-periodiche più importanti che dovrebbero essere ampiamente utilizzate nelle reti 5G.

Contents

Abstract **v**

List of figures **xii**

List of tables **xvii**

Acronyms **xix**

1 Introduction **1**
- 1.1 Simulation of Communication Networks . . . 2
- 1.2 Open Challenges and Research Directions . . . 4
- 1.3 Mathematical Notation . . . 6
- 1.4 Main Contributions . . . 7
- 1.5 Thesis Structure . . . 8

2 Millimeter Wave Channel Modeling **9**
- 2.1 Introduction . . . 9
- 2.2 Channel Modeling at MmWaves . . . 10
  - 2.2.1 Analytical Channel Models . . . 11
  - 2.2.2 Stochastic Spatial Channel Models . . . 11
  - 2.2.3 Deterministic Channel Models . . . 12
  - 2.2.4 Quasi-Deterministic Channel Models . . . 16
  - 2.2.5 Comparison of RT and QD Models with Measurements . . . 20
  - 2.2.6 Blockage in Millimiter Waves . . . 24
- 2.3 Simplification of a Stochastic SCM: 3GPP TR 38.901 . . . 28
  - 2.3.1 An Introduction to 3GPP TR 38.901 . . . 28
  - 2.3.2 Analysis and Profiling . . . 29
  - 2.3.3 Channel Simplification Results . . . 30
- 2.4 Simplification of an RT-based Channel Model . . . 33
  - 2.4.1 Simulation Scenarios . . . 36
  - 2.4.2 Link-Level Performance Results . . . 39
  - 2.4.3 End-to-End Performance Results . . . 42
  - 2.4.4 Computational Performance . . . 46
  - 2.4.5 Design Guidelines . . . 51
- 2.5 Blockage Modeling . . . 52
  - 2.5.1 Framework Overview . . . 52
  - 2.5.2 Example . . . 54
- 2.6 Conclusions . . . 56

3 Antenna Array Modeling and Optimization **59**
- 3.1 Introduction . . . 59
- 3.2 Antenna Optimization: Related Works . . . 61
- 3.3 Antenna and Beamforming Modeling . . . 62
- 3.4 Framework Description . . . 64

3.4.1 Network Simulator 65
3.4.2 Data Analysis and Machine Learning 66
3.4.3 Optimization 68
3.5 Regular UPA Optimization 69
3.5.1 Scenario Description 69
3.5.2 Results 70
3.6 Thinned UPA Optimization 72
3.6.1 Antenna Array Generation 72
3.6.2 Scenario Parameters 73
3.6.3 Learning Methods 73
3.6.4 Optimization Results 74
3.7 Antenna Array Modeling on ns-3 77
3.7.1 ns-3 Simulation Setup 77
3.7.2 Simulation Results 79
3.8 Conclusions 82

4 WiGig MAC Scheduling **85**
4.1 Introduction 85
4.2 State of the Art 87
4.2.1 Available Research Tools 88
4.3 IEEE 802.11ad Overview 89
4.3.1 Beacon Header Interval 89
4.3.2 Data Transmission Interval 90
4.3.3 Scheduling in IEEE 802.11ad 90
4.4 WiGig Scheduling Framework on ns-3 91
4.4.1 Simulation Setup 92
4.4.2 Simulation Results 94
4.5 Mathematical Framework Description 99
4.5.1 Feasibility Check Algorithm 102
4.5.2 Simple Scheduler 103
4.5.3 Max-Min Fair Scheduler 103
4.5.4 Scheduling Performance 106
4.6 Conclusions 110

5 Extended Reality Applications **113**
5.1 Introduction 113
5.2 State of the Art 116
5.3 VR Traffic: Acquisition and Analysis 118
5.3.1 Acquisition Setup 118
5.3.2 Traffic Analysis 119
5.4 Traffic Model 123
5.4.1 Distribution fitting 123
5.4.2 Generative Model 125
5.5 Ns-3 Implementation 126
5.5.1 Bursty Application 127
5.5.2 Burst Generator Interface 127
5.5.3 Burst Sink 129
5.6 Simulation Results 129
5.6.1 Model Validation 130

5.6.2 Use Case Example . . . 132
5.7 XR Traffic Modeling Roadmap . . . 134
5.7.1 Exploiting First-Order Statistics . . . 134
5.7.2 Introducing Temporal Correlation . . . 134
5.7.3 Introducing Head Tracking . . . 135
5.7.4 Full Traffic Emulator . . . 135
5.7.5 QoE-centric XR . . . 136
5.8 Conclusions . . . 136

6 Conclusion **139**

References **142**

List of Publications **161**

Listing of figures

1.1 Elements Involved in the End-to-end Simulation of a Communication Network . 3

2.1 Visualization of the MoI algorithm for a second-order reflection ($r = 2$). 14
2.2 Example of reflection tree for $R = 2$. 15
2.3 Graphical representation of the QD parameters. 17
2.4 CAD model of NIST's lecture room. 22
2.5 Example of comparison between measurements and ray-tracer. 23
2.6 Reduced multiple reflection QD model applied to RT-based channel traces with up to 2$^{\text{nd}}$ order reflections. Rays with path gain below -120 dB are not shown, to more closely resemble the dynamic range of the channel sounder. 24
2.7 Comparison between CDFs of MPC path gain, absolute delay, and RMS delay spread with and without QD model with respect to the measurements. 25
2.8 Results from the profiling of the 3GPP channel model described in 3GPP TR 38.901, for different square antenna arrays at the transmitter and the receiver. . 30
2.9 Computation time required to generate an instance of the channel matrix (a) and performance gain introduced by the simplification (b), as a function of the number of antenna elements at the gNB, for different configurations at the UE and different combinations of simplification parameters for the channel. 32
2.10 Cumulative Distribution and Probability Density Functions of the narrowband SINR Γ of a scenario composed of UEs and gNBs with 16 and 64 antenna elements, respectively. 33
2.11 Wideband results for the 3GPP TR 38.901 channel simplifications. 34
2.12 Mean Singular Value Ratio of channel matrices for a scenario with UEs equipped with 16 antenna elements, and gNBs with 64 antenna elements. 34
2.13 Visual representations of our simulation scenarios. 37
2.14 Evolution of the SINR experienced when the test RX moves in the *L-Room* scenario along the path described in Fig. 2.13b. 40
2.15 Cumulative Distribution Function of the SINR in different scenarios vs. R, with $\gamma_{\text{th}} = -\infty$ dB. 41
2.16 Average SINR vs. γ_{th} as a function of the antenna architecture in the *L-Room* scenario, with $R = 4$. 41
2.17 End-to-end performance vs. R for the *L-Room* scenario with an offered UDP CBR traffic of 800 Mbps. 41
2.18 End-to-end performance vs. R and γ_{th} for the *L-Room* scenario with full-buffer TCP traffic. 42
2.19 Average throughput considering $\gamma_{\text{th}} = -\infty$ dB. 44
2.20 Average throughput for the 800 Mbps UDP traffic considering different antenna architectures. 45
2.21 Throughput performance for the *L-Room* scenario with a purely deterministic and quasi deterministic channel model, with a UDP CBR traffic of 800 Mbps. . . 46
2.22 Simulation runtime vs. R and γ_{th} for the *L-Room* scenario. 47
2.23 Trade-off between the SINR performance and the speedup obtained with the different simplification parameters for the three scenarios. 49

2.24 Trade-off between the throughput performance and the speedup obtained with the different simplification parameters for the three scenarios. 50

2.25 Visual representation of the blockage evaluation scenario. 54

2.26 Results of our first simple simulation campaigns. 55

3.1 Workflow of the proposed framework. The diagram highlights how the parameter optimization is achieved using an ML-based emulator. 65

3.2 Correlation between selected inputs and outputs. 66

3.3 Plots show the nRMSE as a function of the number of training samples. 67

3.4 Representation of a one-dimensional plot obtained by fixing all the array parameters except one. 68

3.5 Comparison among the network performance obtained with the baseline configuration (blue bar), with the optimal configuration identified using the simulator samples (orange bar) and using the emulator (green bar). 71

3.6 Example of a generated array. Dashed lines separate the four quadrants, while black and gray dots represent respectively the activated antennas and the array lattice. The top-left quadrant is generated and then mirrored to the other three. 72

3.7 Correlation plot of the four input parameters vs the output metric (y-axis) and the bound metric (color). 73

3.8 Plot showing cross-validation scores on the nRMSE metric with increasing training size. 74

3.9 Visual representation of the activation probability of any given element from the lattice using the optimal parameters α_y^*, α_z^*. Elements outside the central column are never activated, indicating that a vertically-shaped antenna is optimal. 75

3.10 Performance comparison between different classes of antennas. 76

3.11 Visualization of the input configuration around the optimal value. Slices of the input configuration are taken to show how the different parameters affect the performance. 76

3.12 Reference scenario. 78

3.13 Temporal evolution of the signal quality experienced by UT_1. 79

3.14 Comparison of the SNR/SINR CDFs for different BF schemes. 80

3.15 Comparison of the SNR/SINR CDFs for different phased antenna array configurations at the UT side. 80

3.16 Comparison of the SNR CDFs for different antenna element patterns. 81

3.17 APP-layer throughput for different inter-packet intervals. 81

4.1 Graphical representation of sector structure in IEEE 802.11ad. 89

4.2 Representation of a Beacon Interval (BI). 89

4.3 Representation of ADDTS scheduling in IEEE 802.11ad. 91

4.4 Performance of the different scheduling configurations with a bursty application with deterministic period $T = 102.4$ ms. 94

4.5 Performance of the four scheduling configurations using a bursty application with period equal to $T = 102.4$ ms, and an offered rate R for each user. 97

4.6 Average delay of the different scheduling configurations with a bursty application with normally distributed period, with mean equal to $T = 102.4$ ms and standard deviation equal to a fraction of its mean $\sigma = \rho T$. 98

4.7 Example of allocations A_1 (orange) and A_2 (blue), with $p_2 = 2p_1$. 99

4.8 Feasibility check for an *infeasible* pair of allocations, where A_1 (blue) was a pre-existing allocation with $p_1 = \frac{1}{2}$, and the algorithm is checking whether a new allocation A_2 (orange) with $p_2 = \frac{1}{3}$ is compatible. 100
4.9 Feasibility check for a *feasible* pair of allocations, where A_1 (blue) was a pre-existing allocation with $p_1 = \frac{1}{4}$, and the algorithm is checking whether a new allocation A_2 (orange) with $p_2 = \frac{1}{2}$ is compatible. 101
4.10 Representation of a collision between A_n and A_N. 105
4.11 Results for Scenario 1. 107
4.12 Results for Scenario 2. 109

5.1 Portion of traffic trace from *Virus Popper* (50 Mbps, 30 FPS). For this trace, 130–140 individual fragments make up a video frame burst. 120
5.2 Results from acquired VR traffic traces. 121
5.3 Video frame distributions for *Virus Popper* (30 Mbps, 60 FPS). 122
5.4 Video frame fit qualities for the *Google Earth VR – Cities* application. 123
5.5 Comparison of the three best fitting distributions for *Virus Popper* (30 Mbps, 60 FPS). 124
5.6 Generalization models for the *Google Earth VR – Cities* application. 125
5.7 Simulation results for a single user streaming the *Google Earth VR – Cities* application over a Wi-Fi link. 131
5.8 Simulation results for multiple users streaming the *Google Earth VR – Cities* application over a Wi-Fi link. 133

Listing of tables

2.1 NIST's Lecture Room material library. 21
2.2 Characteristics of the simulated scenarios. 36

3.1 Numerical results shown in Fig. 3.5. 71
3.2 Simulation parameters. 79

4.1 Simulation parameters 92

5.1 Parameters of the proposed generative model. 126
5.2 List of simulation parameters 130

Acronyms

Symbols

3DoF	3 Degrees of Freedom. 114, 117, 118
3GPP	3rd Generation Partnership Project. 1, 2, 4, 9, 10, 12, 28, 29, 28, 30, 31, 32, 33, 38, 56, 62, 63, 65, 69, 71, 78, 79, 81, 82, 139
5G	5th Generation. v, vii, 1, 12, 38, 56, 59, 60, 61, 82, 139, 141
6DoF	6 Degrees of Freedom. 114, 135

A

A-BFT	Association-BeamForming Training. 89
AC	Access Category. 87, 90
ADDTS	Add Traffic Stream. xiv, 90, 92, 93, 95, 96
AFBW	Average Fading Bandwidth. 32, 33
AIFS	Arbitration Inter-Frame Space. 87
AMC	Adaptive Modulation and Coding. 38
AoA	Angle of Arrival. 9, 10, 16, 19, 22, 28, 29, 53
AoD	Angle of Departure. 9, 10, 16, 19, 22, 28, 29, 53
AP	Access Point. xxi, 86, 99, 129, 130, 132, 135
APP	Application. v, vii, 5, 7, 78, 81, 93, 98, 110, 120, 129, 130, 140
AR	Augmented Reality. 5, 85, 113
ARD	Automatic Relevance Determination. 74
ATI	Announcement Transmission Interval. 89, 90

B

B-frame	Bipredictive-coded frame. 122
BF	Beamforming. 60, 62, 63, 64, 77, 78, 79, 80, 81, 82, 139
BHI	Beacon Header Interval. 89, 90, 93
BI	Beacon Interval. xiv, 87, 88, 89, 90, 92, 93, 95, 96, 99, 101, 102, 109, 110
BS	Base Station. 77, 78, 79, 80
BTI	Beacon Transmission Interval. 89

C

CAD	Computer-aided Design. 13, 14, 20, 22, 24, 26, 34, 35, 38
CBAP	Contention-Based Access Period. 87, 88, 90, 91, 92, 93, 95, 96, 97, 99
CBR	Constant Bitrate. xiii, 40, 41, 111, 120, 122, 130, 136, 140
CDF	Cumulative Distribution Function. xiii, 23, 22, 40, 43, 46, 79, 122, 123, 124, 128
CI	Confidence Interval. 43
CI/CD	Continuous Integration and Developement. 52

D

DCF	Distributed Coordination Function. 87

DKED	Double Knife-Edge Diffraction. 25, 26, 28
DL	Downlink. 114, 117, 119, 120, 121, 135
DMG	Directional Multi-Gigabit. 89, 90, 111
D-ray	Deterministic Ray. 16, 17, 18, 22
DTI	Data Transmission Interval. 87, 88, 89, 90, 91, 92, 108

E

E2E	End-to-End. 86, 87, 111, 139
EDCA	Enhanced Distributed Channel Access. 90
EM	ElectroMagnetic. 13
ESE	Extended Schedule Element. 90

F

FEC	Forward Error Correction. 129
FoV	Field-of-View. 20, 116, 117
FPS	Frames Per Second. 117, 135, 137

G

gNB	Next Generation Node Base. xiii, 28, 31, 33, 62
GoP	Group of Pictures. 117, 122, 127
GPR	Gaussian Process Regression. 69

H

HMD	Head Mounted Device. 113, 114, 129, 130, 136

I

I-frame	Intra-coded frame. 122
IEEE	Institute of Electrical and Electronics Engineers. xiv, 1, 2, 16, 85, 86, 87, 88, 89, 90, 110, 111, 129, 140
IFI	Inter-Frame Inter-arrival. 120, 122, 123, 124, 125, 126, 132
ITU	International Telecommunication Union. 25

K

KED	Knife-Edge Diffraction. 26, 27
KPI	Key Performance Indicator. 94, 102, 107
KS	Kolmogorov-Smirnov. 23, 123, 124

L

LCF	Level Crossing Frequency. 32, 33
lcm	least common multiple. 100
LoS	Line-of-Sight. 12, 14, 27, 28, 29, 31, 36, 38, 39, 40, 51

M

MAC	Medium Access Control. v, vii, 5, 7, 8, 38, 81, 85, 87, 88, 90, 91, 93, 98, 110, 140

MCS	Modulation and Coding Scheme. 92, 93, 96, 111, 132
METIS	Mobile and wireless communications Enablers for the Twenty-twenty Information Society. 26
MIMO	Multiple Input, Multiple Output. 64, 77, 85
ML	Machine Learning. 7, 60, 61, 64, 65, 66, 75, 82, 83, 112, 139
MLP	Multi-Layer Perceptron. 74
mmW	Millimeter Wave. v, vii, 1, 2, 4, 6, 7, 8, 9, 10, 11, 12, 13, 16, 23, 25, 29, 38, 46, 56, 57, 59, 60, 61, 65, 77, 82, 85, 87, 88, 89, 135, 139
MoI	Method of Images. 13, 14, 15, 33, 56, 139
MPC	Multi Path Component. xiii, 9, 11, 12, 17, 18, 19, 22, 34, 35, 36, 39, 40, 41, 45, 47, 51, 56
MR	Maximum Rate. 113
MU-MIMO	Multi-User Multiple Input, Multiple Output. 63, 111

N

NIST	National Institute of Standards and Technology. 13, 20, 56
NLoS	Non-Line-of-Sight. 12, 29, 31, 35, 36, 38, 39, 43, 51
NN	Neural Network. 61, 70
NR	New Radio. 1, 2, 9, 60, 78, 79, 82, 139
nRMSE	Normalized Root Mean Square Error. xiv, 48, 50, 51, 67, 68, 74
ns-3	Network Simulator 3. 2, 4, 5, 6, 7, 8, 10, 38, 40, 46, 47, 48, 50, 60, 64, 77, 82, 83, 86, 87, 88, 91, 110, 111, 126, 128, 129, 137, 139, 140

O

O2I	Outdoor-to-Indoor. 31, 65, 69, 71
OFDM	Orthogonal Frequency Division Multiplexing. 32, 38

P

P-frame	Predictive-coded frame. 122
PBSS	Personal Basic Service Set. xxi, 86
PCP	Personal Basic Service Set (PBSS) Central Point. xxi, 86
PCP/AP	PBSS Central Point/Access Point. 86, 89, 90, 91, 92, 93, 95, 98, 111
PDCP	Packet Data Convergence Protocol. 38, 41
PDF	Probability Density Function. 122, 123, 125
PHY	Physical. v, vii, 4, 7, 8, 10, 85, 90, 92, 93, 96, 139

Q

QD	Quasi Deterministic. xiii, 7, 11, 16, 17, 19, 20, 22, 23, 24, 22, 33, 36, 38, 39, 45, 46, 48, 51, 56, 139
QoE	Quality of Experience. 5, 114, 116, 136
QoS	Quality of Service. 85, 86, 87, 90, 91, 96, 110, 111, 117, 137, 140

R

RAN	Radio Access Network. 1
RCS	Radar Cross-Section. 27
RF	Radio Frequency. 26, 59, 63, 77

RL	Reinforcement Learning. 87, 112
RLC	Radio Link Control. 38, 81
RRC	Radio Resource Control. 38
RT	Ray Tracer. xiii, 4, 9, 10, 11, 12, 13, 16, 20, 22, 35, 36, 38, 41, 46, 47, 48, 51, 56, 139
RX	Receiver. 9, 13, 14, 16, 20, 22, 23, 34, 35, 36, 38, 39, 40, 41, 64, 77

S

SCM	Spatial Channel Model. 4, 8, 9, 10, 11, 12, 28, 29, 56, 139
SINR	Signal-to-Interference-plus-Noise Ratio. xiii, 28, 31, 33, 38, 39, 40, 41, 42, 43, 48, 51, 64, 67, 69, 71, 78, 79, 80
SIR	Signal-to-Interference Ratio. 31, 32
SNR	Signal-to-Noise Ratio. 55, 78, 79, 80, 81
SP	Service Period. 86, 87, 90, 91, 92, 93, 95, 96, 97, 98, 99, 102, 103, 110, 111
SSW	Sector Sweep. 89
STA	Station. 86, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 102, 103, 111, 129, 130, 132
SVD	Singular Value Decomposition. 31, 38, 64, 77, 79, 81
SVM	Support Vector Machine. 70, 73
SVR	Support Vector Regressor. 69, 70, 73

T

TCP	Transmission Control Protocol. xiii, 41, 43, 46, 129
TDD	Time Division Duplexing. 38
TSPEC	Traffic Specification. 90, 91, 111
TX	Transmitter. 9, 13, 14, 16, 20, 22, 36, 38, 64, 77

U

UDP	User Datagram Protocol. xiii, 39, 40, 41, 42, 43, 45, 78, 119, 120, 122, 129
UE	User Equipment. xiii, 28, 31, 33, 38, 62, 70, 78, 79, 80
UL	Uplink. 114, 117, 119, 120, 121, 135
ULA	Uniform Linear Array. 70, 80, 82
UMa	Urban Macro. 30
UMi	Urban Micro-cell. 65, 69, 71, 82
UPA	Uniform Planar Array. 7, 59, 69, 75
UTD	Uniform Theory of Diffraction. 26

V

VBR	Variable Bit Rate. 87, 91, 136
VR	Virtual Reality. 5, 85, 86, 93, 113, 114, 117, 118, 119, 120, 123, 126, 128, 129, 130, 132, 135, 137, 140

W

Wi-Fi	Wireless Fidelity. vii, xv, 2, 85, 87, 96, 111, 118, 129, 130, 132, 140
WiGig	Wireless Gigabit. v, vii, 7, 8, 85, 86, 87, 88, 93, 110, 111, 140
WLAN	Wireless Local Area Network. 1, 88, 117

X

XR eXtended Reality. v, vii, 7, 8, 85, 113, 114, 115, 116, 117, 123, 125, 133, 134, 135, 136, 137

1 Introduction

Recent developments have paved the way towards the 5$^{\text{th}}$ Generation (5G) of cellular networks and enhanced Wireless Local Area Network (WLAN) designs, to address the traffic demands of the 2020 digital society [1]. In particular, 5G systems will support very high data rates (with a peak of 20 Gbps in ideal conditions), ultra-low latency (around 1 ms for ultra-reliable communications), and a 100x increase in energy efficiency with respect to previous wireless generations [1]. To meet those requirements, the 3rd Generation Partnership Project (3GPP) has released a set of specifications for New Radio (NR), the new 5G Radio Access Network (RAN). Similarly, the Institute of Electrical and Electronics Engineers (IEEE) has developed amendments to 802.11 networks, namely 802.11ad [2] and 802.11ay [3], which operate in the Millimeter Wave (mmW) band.[*] 3GPP NR carrier frequency can be as high as 52.6 GHz for Release 15 (even though future Releases may include extensions up to 71 GHz [4]), while IEEE 802.11ad and 802.11ay exploit the unlicensed spectrum at 60 GHz [3].

The vast amount of available spectrum at mmW frequencies will enable multi-Gbps transmission rates [5], unlocking new and futuristic applications for both the business and the consumer markets. Moreover, the short wavelength makes it practical to build large antenna arrays to establish highly directional communications, thus boosting the network performance via beamforming or spatial multiplexing [6]. Despite these promising characteristics, propagation at mmWs raises several challenges for the design and performance of the whole protocol stack [7]. First, the communication suffers from severe path loss, which prevents long-range omni-directional transmissions. Second, mmW links are highly sensitive to blockage from common materials e.g., brick and mortar. Third, the delay and the Doppler spread (which determine the time and frequency selectivity of the channels) are particularly strong at these frequencies

[*] The mmW band is officially considered to be the band comprising wavelength ranging from 10 to 1 mm, i.e., frequencies between 30 and 300 GHz. Many works in the current literature still consider mmW frequencies down to 6 GHz, and often call sub-Terahertz frequencies above 100 GHz.

and may lead to network disconnections [8]. Finally, directional communications require the precise alignment of the transmitter and receiver beams, hence implying an increased control overhead for channel estimation and mobility management [9, 10].

The combination of these phenomena makes the mmW channel extremely volatile to mobile users. Although some early performance evaluations have suggested that mmW networks can offer orders of magnitude greater capacity than legacy systems (e.g., [11]), a deeper understanding of the propagation channel is required to reliably characterize such networks. In this sense, experimental testbeds make it possible to examine the network performance in real-world environments with extreme accuracy [12]. However, the prohibitive cost and limited flexibility of these platforms make this approach impractical for most of the research community [13].

Network simulators are fundamental tools to assess the effectiveness of novel designs, architectures, and algorithms for networking problems, offering the possibility to monitor the performance of the overall system in a controlled environment, with different scenarios and parameter settings, and without the need for a real deployment. Among the many simulation tools that have been developed so far, either commercial or open source, Network Simulator 3 (ns-3) stands out for its modularity, flexibility, and realism while being an active open-source project. Indeed, ns-3 provides several simulation modules which can be easily interfaced together and extended in order to simulate even complex and realistic deployments and to account for all the phenomena influencing the network behavior using the desired level of abstraction.

In order to provide a solid support for the research in this field, simulation tools must be updated in parallel with the evolution of the communication technology, either by enhancing the existing models or by adding new ones. In this regard, ns-3 is being enriched to provide support for the simulation of the latest cellular (i.e., 3GPP NR [14]) and Wireless Fidelity (Wi-Fi) technologies (i.e., IEEE 802.11ax [15] and 802.11ad/ay [16, 17]).

The objective of this thesis is to describe and analyze the work that has been done to advance the state of the art in different parts of the communication stack. We discuss the current limitations for the simulation of communication networks with ns-3 and provide some future directions to fill the gaps. The remainder of this introductory chapter is organized as follows: Sec. 1.1 will discuss why and how communication networks can be simulated; Sec. 1.2 will describe the main open challenges and research directions when simulating complex network architectures; Sec. 1.3 will introduce the mathematical notation that will be used throughout the remainder of this manuscript; Sec. 1.4 will contain the main contributions brought by our works; finally, Sec. 1.5 will conclude the introduction, briefly describing the structure of this thesis.

1.1 Simulation of Communication Networks

The evaluation of communication systems typically involves three different methodologies, which have different purposes and are usually exploited jointly in order to fully understand the system behavior, namely analytical modeling, real-world measurements, and simulation. In particular, analytical modeling provides a general characterization of the system and a preliminary

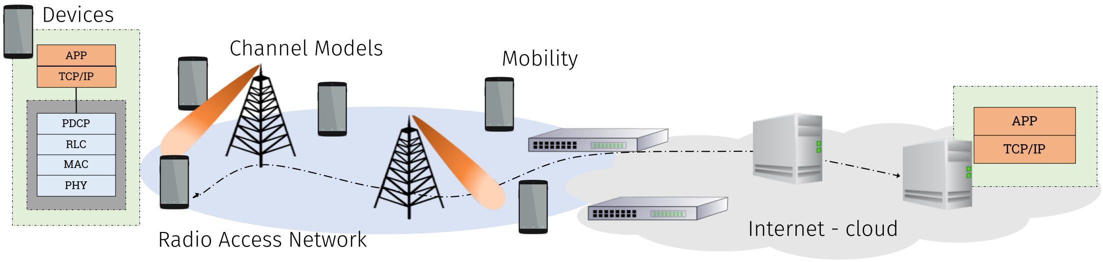


Fig. 1.1: Elements Involved in the End-to-end Simulation of a Communication Network

evaluation, usually through the derivation of bounds and/or approximations for the system performance. However, it may be difficult to devise mathematical models capable of capturing all the relevant dynamics, and several assumptions may be needed to make these models tractable. On the other hand, measurement campaigns on prototypes or real systems could provide very accurate results, but they are very expensive and difficult to conduct. Sometimes the realization of a working prototype may not even be feasible, because the required hardware may not be available on the market. Finally, simulations consist in mimicking the system operations running random experiments through a computer program, relying on one or multiple models describing the system and the phenomena influencing its behavior, with a certain level of abstraction. The latter is chosen according to the desired evaluation accuracy, allowing the testing of the system performance at different scales and in different situations. Also, simulations make it possible to arbitrarily set the operating conditions under which the system has to be tested, and to reproduce them at any time. Therefore, simulation allows the comparison of the performance of different variants of the system or different configuration options and is consequently an important tool in network architecture and protocol design.

Nonetheless, the reliability of the evaluation results mainly depends on the quality of the simulation models, which have to be detailed enough to include the characterization of all the phenomena of interest, and this may be challenging when the system is too complex.

A communication network is an example of an extremely complex system, as shown in Fig. 1.1, in which different elements impact the overall performance in different ways and at different scales. For instance, user mobility, data traffic, and the propagation environment may dramatically change the network behavior, thus their characterization is of fundamental importance to obtain a proper level of accuracy [18]. To this aim, network simulators typically provide multiple simulation modules, each focusing on the modeling of a different phenomenon, which can be interfaced together to obtain a thorough description of the overall system. Also, each module may provide multiple models describing the same phenomenon in different ways, e.g., at different levels of abstraction or using different approaches.

In general, the simulation of communication networks involves both the modules implementing the communication technology of interest, and other modules which take into account the external phenomena influencing the system behavior. While the realization of standard-defined protocol models is generally straightforward, because they are completely defined by one or multiple documents, the design of models to account for the external phenomena is more problematic because of their stochastic nature.

1.2 Open Challenges and Research Directions

In this section we will identify a number of open challenges that we faced while running extensive simulation campaigns, mainly related to mmW communication networks [19], while suggesting possible research directions that can improve the suitability of ns-3 for the simulation of communication networks in different scenarios [20].

Channel Models: The channel model describes the propagation of the signal through the surrounding environment, taking into account different factors influencing this phenomenon, such as the carrier frequency, the spatial characteristics of the channel, the presence of blocking objects. Given that the Physical (PHY) layer, and thus the performance of communication networks, strongly depends on the nature of the wireless channel, proper modeling of its behavior is of primary importance to obtain accurate simulation results. This is particularly true for mmW systems [21], in which the high carrier frequency and the directionality of the communications make the propagation of the signals very sensitive to the characteristics of the environment, and even more so for Terahertz systems [22]. Indeed, the presence of buildings, trees, or even the rain may reflect, attenuate and refract the signals, hence their influence has to be taken into account for proper modeling of the channel [23]. Moreover, antenna arrays will be used to overcome the high attenuation experienced at such high frequencies by increasing the directionality of the propagation [21]. In this context, channel models currently implemented in ns-3 (e.g., Friis, Okumura Hata, trace-based fading model) do not take into account the necessary characteristics needed to properly model multi-antenna systems at these higher frequencies. To solve this problem, either Ray Tracer (RT)-based simulation or stochastic Spatial Channel Models (SCMs) should be used. While RT yields the most realistic results, it is complex to program, debug, and calibrate as well as very computationally demanding to execute. Furthermore, it has to be specific for a given scenario, defining the geometry of the environment, materials of walls and obstacles, positions of receiver and transmitter. A more lightweight albeit less precise solution can be found in stochastic SCMs. These are stochastic channel models that consider multiple rays coming from different angles with different powers and delays, thus being able to model complex wideband channels. Although SCMs are still much faster than ray tracers, channels like 3GPP 38.901 [24] can act as bottlenecks given the large number of parameters and computations, especially when large arrays are considered. This is exacerbated by the fact that the code is rarely optimized.

Antennas: To overcome the high attenuation experienced at very high frequencies, next-generation communication networks will use multi-antenna systems capable of directing the transmit power in the desired direction through suitable beamforming. The ns-3 `antenna` module provides single-element antenna models with isotropic, parabolic and cosine radiation patterns, but lacks a general model for antenna arrays which is essential for the simulation of next-generation communication networks [25].

Vendor-Specific Protocols: In general, communication standards are needed to specify how the information should be transmitted to the final users and transferred across the network, containing best practices emerged from the collective experience of both academic and industry experts. Defining the basic signaling, architectures, and protocols allows for different devices produced by different manufacturers to work together, vastly increasing market competition and cooperation, finally resulting in better and cheaper products. Nevertheless, not every single part of the communication system is defined by standards in detail, allowing different vendors to differentiate themselves from their competitors by developing better inner protocols and processes which do not affect interoperability. One example of this is Medium Access Control (MAC) scheduling, meaning how access points or base stations schedule communications with or between users whenever possible. Given the complexity of the task, different optimizations can be aimed for, such as maximimizing sum-capacity or fairness, or minimizing interference or latency. While some standard scheduling techniques are well known, constraints given by specific communication standards or peculiar types of traffic sources might require custom schedulers to be designed, developed, and tested.

Applications: Application (APP) layer models are used to mimic the generation of the user traffic. This is a very important aspect in the simulation, because it determines the load under which the network has to operate. The usage of inappropriate and unrealistic traffic models may lead to evaluating the system under conditions that are not representative of those encountered in real-world scenarios, thus potentially leading to wrong conclusions. Moreover, with the emergence of new applications (e.g., Virtual Reality (VR) and Augmented Reality (AR), self-driving cars, tactile Internet), communication networks have to deal with several types of traffic with very diverse characteristics, therefore a proper modeling of the behavior of the upper layers of the protocol stack is fundamental to obtain accurate results from the simulations. While the need for accurate application models holds for the simulation of any kind of network, in mobile scenarios the user generally uses different interaction patterns than in wired networks. For example, bulk transfers of data are generally uncommon, while video streaming, web browsing, chatting and VoIP are more popular. In this regard, while ns-3 generally offers a number of both deterministic and statistical packet sources, it lacks applications that model complex and realistic interactions and expose Quality of Experience (QoE) metrics. There have been a few attempts aimed at simulating video streaming, through either the parsing of traces or the implementation of dynamic adaptive streaming over HTTP, however none of them has been integrated in ns-3 [26, 27].

Scalability: There is a trade-off between the accuracy and the complexity of the simulations. For example, the high level of detail in the scheduling of synchronization signals for the communication stacks may yield an excessive overhead in terms of simulation complexity.

Similarly, complex channel models may come very close to mimicking a real wireless channels, but the sheer amount of operations needed to create a single channel between a pair of nodes might result in excessively demanding simulations, or even make them infeasible when scaling to multiple nodes or long simulations. Therefore, it makes sense to investigate and implement mechanisms that allow the simulation to reduce the computation complexity, being it scheduled events or mathematical operations, and consequently the simulation runtime. Even though it should be noted that this approach is not always possible, finding scalable approaches where the simulation complexity is minimized according to the simulated scenario is a worthwhile research direction. Succeeding in this task would allow researchers to simulate more complex and realistic scenarios more accurately, making it possible to produce results more faithful to the real world.

Support in ns-3: Ns-3 provides different simulation modules, including multiple implementations of the aforementioned models. Just to mention a few relevant modules, the `spectrum` and the `propagation` modules deal with the modeling of the wireless channel, the `antenna` module can be extended to support different types of antenna radiation patterns, while the `application` module provides several traffic generation models. While the usage of these modules is a first important step for realistic simulation scenarios, their modeling capabilities could be improved to account for a number of factors that are important in next-generation networks. In the following sections, we will outline our contributions to the improvements on both the ns-3 simulator and to network simulation in general, as well as highlight the subsequent research that was made possible thanks to our newly built tools.

1.3 Mathematical Notation

Throughout this thesis, boldface letters denote random variables while non-boldface letters denote deterministic variables or realizations of random variables. Simple math font (e.g., a, A) is used for both scalar and vector variables, while bold math font is used for random variables (e.g., $\boldsymbol{a}$, $\boldsymbol{A}$). The function $d(x_1, x_2)$ corresponds to the euclidean distance between points x_1 and x_2 in 3D space. Finally, the following notation and distributions for random variables are assumed:

- $\mathbf{X} \sim \mathcal{N}\big(\mu, \sigma^2\big)$: Normal distribution with $\mathbb{E}[\mathbf{X}] = \mu$ and $\mathrm{var}(\mathbf{X}) = \sigma^2$
- $\mathbf{X} \sim \mathcal{R}(s, \sigma)$: Rician distribution where $s, \sigma \geq 0$. It can be generated as $\mathbf{X} = \sqrt{\mathbf{Y}^2 + \mathbf{Z}^2}$, where $\mathbf{Y} \sim \mathcal{N}\big(s, \sigma^2\big)$, $\mathbf{Z} \sim \mathcal{N}\big(0, \sigma^2\big)$.
- $\mathbf{X} \sim \mathcal{L}\big(\mu, \sigma^2\big)$: Laplacian distribution with $\mathbb{E}[\mathbf{X}] = \mu$ and $\mathrm{var}(\mathbf{X}) = \sigma^2$
- $\mathbf{X} \sim \mathcal{E}(\lambda)$: Exponential distribution with $\mathbb{E}[\mathbf{X}] = \frac{1}{\lambda}$ and $\mathrm{var}(\mathbf{X}) = \frac{1}{\lambda^2}$
- $\mathbf{X} \sim \mathcal{U}[a, b]$: Uniform distribution in the closed interval $[a, b]$

1.4 Main Contributions

The collaborative work done for this thesis brings diverse contributions to the state of the art in multiple topics. Specifically, regarding mmW channel modeling, the main contributions can be summarized as follows:

1. Implementation, often open-source, of popular mmW channel models, such as 3GPP TR 38.901, ray-tracing, and quasi-deterministic models;
2. Performance profiling, analysis, and optimization of mmW channel generation with different channel models;
3. Simplification with the aim of channel generation time reduction of stochastic and deterministic mmW channel models;
4. Mathematical formalization and comparison with real-world measurements of a mmW Quasi Deterministic (QD) channel model;
5. PHY-layer evaluation of mmW channel simplifications;
6. Full-stack simulations and evaluation of mmW channel simplifications;
7. Modeling of interactions between the mmW channel and common obstacles.

Regarding antenna modeling and optimization, instead, our contributions include:

8. Machine Learning (ML)-based optimization framework for complex simulators;
9. Optimization of regular Uniform Planar Arrays (UPAs) in urban scenarios;
10. Optimization of thinned UPAs in urban scenarios;
11. Development of an ns-3 framework for antenna arrays (now part of the official ns-3.34 release) and beamforming.

Focusing on the Wireless Gigabit (WiGig) standard, and specifically on MAC scheduling, our main contributions are:

12. Implementation of a flexible WiGig MAC scheduling framework in ns-3;
13. Full-stack performance evaluation of simple WiGig schedulers for periodic and quasi-periodic applications;
14. Mathematical formalization of WiGig MAC schedulers;
15. Design, development, and evaluation of two complex schedulers.

Finally, to support our work on scheduling, we contributed to APP layer modeling as follows:

16. Traffic acquisition and analysis of a real eXtended Reality (XR) streaming application;

17. Made XR traffic traces publicly available;
18. Proposed the first generative traffic model for XR applications (to the best of our knowledge);
19. Developed a flexible ns-3 application interface for bursty traffic, natively supporting our proposed XR traffic model and acquired traffic traces.

1.5 Thesis Structure

The thesis is structured as follows. After this introductory chapter, Chapter 2 will describe our work done regarding mmW channel modeling, ranging from stochastic SCMs to ray-tracing models, from simplifications for scalability improvements to more faithful modeling of relevant channel behaviors, from PHY-only to full-stack simulations results.

In Chapter 3 we will propose heuristic optimizations for antenna arrays, specifically tailored for urban scenarios, while describing a more geneneral optimization framework that we developed.

Leaving the physical part of the communication systems, Chapter 4 will contain our efforts to propose a MAC scheduling framework tailored specifically for WiGig standards, aiming to support modern and future relevant applications. The main focus will be the support of periodic and quasi-periodic applications, analyzing the problem both analytically and through simulations, after briefly explaining the core scheduling mechanisms of the IEEE 802.11ad/ay standards.

To support and better evaluate our works on scheduling, in Chapter 5 we propose traffic models for XR traffic.

Finally, Chapter 6 will conclude this thesis, discussing the impact of what has been achieved.

2
Millimeter Wave Channel Modeling

2.1 Introduction

Theoretical analyses and computer-aided simulations have emerged as important tools in evaluating the performance of novel solutions and the interplay between the mmW channel and the deployment and protocol design. Both analysis and simulation, however, require proper modeling of signal propagation to accurately reproduce the behavior of mmW systems [23, 28]. On one side, analytical studies model the channel using a Nakagami-m or Rayleigh distribution (e.g., [29]). This approach, while simplifying the analysis significantly, assumes a rich multipath channel when in fact it is sparse at mmWs [30].

Similarly, stochastic SCMs, e.g., [24] for 3GPP NR, characterize the channel as a combination of random variables fitted from real-world measurements, providing a more realistic assessment of the mmW network performance compared to their analytical counterparts [31]. Still, the stochastic nature of these models may prevent researchers from evaluating the impact of the channel dynamics in specific environments, and may respond poorly to the need for an accurate characterization of the spatio-temporal evolution of the channel's Multi Path Components (MPCs) [32].

Conversely, RTs can be used to precisely model the propagation of mmW signals in specific scenarios [33, 34]. Unlike analytical or stochastic models, RTs are based on the geometry of the scenario and characterize the different propagation properties of each MPC, including time delay, Doppler shift, polarization, Angle of Departure (AoD) at the Transmitter (TX), and Angle of Arrival (AoA) at the Receiver (RX), thus providing higher accuracy [35]. Moreover, simulators can use ray tracing to model the temporal and spatial evolution of the channel, a necessary feature for a proper planning of wireless systems. However, the generation of the MPCs can be computationally expensive, limiting the scalability of simulations. It is thus fundamental to find a compromise between accuracy and reliability, a research challenge that, to date, has not yet been thoroughly addressed in the literature.

As already mentioned in Chapter 1, the mmW is particularly prone to signal blockage due to high propagation losses, high penetration losses, and deep diffraction shadows at these high frequencies. While some studies have been done to this regard, detailed full-stack simulations should better analyze what happens in case of sudden blockages, and how to recover from such an event. While stochastic models can optionally include a concept of blockage, the reality is often more complex and thus RT-based model should be used to more accurately model reality. We propose a tool to integrate a pre-existing RT channel model with blockers interacting with the radio environment. This, together with our RT channel integration on ns-3, will offer the possibility to more accurately study these phenomena.

The remainder of this chapter is structured as follows. In Sec. 2.2 we review the most relevant analytical, stochastic, and deterministic models for the mmW channel, describing the basic mathematics behind channel modeling, with a focus on stochastic SCMs, deterministic, and quasi-deterministic channel models. Furthermore, our proposal for a measurement-based quasi-deterministic model will be described in Sec. 2.2.4 and compared against measurements in Sec. 2.2.5.

Secondly, in Sec. 2.3 we will describe and analyze the simplifications brought upon the 3GPP TR 38.901 channel model, focusing on the PHY layer metrics.

Thirdly, moving towards (quasi-)deterministic channel models, in Sec. 2.4 we describe the proposed simplifications to be applied to the RT-based channel, providing both PHY layer and end-to-end metrics, as well as thorough computational performance analysis and some RT simulation guidelines.

Fourthly, in Sec. 2.5 we will introduce a currently ongoing project, still in its early stages and with no publications available, yet. The project addresses the problem of modeling the impact of second-order effects on mmW channels, specifically the susceptibility of the mmW signals to physical blockers.

Finally, Sec. 2.6 concludes the chapter, listing our main contributions and suggesting some relevant future works.

2.2 Channel Modeling at MmWaves

The modeling of the mmW channel, in terms of propagation and fading, has been a key research topic in recent years [23]. Multiple measurement campaigns have tried to characterize the properties of the mmW spectrum in diverse scenarios and environments (e.g., urban [5], rural [36], and indoor [35]), and have led to the definition of different channel models.

Comprehensive reviews, discussing propagation, fading, and beamforming models can be found in [30, 37], while [38] specifically focuses on propagation loss. These efforts have identified the key factors for an accurate modeling of the mmW channel. First, multipath components are sparse in the angular domain, thus an accurate model should explicitly characterize the AoA and AoD of the different taps. Moreover, blockage at mmWs has a more remarkable impact on the link dynamics than at sub-6 GHz, which should be accounted for. Finally, rough surfaces could generate more diffuse scatterers than at longer wavelengths.

The aforementioned measurement campaigns have led to different modeling approaches for the mmW channel, which have various degrees of complexity and accuracy, and can be applied to different contexts and evaluations. In the next paragraphs, we will review three broad families, i.e., analytical, stochastic, and quasi-deterministic channel models.

In general, when multi-antenna systems are considered, the channel is not modeled anymore as a complex scalar time-varying impulse response $\mathbf{h}(t, \tau)$, but rather by a matrix $\mathbf{H}(t, \tau) \in \mathcal{C}^{U \times S}$, with S (U) being the number of antenna array elements at the transmitter (receiver). Each entry (i, j) in the matrix $\mathbf{H}$ models the channel between two specific antenna elements, and represents the joint effect of different MPCs. Each MPC is characterized by angles of departure and arrival, power, and delay. The interaction with the antenna arrays can be modeled by pre- and post-multiplying $\mathbf{H}$ by the beamforming vectors of the transmitter and receiver, respectively [31].

Both simulators rely on the computation of the channel matrix $\boldsymbol{H}$ to describe the channel obtained from the MPCs provided by the RT and QD. Given M rays, with path power gain PG_m, phase shift Φ_m, delay τ_m, and angles of departure $\mathbf{AoD}_m$ and angles of arrival $\mathbf{AoA}_m$, the matrix for the carrier frequency f_c is computed as [37]

$$\boldsymbol{H} = \sum_{m=1}^{M} \sqrt{PG_m}\, e^{j(-2\pi\tau_m f_c + \Phi_m)}\, \boldsymbol{a}_{\mathrm{rx}}^{*}(\mathbf{AoA}_m)\, \boldsymbol{a}_{\mathrm{tx}}^{H}(\mathbf{AoD}_m), \tag{2.1}$$

where $\boldsymbol{a}_{\mathrm{rx}}(\boldsymbol{\theta})$ and $\boldsymbol{a}_{\mathrm{tx}}(\boldsymbol{\theta})$ are the receiver and transmitter array responses in the 3D angle $\boldsymbol{\theta}$, $(\cdot)^*$ is the conjugate operator, and $(\cdot)^H$ is the Hermitian operator.

2.2.1 Analytical Channel Models

MmW analytical studies generally used simplified channel models, based on propagation and a single random variable for fading, with Rayleigh or Nakagami-m distributions [29]. Moreover, these are usually combined with a sectorized beamforming model for directional transmissions. This accounts for the beamforming gain G by assigning a maximum gain G_M to a main lobe, of angular width θ_b, and a lower gain G_m for the simplified side lobes in the complementary angular space [39]. This simplified model limits the accuracy in the representation of the interaction between the mmW propagation, the realistic antenna arrays, and the beamforming strategies [23].

2.2.2 Stochastic Spatial Channel Models

An improved characterization of the mmW channel can be achieved using stochastic Spatial Channel Models (SCMs) [40]. For stochastic SCMs, the MPCs are generated from a set of random distributions, whose parameters are determined by statistical fits on channel measurements. The channel matrix thus has a stochastic nature, with the advantage that multiple instances can be randomly generated for generic, large scale scenarios.

An SCM is given by a propagation loss and a fading model. The first characterizes the Line-of-Sight (LoS) state of the link (probabilistically or with a precise description of the environment) and the average channel gain, with different equations for the LoS and Non-Line-of-Sight (NLoS) conditions [38].

Notable examples of stochastic channel models are those derived from the WINNER and the WINNER-II models [41], e.g., the 3GPP TR 38.901 channel model for 5G deployments [24]. These models have been extensively used in the performance evaluations of mmW networks [42], and are also integrated with popular network simulators [43]. The NYU channel model for 28 GHz and 73 GHz is also based on a stochastic SCM [31].

2.2.3 Deterministic Channel Models

The stochastic nature of the aforementioned channel models makes them generic: they can model a common rural or urban scenario, but not a specific scenario (e.g., Times Square in NYC). Therefore, they do not accurately model the interactions of the mmW signal in a peculiar deployment, and cannot be used for detailed planning and capacity studies in real-world contexts.

As discussed in Sec. 2.1, RTs can, instead, provide extremely accurate propagation results in a given environment, provided that its characterization in the simulation is accurate enough. With respect to stochastic channels, an RT generates the exact MPCs that arise from a direct or reflected propagation path in the scenario [33]. Ray Tracers have thus been the basis for several performance evaluation studies at mmWs, e.g., [34, 44].

However, while being extremely precise, RTs models are also more computationally intensive than stochastic models for the generation of a single channel instance, especially if the number of scattering and reflecting surfaces in the scenario is large. A number of optimizations have been studied for RTs in general [45]. Two main aspects are generally considered when trying to enhance the RT performance [46]: (i) reducing the number of objects on which the ray-object intersection test has to be performed, and (ii) accelerating the ray-object intersection test. *Space division methods* [47] partition the elements of the simulation space into clusters and perform the intersection test with the clusters rather than the single objects. *Transmitter-dependent methods* exploit the knowledge of the transmitter node location to pre-process the scene and discard obstacles that are not relevant to the simulation, such as non-illuminated elements [48], obstructed angular sectors [49], and those reached by a negligible amount of power [50]. Similarly, *receiver-dependent methods* reduce the number of required operations based on the spatial distribution of the receivers [51]. Notice that both transmitter- and receiver-dependent methods require the pre-processing to be performed whenever the reference node moves, thus leading to a significant overhead in dynamic environments. Other techniques speed up the RT's performance by either preemptively performing an exhaustive simulation to produce a coverage map for all the receivers' positions [52], or interpolating the metrics observed between a grid of simulated points [53]. Finally, [54] analyzes the performance of an RT as a function of the number of reflections, focusing only on the mean square error with respect to measurements in different indoor locations.

To be more precise, RT is widely used to accurately simulate ElectroMagnetic (EM) propagation in an environment whose geometry is described by a Computer-aided Design (CAD) model [55]. This technique is based on the solution of Maxwell's equations for the far field when the operating frequency tends to infinity, where EM waves exhibit ray-like properties, i.e., the flow of power propagates along straight lines and reflects specularly on flat surfaces. At mmWs, where the wavelength of the signal is much shorter than the typical size of the reflecting obstacles, ray tracing can therefore be used as a first approximation to model the propagation effects, whereas secondary effects such as diffraction, diffuse scattering, polarization, and refraction, should also be accounted for if a better accuracy is desired.

Some of these effects can be particularly significant in the propagation of mmW signals. In this frequency band, the shorter wavelength leads to a higher effective roughness of the surfaces, thus increasing the amount of scattered power and, consequently, the reflection loss. The effect is twofold: on one side, higher-order reflections are expected to be weaker and thus affect the communication less than at lower frequency but, on the other side, proper modeling of the scattered rays should be taken into consideration [56]. Conversely, higher penetration loss will reduce the received power in the cluttered areas, improving frequency reuse and reducing the cross-interference between close-by radiators. Finally, diffraction shadows are deeper at higher frequency, making diffraction a less prominent means of propagation in the mmW band.

For the results of this thesis, we use an open-source RT developed jointly by the SIGNET group at the University of Padova[*] and the U.S. National Institute of Standards and Technology (NIST)[†]. It uses triangles described in CAD files as the basic 3D surface element units, which can then be combined to define complex shapes.

The RT only supports specular reflections and, optionally, diffuse scattering, ignoring effects such as diffraction, penetration, and polarization. Specifically, polarization is not considered to further simplify the software from the Fresnel equations. Thus, reflected rays experience a 180° phase rotation and a random reflection loss $\boldsymbol{RL}$ whose detailed distribution has been experimentally characterized and is reported in the material library available at [57].

Multiple network nodes, playing the roles of TXs and RXs, are modeled as points, and can be deployed simultaneously, allowing the calculation of interfering channels. Furthermore, trace-based mobility is supported, making it possible to create complex scenarios with multiple base stations and mobile users. Given N nodes and t time steps, the simulator computes a channel instance for each time step and for each node pair. Considering a symmetric channel for a given node pair, $tN(N-1)/2$ channel instances have to be calculated.

The RT uses the Method of Images (MoI) [55], a well-known method for ray tracing originally adapted from geometrical optics [58] to be used in computer graphics and in the early works on channel modeling for cellular networks [59]. The Method of Images (MoI) computes specular reflections, assumed to be independent across time and node pairs. For the simplest case, i.e.,

[*] http://signet.dei.unipd.it/

[†] https://www.nist.gov/ctl

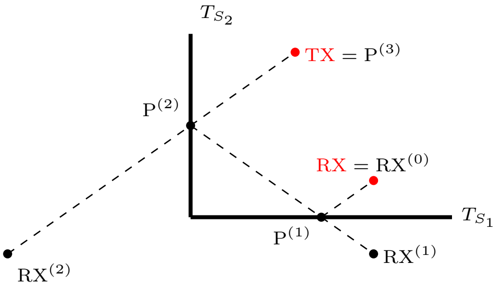


Fig. 2.1: Visualization of the MoI algorithm for a second-order reflection ($r = 2$) [55].

first order reflections, it defines the virtual image of a node, for example the RX, to be a specular image with respect to a surface. Formally, RX$^{(1)}$ is the specular image of the RX, defined as RX$^{(0)}$, across the surface S. The specular reflection point P$^{(1)}$ between the RX and the TX coincides with the intersection of the segment $\left(\mathrm{RX}^{(1)}, \mathrm{TX}\right)$ with the surface S, as shown in Fig. 2.1.

Since triangles are used as the basic surface units of the CAD environment, S_i is the plane generated by a given triangle T_i, $i = 1, \dots, T$, where T is the total number of triangles of the CAD environment, and it must be verified that P$^{(1)}$ is a point within the area shaped by T_i, otherwise the reflection will not be valid and will thus be discarded. Finally, every segment of the ray, namely $\left(\mathrm{RX}^{(0)}, \mathrm{P}^{(1)}\right)$ and $\left(\mathrm{P}^{(1)}, \mathrm{TX}\right)$, has to be checked against the remaining triangles of the environment T_j, $j = 1, \dots, T$, $j \neq i$ for obstruction. If any segment of the ray is obstructed, the whole ray is considered obstructed and thus discarded.

The MoI applies recursively when multiple reflections are considered, computing the n-th virtual image of the RX, RX$^{(\mathrm{n})}$, and the respective specular reflection point P$^{(\mathrm{n})}$ as shown in Fig. 2.1. Thus, for each ray of reflection order $r > 0$, r geometrical operations must be done to compute the ray path and, if the ray is valid, each of the $r+1$ segments needs to be checked for obstruction over the $T-1$ triangles of the CAD environment. Summing the geometrical and the obstruction-check operations, the number of operations per ray is thus

$$n_{\mathrm{ray}}(r) = r + (r+1)T, \quad T \gg 1. \tag{2.2}$$

To compute all possible reflections between a given node pair, all possible paths have to be computed and tested for obstruction. Considering increasing reflection orders, the first one to be tested is the direct ray, i.e., the segment $(\mathrm{RX}, \mathrm{TX})$. Subsequently, all first order reflections are computed, i.e., those rays starting from the TX, reflecting off a triangle T_i, $i = 1, \dots, T$ and reaching the RX. Then, second order reflections starting from the TX, reflecting first on triangle T_{i_1}, $i_1 = 1, \dots, T$ and then on triangle T_{i_2}, $i_2 = 1, \dots, T$, $i_2 \neq i_1$ to finally reach the RX, and so on up to a maximum reflection order R.

All the reflections can be encoded in a *reflection tree*, as represented in Fig. 2.2, where each node of the tree corresponds to a possible ray, the node depth corresponds to the reflection

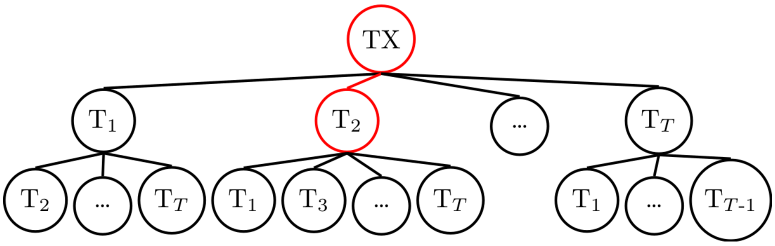


Fig. 2.2: Example of reflection tree for $R = 2$ (see the visibility tree from [60]). The highlighted path of depth $d = 1$ corresponds to a ray that starts from the TX node and reflects on triangle T_2 before possibly reaching the RX node.

order r starting from 0 at the root of the tree, and the path starting from the root describes the ordered tuple of reflecting triangles to be tested. The reflection tree is a simplified concept with respect to the *visibility tree* described in [60]. The added flexibility, and thus complexity, introduced by the triangular tessellation is able to support fine details in a scene, but computing whether all pairs of triangles are in mutual LoS requires a significant overhead and was thus not implemented. Instead, we implement only a basic visibility algorithm, able to pre-process the scenario and discard interactions between triangles which are trivially not in LoS. The model discards all interactions between a given triangle T_i and all other triangles $\{T_j\}$, $j \neq i$, which are fully behind T_i, exploiting the directionality of a surface as is typical for CAD models.

The complexity of a single channel instance $n_{\mathrm{ch}}(r)$, then, is determined by the total number of operations required for all the nodes of the reflection tree, where we consider the upper bound given by the total number of triangles ignoring the visibility pre-processing step. At depth $r = 0$ we consider only the direct ray, at depth $r = 1$ we consider the T possible first-order reflections, then, in general, at depth $r \geq 2$ we consider $T(T-1)^{r-1} < T^r$ possible r-order reflections.

Thus, combining the geometrical complexity $n_{\mathrm{ray}}(r)$ derived in (2.2) with the upper limit of the reflection tree depth (T^r), the number of operations per channel instance $n_{\mathrm{ch}}(r)$ considering only reflection order r is upper bounded as

$$n_{\mathrm{ch}}(r) \leq n_{\mathrm{ray}}(r) T^r = (r + (r+1)T)T^r. \tag{2.3}$$

Then, considering the reflection order up to R, the overall number of operations n_{ch} required by the MoI to compute a channel instance between a pair of nodes is:

$$\begin{aligned} n_{\mathrm{ch}} = \sum_{r=0}^{R} n_{\mathrm{ch}}(r) &\leq \sum_{r=0}^{R} (r + (r+1)T)T^r \\ &= T \sum_{r=0}^{R} T^r + (T+1) \sum_{r=0}^{R} r T^r. \end{aligned} \tag{2.4}$$

The first term is a truncated geometric series and the second is a special case of (truncated) arithmetic-geometric series, known as Gabriel's staircase, that can be solved as follows:

$$\begin{aligned} n_{\text{ch}} &\leq T\frac{T^{R+1}-1}{T-1} + (T+1)\frac{T(RT^{R+1}-(R+1)T^{R}+1)}{(T-1)^2} \\ &\sim T^{R+1} + RT^{R+1} \quad \text{for } T \gg 1 \end{aligned} \tag{2.5}$$

thus denoting a complexity per channel instance equal to $\mathcal{O}\big(RT^{R+1}\big)$, and a total complexity for N nodes and t time steps equal to $\mathcal{O}\big(tN^2RT^{R+1}\big)$. The last step in (2.4) is justified by considering that typical values for R and T are in the order of 1–4 and 100–10 000, respectively, thus making T the dominating term in the formula.

2.2.4 Quasi-Deterministic Channel Models

Although RT is an extremely powerful and flexible way of modeling the wireless channel, considering only the deterministic components of a channel might not always be enough. Minute details of the environment, unexpected objects, and even the surface roughness of some materials can easily increase the complexity of the mmW channel, needing more rays which are hard of even impossible to predict.

In Sec. 2.4, the contribution of diffuse components will be evaluated, which alone can account for up to 40% of the total received power according to measurement campaigns [35]. The stochastic model for diffused rays is based on the specifications proposed for IEEE 802.11ay channel modeling [61] and its parameters are obtained from accurate measurement campaigns [35]. Further details are given in [62].

In this section, we will provide a step-by-step tutorial on how to generate a channel with a QD model, with a precise and rigorous mathematical formulation.

The QD model considers as a basis a deterministic channel described in Sec. 2.2.3, which can be computed through ray tracing for time t, given an environment geometry, and TX and RX positions [33]. The computed Deterministic Rays (D-rays) will then be the baseline for the multipath components randomly generated by the QD model. If present, the direct ray is treated separately as it does not generate any diffuse component.

The QD model can be realized from the model for a first-order reflection and from it generalized to higher-order reflections. For reasons that will become clear later on, we define the instant in which the direct ray should arrive at the RX (even if it is actually blocked) as $t_0 = t + t_{\text{dir}}$, where $t_{\text{dir}} = \frac{d(\text{TX},\text{RX})}{c}$, and c is the speed of light. From now on we will consider a frame of reference in the variable τ relative to time t_0, where $\tau = 0$ corresponds to t_0. Given this choice, the direct ray, if it exists, will arrive at time $\tau = 0$, whereas the reflected D-rays will arrive at times $\tau > 0$.

First-order reflections In this section, we will provide a step-by-step tutorial on how to generate a channel with a QD model, with a precise and rigorous mathematical formulation. The pseudo-code for this algorithm is reported in Alg. 2.1, while a graphical representation of the parameters is shown in Fig. 2.3.

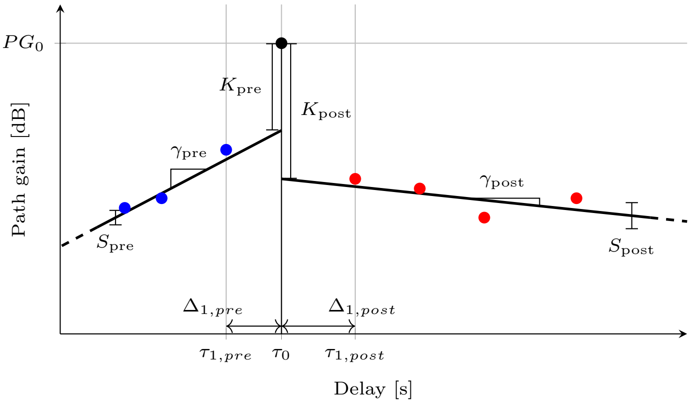


Fig. 2.3: Graphical representation of the QD parameters. The D-Ray is shown in black, while realizations of pre- and post-cursors are shown in blue and red, respectively.

Statistics for all rays are assumed independent of their arrival time. We thus consider, without loss of generality, a single reflected D-ray with arrival time $\tau_0 > 0$, path gain PG_0, AoD along the azimuth/elevation axes $AoD_{az/el,0}$, and AoA $AoA_{az/el,0}$. The same procedure will be repeated for all other reflected D-rays.

A cluster can be defined as the set with a D-ray and the corresponding MPCs. The total number of MPCs of a given cluster will be $N_{MPC} = N_{pre} + 1 + N_{post}$, including pre-cursors (i.e., diffuse components that are received before the D-ray), main cursor (i.e., the D-ray), and post-cursors (i.e., received after the D-ray). Based on some experimental evidence, we suggest to use $N_{pre} = 3$ and $N_{post} = 16$, although these numbers may vary in different locations and for different models.

The arrival times of the MPCs are modeled as a Poisson process, meaning that their inter-arrival times are independent and exponentially distributed. Namely, the post-cursors arrival times $\boldsymbol{\tau}_{i,post}$ are random variables generated based on inter-arrival delays $\boldsymbol{\Delta}_{i,post} = \boldsymbol{\tau}_{i,post} - \boldsymbol{\tau}_{i-1,post}$ as follows

$$\boldsymbol{\Delta}_{i,post} | \boldsymbol{\tau}_{i-1} \sim \mathcal{E}(\boldsymbol{\lambda}_{post}), \tag{2.6}$$

for $i = 1, \ldots, N_{post}$, where the arrival rate $\boldsymbol{\lambda}_{post} \sim \mathcal{R}\big(s_{\lambda_{post}}, \sigma_{\lambda_{post}}\big)$ is a random variable itself. With slight abuse of notation, we consider $\boldsymbol{\tau}_{0,post} = \tau_0$, i.e., the time of arrival of the D-ray. Post-cursors arrival times are then computed as

$$\boldsymbol{\tau}_{i,post} = \boldsymbol{\tau}_{i-1,post} + \boldsymbol{\Delta}_{i,post} = \tau_0 + \sum_{j=1}^{i} \boldsymbol{\Delta}_{j,post}, \tag{2.7}$$

for $i = 1, \ldots, N_{post}$. Please note that random parameters such as λ_{post} should be extracted independently for each D-ray.

Pre-cursors will be similarly generated, with the difference that (2.7) will subtract inter-arrival delay, thus making $\boldsymbol{\tau}_{i,pre} < \tau_0$ for $i = 1, \ldots, N_{pre}$.

Since the number of pre/post-cursors was empirically extrapolated from measured data

Algorithm 2.1 Single Reflection QD Generator

```
function GetMpcsFirstReflection(Cursor: $\tau_0$, $PG_{0,D}$, $AoD_{az/el,0}$, $AoA_{az/el,0}$, Material)
    $RL \leftarrow \mathcal{R}(s_{RL,Material}, \sigma_{RL,Material})$
    $PG_0 = PG_{0,D} - (RL - \mu_{RL})$
    PreCursors ← ComputePre/PostCursors($\tau_0$, $PG_0$, $AoD/AoA_{az/el,0}$, Material)
    PostCursors ← ComputePre/PostCursors($\tau_0$, $PG_0$, $AoD/AoA_{az/el,0}$, Material)
        return PreCursors, Cursor, PostCursors

function ComputePre/PostCursors($\tau_0$, $PG_0$, $AoD/AoA_{az/el,0}$, Material)
    $\lambda \leftarrow \mathcal{R}(s_{\lambda,Material}, \sigma_{\lambda,Material})$
    $\Delta_i \leftarrow \mathcal{E}(\lambda), \quad i = 1, \dots, N_{pre/post}$
    $\tau_i = \tau_0 \pm \sum_{j=1}^{i} \Delta_i$        ▷ Add for post-cursors, subtract for pre-cursors
    Remove pre-cursors with $\tau_i < 0$, update $N_{pre/post}$
    $K_{dB} \leftarrow \mathcal{R}(s_{K,Material}, \sigma_{K,Material})$
    $\gamma \leftarrow \mathcal{R}(s_{\gamma,Material}, \sigma_{\gamma,materia})$
    $\sigma_{s,Material} \leftarrow \mathcal{R}\left(s_{\sigma_{s,Material}}, \sigma_{\sigma_{s,Material}}\right)$
    $S_i \leftarrow \mathcal{N}\left(0, \sigma^2_{s,Material}\right)$
    $PG_i = PG_{0,dB} - K_{dB} - 10\log_{10}(e)\,\frac{|\tau_i - \tau_0|}{\gamma} + 10\log_{10}(e)\,S_i$
    Remove MPCs with $PG_i \geq PG_0$, update $N_{pre/post}$
    $\sigma_\alpha \leftarrow \mathcal{R}(\mu_{\sigma_\alpha}, \sigma_{\sigma_\alpha})$
    $\alpha_{AoD/AoA,az/el,i} \leftarrow \mathcal{L}(0, \sigma^2_\alpha)$
    $AoD/AoA_{az/el,i} \leftarrow AoD/AoA_{az/el,0} + \alpha_{AoD/AoA,az/el,i}$
    Wrap angles in $az = [0, 360)$, $el = [0, 180]$
    $\phi_i \leftarrow \mathcal{U}[0, 2\pi)$
        return ($\tau_i$, $PG_i$, $AoD/AoA_{az/el,i}$, $\phi_i$)
```

from [35], during the QD model generation some of them may not follow some basic assumptions. For example, when a D-ray has a delay τ_0 close to 0, some of its generated pre-cursors might arrive before the direct ray itself. Since this situation cannot happen in the physical reality, rays with $\tau_{i,pre} < 0$ are removed and N_{pre} is consequently updated.

The path gain of the D-ray is

$$\mathbf{PG}_0 = 20\log_{10}\left(\frac{\lambda_c}{4\pi\ell_{ray}}\right) - \mathbf{RL}_{dB}, \tag{2.8}$$

where λ_c is the wavelength of the carrier frequency, ℓ_{ray} is the total ray length, and $\mathbf{RL} \sim \mathcal{R}(s_{RL}, \sigma_{RL})$ is the random reflection loss factor given by the reflecting surface's material. If only the deterministic part of the ray-tracer is considered, the path gain $PG_{0,D}$ only includes the mean reflection loss μ_{RL}.

Once the arrival times $\boldsymbol{\tau}_i$ are known, the path gains for the MPCs can be computed as

$$\begin{aligned}\mathbf{PG}_{pre/post,i,\mathrm{dB}} = \boldsymbol{PG}_{0,\mathrm{dB}} - \boldsymbol{K}_{pre/post,\mathrm{dB}} + \\ - \frac{|\boldsymbol{\tau}_{i,pre/post} - \tau_0|}{\boldsymbol{\gamma}_{pre/post}}(10\log_{10} e) + \\ (10\log_{10} e)\mathbf{S}_{pre/post},\end{aligned} \tag{2.9}$$

where

- $\mathbf{K}_{pre/post,\mathrm{dB}} \sim \mathcal{R}\left(s_{K_{pre/post}}, \sigma_{K_{pre/post}}\right)$ is a loss factor,

- $\boldsymbol{\gamma}_{pre/post} \sim \mathcal{R}\big(s_{\gamma_{pre/post}}, \sigma_{\gamma_{pre/post}}\big)$ is the power-delay decay constant,
- $\mathbf{S}_{pre/post} \sim \mathcal{N}\Big(0, \boldsymbol{\sigma}^2_{s,pre/post}\Big)$ is the standard deviation of the power-delay decay, where $\boldsymbol{\sigma}_{s,pre/post} \sim \mathcal{R}\big(s_{\sigma_{s,pre/post}}, \sigma_{\sigma_{s,pre/post}}\big)$.

While $\mathbf{K}_{pre/post,\text{dB}}$, $\boldsymbol{\gamma}_{pre/post}$, and $\boldsymbol{\sigma}_{s,pre/post}$ are independent across clusters, and $\mathbf{S}_{pre/post}$ is independently extracted for each MPC.

Since the main cursor is the one with the maximum PG when extracting the statistics from the measurements, MPCs with $\mathbf{PG}_{pre/post,i} \geq PG_{0,D}$ are removed, updating, in this case, $N_{pre/post}$.

Finally, the angle of departure in azimuth (and similarly the AoD in elevation and the AoAs in azimuth and elevation) of the MPCs are computed as

$$\mathbf{AoD}_{az,i} = AoD_{az,0} + \boldsymbol{\alpha}_{AoD,az,i}, \tag{2.10}$$

where $\boldsymbol{\alpha}_{AoD,az,i} \sim \mathcal{L}\Big(0, \boldsymbol{\sigma}^2_{\alpha_{AoD,az}}\Big)$ is the angle spread. The variance is itself a random variable independently extracted for each cluster, i.e., $\boldsymbol{\sigma}^2_{\alpha_{AoD,az}} \sim \mathcal{R}\Big(s_{\sigma^2_{\alpha_{AoD,az}}}, \sigma_{\sigma^2_{\alpha_{AoD,az}}}\Big)$.

Finally, the phase shift $\boldsymbol{\phi}_i$ due to both diffusion and Doppler shift is considered $\mathcal{U}[0, 2\pi)$ independently for each diffuse MPC.

Higher-order reflections For the n^{th} reflection order, with $n > 1$, multiple heuristics can be thought of to compute the diffuse components. Unfortunately, the measurements taken and the models adopted to process them do not allow for a reliable confirmation of the proposed heuristics, but an extension to higher reflection orders is nevertheless needed for inclusion in a generic ray-tracer.

The path gain for specular rays with n reflections is extended as follows:

$$\mathbf{PG}_0 = 20\log_{10}\left(\frac{\lambda_c}{4\pi\ell_{ray}}\right) - \sum_{i=1}^{n} \mathbf{RL}_{i,dB}, \tag{2.11}$$

where $\boldsymbol{RL}_{i,dB} \sim \mathcal{R}(s_{RL,i}, \sigma_{RL,i})$, and $(s_{RL,i}, \sigma_{RL,i})$ refers to the statistics associated to the material of the i-th reflector of the given ray.

We propose two simple heuristics: a complete multiple reflection QD model and a reduced multiple reflection QD model.

Complete multiple reflection QD model Upon the first scattering event, all components produced – both specular and diffuse – behave as independent components and their remaining paths are traced accordingly. We assume that every diffuse ray closely follows the path of the main cursor and further generates $N_{pre} + N_{post}$ diffuse MPCs at each bounce. The total number of MPCs generated by a single deterministic rays at the n-th reflection will thus be $N_{MPC} \sim (N_{pre} + 1 + N_{post})^n$.

Algorithm 2.2 Reduced Multiple Reflection QD Generator

```
function GetMpcsMultipleReflection(Cursor, MaterialList, MaterialLibrary)
    CursorOutput ← Cursor
    for Material $\in$ MaterialList
        OtherMaterialsList ← MaterialList $\setminus$ {Material}
        PreCursors, PostCursors ← $\varnothing$
        CurrentPreCursors, CursorOutput, CurrentPostCursors ← GetMpcsFirstReflection(CursorOutput, Material)
        PreCursors ← Concatenate(PreCursors, OtherMaterialsReflLoss(CurrentPreCursors, OtherMaterialsList, MaterialLibrary))
        PostCursors ← Concatenate(PostCursors, OtherMaterialsReflLoss(CurrentPostCursors, OtherMaterialsList, MaterialLibrary))
    return PreCursors, CursorOutput, PostCursors

function OtherMaterialsReflLoss(Cursors, OtherMaterialsList, MaterialLibrary)
    for Cursor $\in$ Cursors
        for Material $\in$ OtherMaterialsList
            $RL \leftarrow \mathcal{R}(s_{RL,Material}, \sigma_{RL,Material})$
            Cursor.$PG$ ← Cursor.$PG + (RL - \mu_{RL,Material})$
    return Cursors
```

Reduced multiple reflection QD model In order to reduce the exponential complexity of the complete model, the reduced model neglects diffuse rays beyond the first order given their multiplicatively high attenuation. Instead, only diffuse rays generated directly by the deterministic ray are taken into account, each generated with the QD parameters associated to the impinging reflecting surface. Moreover, we assume that every diffuse component closely follows the main cursor, thus reflecting on the same reflectors (see Alg. 2.2). Consequently, every reflector produces $N_{pre} + N_{post}$ diffuse components, thus yielding a maximum of $N_{MPC} \sim n(N_{pre} + N_{post}) + 1$, including the deterministic ray and possible rays discarded during their generation (see Sec. 2.2.4).

2.2.5 Comparison of RT and QD Models with Measurements

Given the structure of this QD model, every material must have a set of parameters for it to be appropriately simulated. It follows that given the CAD file of an environment, every surface must be associated with a material with all the necessary simulation parameters taken, for example, from a material library.

We report in the following tables examples of material libraries from NIST's Lecture Room, reformulating the mean and variance provided per material [35] into the s and σ parameters needed to generate the random parameters of the model. Measured data were taken from different TX positions pointing towards the center of the room, where a mobile RX sounder moved around the tables. Specifically, as shown in Fig. 2.4, considering the bottom-left corner as the origin $(x_0, y_0, z_0) = (0, 0, 0)$, TX$_1$ is positioned in $(2, 3, 2.5)$ m, TX$_2$ in $(8, 3, 2.5)$ m, TX$_3$ in $(8, 17, 2.5)$ m, TX$_4$ in $(2, 17, 2.5)$ m, and the RX performs a loop around the table.

Given that the channel sounder's TX had a limited angular Field-of-View (FoV), it was possible to characterize different surfaces, e.g., different walls by varying the TX positions during the measurement campaign. The model parameters per position have been reformatted accordingly in Table 2.1. Please note that, given the geometry of the room and the limited FoV,

Tab. 2.1: NIST's Lecture Room material library.

		Left Wall (TX$_2$)	Bottom Wall (TX$_3$)	Right Wall (TX$_1$)	Top Wall (TX$_1$)	Tables (TX$_1$)	Ceiling (TX$_1$)
$K_{dB} \sim \mathcal{R}(s,\sigma)$	$(s_{K_{pre}}, \sigma_{K_{pre}})$	(5.1196, 1.7485)	(1.4809, 2.1325)	(0, 0)	(0.5913, 4.5206)	(0, 0)	(3.6167, 7.2715)
	$(s_{K_{post}}, \sigma_{K_{post}})$	(6.2208, 3.5421)	(7.1809, 2.5325)	(0.2641, 3.1699)	(0.33, 3.7213)	(3.7738, 1.8748)	(7.1103, 2.2712)
$\gamma \sim \mathcal{R}(s,\sigma)$	$(s_{\gamma_{pre}}, \sigma_{\gamma_{pre}})$	(0.6742, 0.9992)	(0.9006, 0.2325)	(0, 0)	(0.0094, 0.2285)	(0, 0)	(0.9595, 0.901)
	$(s_{\gamma_{post}}, \sigma_{\gamma_{post}})$	(0.0658, 1.2034)	(0.6881, 0.3566)	(0.0412, 0.8648)	(0.0792, 1.1572)	(0.53, 0.4837)	(0.0717, 1.2794)
$\sigma_s \sim \mathcal{R}(s,\sigma)$	$(s_{\sigma_{s,pre}}, \sigma_{\sigma_{s,pre}})$	(0.0119, 0.3087)	(0.5553, 0.129)	(0, 0)	(0.243, 0.273)	(0, 0)	(0.2122, 0.0935)
	$(s_{\sigma_{s,post}}, \sigma_{\sigma_{s,post}})$	(0.4144, 0.1507)	(0.26, 0.1003)	(0.6367, 0.3209)	(0.201, 0.1901)	(0.3309, 0.4614)	(0.7679, 0.2484)
$\lambda \sim \mathcal{R}(s,\sigma)$	$(s_{\lambda_{pre}}, \sigma_{\lambda_{pre}})$	(0.9775, 0.3449)	(0.9172, 0.2241)	(0, 0)	(0.619, 1.1299)	(0, 0)	(0.8119, 0.2421)
	$(s_{\lambda_{post}}, \sigma_{\lambda_{post}})$	(0.8153, 0.6948)	(1.4106, 0.5832)	(0.9879, 0.4235)	(0.8655, 0.3762)	(0.8099, 0.076)	(0.7785, 0.1426)
$\sigma_\alpha \sim \mathcal{R}(s,\sigma)$	$(s_{\sigma_{\alpha,az}}, \sigma_{\sigma_{\alpha,az}})$	(0.1016, 2.2504)	(1.9426, 1.5726)	(3.2889, 1.3202)	(2.117, 2.1206)	(1.6594, 3.1974)	(1.9829, 0.9094)
	$(s_{\sigma_{\alpha,el}}, \sigma_{\sigma_{\alpha,el}})$	(2.9947, 1.6613)	(2.6946, 1.3948)	(3.2812, 1.8865)	(2.741, 1.7964)	(4.0345, 2.6859)	(2.696, 1.1135)
$RL \sim \mathcal{R}(s,\sigma)$	(s_{RL}, σ_{RL})	(9.8412, 3.4424)	(8.5025, 4.2343)	(10.1562, 3.5164)	(6.7238, 5.9352)	(5.2106, 3.4013)	(6.5833, 2.1943)
	μ_{RL}	10.7	9.84	10.8	9.27	6.58	6.9

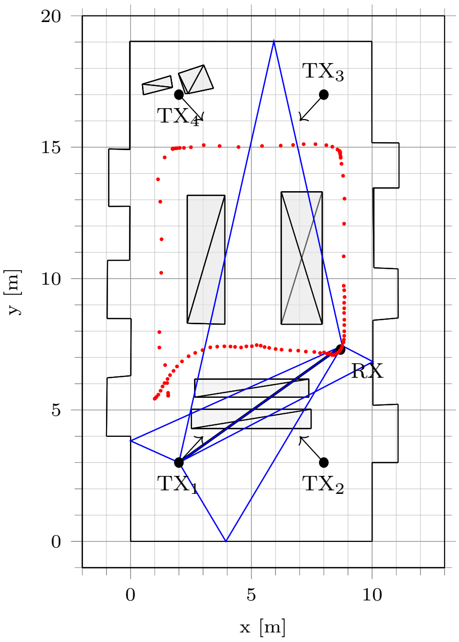


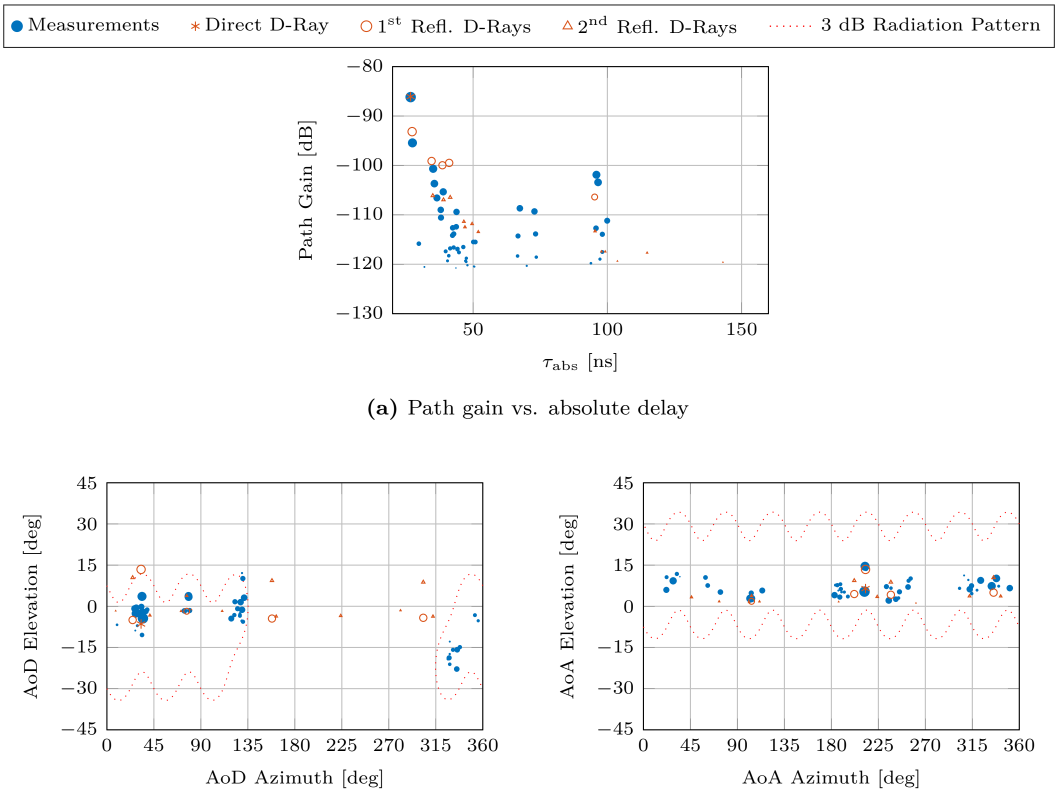
Fig. 2.4: CAD model of NIST's lecture room. The 108 RX positions from the measurement traces are shown in red. As an example, the direct and first reflection rays generated with the RT for TX$_1$ and the specific RX position are shown in black and blue, respectively.

it was not possible to properly characterize some materials, in particular the floor [35]. Since no characterization was available from the measurements, no diffuse components were generated and the statistics for the reflection loss were copied from the ceiling, as this is the most similar material in the available library.

Fig. 2.5 shows an example of measured channel compared to the deterministic ray-traced channel for the scenario of Fig. 2.4. As can be seen, the direct ray is correctly identified both in the power-delay domain and in the angular domains, while other rays only partially resemble the measurements. This is due to (i) the approximate CAD model which may be missing some relevant reflectors and (ii) inaccuracies in the measurements.

While delays shown in Fig. 2.5a are in good accord between measurements and RT simulation, path gains are less precise, due to the random reflection losses experienced by the rays. Notice also that the TX only has antennas towards the front (as shown by the antenna pattern in Fig. 2.5b), thus, rays predicted by the RT to depart with an azimuth angle between 135° and 315° were not part of the real measurements. Most of all, though, it is easily noticeable that there exist clusters of rays well defined in the joint path gain, delay, AoD, AoA domain, and are missing, instead, in the channel generated by the RT. Such clusters do not arise from higher order reflections (not shown here), but rather from diffuse MPCs, thus highlighting the need for a valid diffuse QD model.

Figs. 2.6 and 2.7 show how the proposed QD model enhances the realism of a purely deterministic channel, making it significantly more similar to the measured one. Specifically, Fig. 2.6



(a) Path gain vs. absolute delay

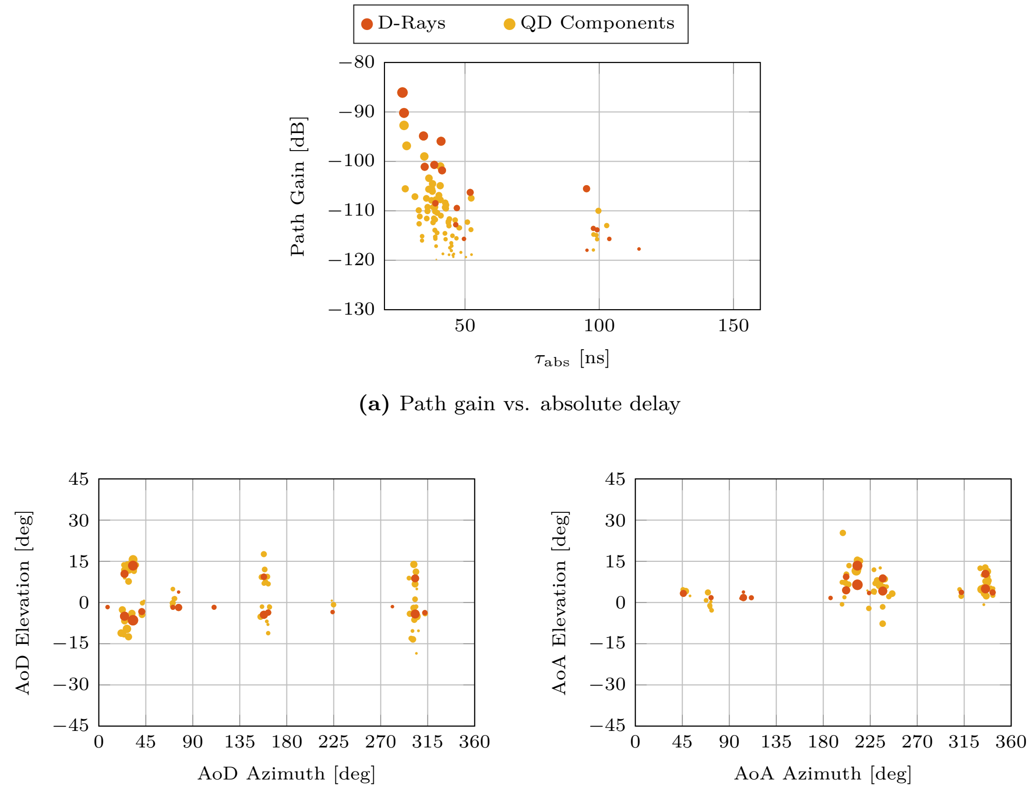
(b) AoD

(c) AoA

Fig. 2.5: Example of comparison between measurements and ray-tracer, based on the channel between TX$_1$ and the RX shown in Fig. 2.4 in the bottom left corner of the loop. In a, τ_{abs} represents the absolute delay of each ray. b and c show the 3 dB radiation patterns of the channel sounders described in [35] approximated with Gaussian beams. In fact, MPCs outside of these regions are not detected in the measurements.

reports an example of a specific channel instance, based on the CAD model shown in Fig. 2.4 and for the same TX/RX locations of Fig. 2.5. With respect to the RT specular reflections from Fig. 2.5, the deterministic rays (in orange), which are generated up to second order reflections, also include a random reflection loss component in the path gain. The diffuse rays added to the model are plotted in blue, with sizes proportional to the respective path gain. By comparing Fig. 2.6 with Fig. 2.5, it is clear that the D-rays alone are not able to fully model the complexity of a real channel, and that the proposed QD model can instead play an important role in this regard. In fact, empirically, rays are parts of clusters with small variations in the angular and delay domains, and large variations in the power gain domain.

Furthermore, the effects of the added rays are clearly shown in Fig. 2.7, which plots the Cumulative Distribution Functions (CDFs) of the path gain (Fig. 2.7a), the absolute delay (Fig. 2.7b), and the RMS delay spread (Fig. 2.7c), similar to the RMS angle spread shown in [35], for the multipath components of the scenarios. The CDFs show the combined statistics of the mmW channel between TX$_1$ and 108 RX positions shown in red in Fig. 2.4). Notably, it



(a) Path gain vs. absolute delay

(b) AoD

(c) AoA

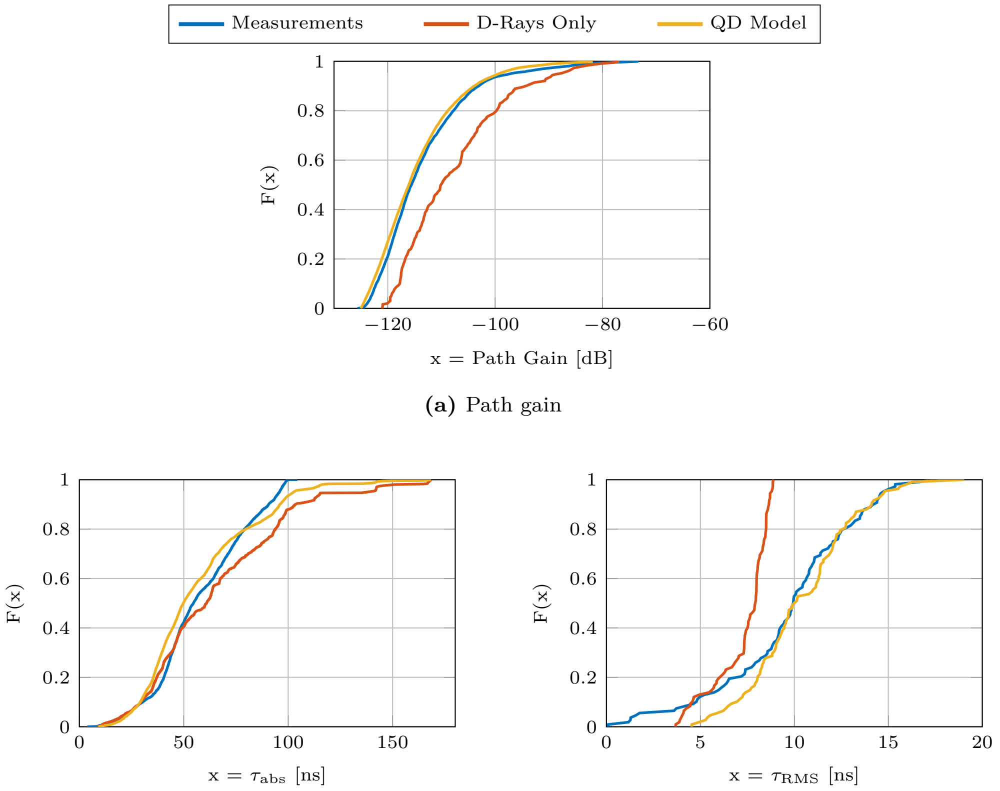
Fig. 2.6: Reduced multiple reflection QD model applied to RT-based channel traces with up to 2$^{\text{nd}}$ order reflections. Rays with path gain below -120 dB are not shown, to more closely resemble the dynamic range of the channel sounder.

is clear how the delays and path gains generated with the proposed QD model are significantly closer to the real measurements with respect to purely deterministic rays alone, with CDF fit improvements from 73 % to 86 % (i.e., Kolmogorov-Smirnov (KS) test improvements of 0.13) for the path gain, from 86 % to 89 % (i.e., KS test improvements of 0.03) for the absolute delay, and from 33 % to 87 % (i.e., KS test improvements of 0.54) for the RMS delay spread.

Finally, the difference in RMS delay spread, especially when the QD model is not used, can be due to the numerous reflections from objects which are not present in the CAD model but are instead part of the measured scenario, e.g., chairs and other small details of the room.

2.2.6 Blockage in Millimiter Waves

With respect to the first project report, we further extended the literature review on signal blockage characterization, especially in light of the implementation of a new software able to process ray-traced channels to include obstructions (see Sec. 2.5 for further details). Our experience on this software guided our review of the current state of the art, trying to find works and standards describing reliable model(s) to be recreated in our simulator, rather than



(a) Path gain

(b) Absolute Delay

(c) RMS delay spread

Fig. 2.7: Comparison between CDFs of MPC path gain, absolute delay, and RMS delay spread with and without QD model with respect to the measurements.

focusing on theoretical or measurement-focused works.

ITU Recommendations We found several International Telecommunication Union (ITU) Recommendations focusing on complex channel propagation, also in the mmW band. Specifically:

- Recommendation ITU-R P.526 [63]: it mainly discusses signal propagation over long ranges, where the Earth curvature and buildings might create strong diffractions, making the diffracted path one of the strongest components, at least for sub-6 GHz communications. It provides simple formulas to compute the Fresnel zones by approximating the complex Fresnel integral
$$F(\nu) = \int_0^\nu e^{j\frac{\pi s^2}{2}} \mathrm{d}s$$
generally used in diffraction calculations, and practical models for automated diffraction loss calculation of a generic path (based on the Bullington model [64]). The Recommendation includes detailed formulas and pseudo-algorithms, which should make a practical implementation fairly easy. Unfortunately, it is not specifically tailored to mmW com-

munications, nor does it include typical higher-frequency propagation phenomena such as human blockage, often modeled using Double Knife-Edge Diffraction (DKED).

- Recommendation ITU-R P.833 [65]: it provides practical models for both single vegetation obstructions (mainly using DKED, referencing the model in [63] for detailed diffraction formulas) and woodlands (based on stochastic models). Depolarization and dynamic effects (e.g., wind on vegetation causing highly dynamic fading) are also described. The frequency ranges from 30 MHz to 100 GHz, making the model suitable for mmWave communication. Furthermore, the Recommendation incorporates several tables with parameters for different types of vegetation, allowing the end user to diversify the vegetation in a given scenario.

- Recommendation ITU-R P.1238 [66]: it mostly describes stochastic models for indoor propagation for frequencies between 300 MHz and 450 GHz. The possibility of modeling specific environments using ray-tracing is also discussed. Specifically, given the nature of the wireless channel in indoor environments, the Recommendation suggests to also include diffraction to properly characterize the channel. Sec. 9 discusses the effects that moving objects in the room might have on the channel, together with a stochastic model accounting for human interaction over long periods of time.

- Recommendation ITU-R P.1410 [67]: it provides a very detailed measurement-based model for terrestrial broadband radio access systems operating in a frequency range from 3 to 60 GHz. It also refers to [65] to include an accurate vegetation model. Furthermore, it describes reflection, scatter loss, and diffraction with ready-to-use formulas for implementing a Radio Frequency (RF) ray-tracing software. Finally, it includes stochastic models for transmission-through-buildings and precipitation loss.

- Recommendation ITU-R P.2040 [68]: it provides an overview of building materials and their effects on propagation, specifically transmission and reflection losses. Electric parameters for many common building materials are also included, allowing the user to include this level of details into a ray-tracing software, and while designing a CAD environment.

METIS project Another prominent source of models was provided by the Mobile and wireless communications Enablers for the Twenty-twenty Information Society (METIS) project, as summarized in the following items.

- METIS Channel Models [69]: Sec. 6 shows a step-by-step algorithm to implement an RT-like map-based channel model. Specifically, Step 7 of Sec. 6 discusses the Knife-Edge Diffraction (KED) model and provides equations to compute it precisely. Step 10 provides two alternative methods for multiple diffractions: a first option uses Berg's recursive model [70], while a second possibility is given by Uniform Theory of Diffraction (UTD) formulas. Step 11 discusses isotropic scattering and scattering from large rough surfaces (e.g., buildings facades). Furthermore, Secs. 8.2 and 8.3 give further guidelines for the correct usage of the map-based and the hybrid METIS Models, as well as technical details about the map-model scenario in Appendix C. In the appendices, some relevant scenarios are also discussed, such as a body blocking scenario (Appendix A.4.3), and detailed propagation scenarios (Appendix B).

- METIS Simulation Guidelines [71]: Secs. 4 and 9 list a number of simulation scenarios. Sec. 9.2 is especially interesting. Specifically, a scenario called *"Dense urban information society"* is described, together with traffic models, user distribution and mobility, and base station deployment options.

3GPP Model Notably, the 3GPP channel model [24] refers to the METIS project for proposing a map-based hybrid channel model, using [69] for the deterministic component of the channel, and adding its stochastic model based on random clusters of rays. The standard also suggests a number of references to compute the different propagation interactions, such as free-space LoS [72], geometry and power computation of specular reflections [73], diffraction geometry [74] and power computation [75], wall penetration geometry [76], and power computation [73]. The METIS project [69] is also referenced to for the computation of scattering upon small objects, considering both isotropic scattering and Radar Cross-Section (RCS) scattering.

The 3GPP channel model [24] also includes possible blockage models. It proposes a stochastic blockage model (Blockage model A) where random rectangular regions are blocked, also considering temporal and spatial correlation for completeness. It also suggests a geometric blockage model (Blockage model B) where rectangular screens are deployed, specifying their relative dimensions and mobility patterns. In both cases, a KED at four edges is defined, using the arctan approximation for the diffraction loss.

Scientific literature Looking through scientific conference proceedings, journals, and Ph.D. dissertations, we found several works on the topic, although harder to implement and usually very specific and hardly generalizable.

We explored a Ph.D. dissertation focusing on radio channel simulation [77], even though its extremely theoretic formulation makes it hard to extract a practical low-complexity computational model. Still, the author assesses the validity of the KED model when the obstacle is narrow enough, using specific solvers for Maxwell's equations.

In [78], the authors show measurements taken at 10.5 GHz in which a metal cylinder affects the radio channel similarly to a human body. This result justifies the usage of such a simple model in practical simulations rather than modeling a detailed person, a significantly harder task to perform and with a much higher computational cost.

The authors of [79] define *virtual scattering centers* to model scattering from a vehicle. Thus, rather than introducing a complex model with a high polygon count, they propose to reduce the complexity of simulations involving cars by only considering a few key points producing scattering, e.g., wing mirrors.

In [80], the authors describe the KED model in detail. Related work [81] further explores the impact of human blockage on the channel model specifically proposed for IEEE 802.11ad [82], based on measurements at 60 GHz.

In [83] the authors propose a low-complexity Fourier-based model for modeling human blockage, which produces a shadowing loss closer to the Fresnel formulas for DKED than the simpler arctan model.

Furthermore, the authors in [84] analyze accurate measurements of human body blockage at 60 GHz and showed that, when an obstacle is close to the path of the direct ray, it also creates a strong reflected ray, thus acting as a 2-ray propagation model. Electrical parameters for the three persons performing the experiment were estimated and provided. The authors also modified the DKED model to better fit the measurements, although the proposal can hardly be generalized, as the model varies as a function of the separation distance between the communicating nodes: however, numerical results are given for distances equal to 4, 6, and 8 m.

Unfortunately, the literature did not provide any reference to the effect of obstructions on reflected rays. We might try to fill this gap by studying ourselves the impact of modeling obstructions and diffractions also for secondary rays; we will investigate whether this makes sense, also in view of the higher computational complexity expected to perform such task.

2.3 Simplification of a Stochastic SCM: 3GPP TR 38.901

In this section, we investigate whether it is possible to simplify the structure of the widely used 3GPP stochastic SCM [24] without compromising the accuracy with respect to the original model, used as a baseline. First, we profile the computational complexity of the model in Sec. 2.3.2 and show that the computations with complex values related to the generation of steering vectors are the main factor that affects the time to generate an instance of the SCM channel. We then proceed to simplify these calculations by removing clusters and subpaths (i.e., spatial components of the channel) and comparing the performance of the baseline and simplified models in Sec. 2.3.3. We show that some metrics (e.g., the distribution of the Signal-to-Interference-plus-Noise Ratio (SINR) in a typical 3GPP scenario [24]) are not (or only marginally) affected by this simplification, while the channel generation time reduces by up to 12.5 times. We also highlight the limitations that such a simplification introduces, and give insights on when it may be legitimate to use the simplified version of the model.

2.3.1 An Introduction to 3GPP TR 38.901

The entry (i, j) of the channel matrix at a given tap delay is given by the combination of N clusters, which model different angular components of the channel between the two transceivers. The power of each cluster is modeled through an exponential power delay profile, which depends on the delay with which each of the different clusters arrives at the receiver. Therefore, the LoS path (if present) is the strongest cluster, associated to the minimum delay, followed by several reflections. Additionally, each cluster can be modeled by the superposition of M subpaths*, which are distributed with certain statistics around the AoA and AoD of the cluster.

A single realization of the matrix $\mathbf{H}$ depends on the combination of large scale and fast fading parameters. The first have an impact on the power delay profile, the angular distribution, the relative strength of the LoS component with respect to the NLoS reflections, and the shadowing. Large scale parameters generally depend on the scenario that is being modeled. Fast fading, instead, models small variations in the channel, e.g., the Doppler spread introduced by the user mobility. The actual parameters may vary in different SCMs, and are generally expressed through random distributions that fit data collected in measurement campaigns.

2.3.2 Analysis and Profiling

In order to proceed with the simplification of the model, an initial analysis is necessary to understand which are the most computationally demanding steps in the channel generation process. To remove the dependency on the implementation as much as possible and decouple the model complexity from the implementation inefficiencies, a 3GPP-compliant [24, 85] network simulator was designed and optimized. With this tool, we verified experimentally that the channel matrix generation takes up to 90% of the simulation time (the remaining overhead is given by the scenario definition, user mobility, beamforming vector computations, statistical computation, among others). Although the performance is implementation-dependent, analyzing how the computation of the different parts contributes to the overall simulation time allows drawing general conclusions.

When antenna arrays are considered, the channel model can no longer be expressed through a time-varying scalar impulse response. Rather, as discussed in Sec. 2.3.1, the channel response is enclosed in a matrix that associates each of the S elements of the transmitting array, to each of the U elements of the receiving array. As channel models generate a number of rays coming from different directions, a way to translate such directionality into the definition of the channel matrix is needed. Considering a narrow-band signal and a small-aperture antenna, the incoming signal seen from the point of view of any given antenna element will be a phase-shifted copy of the original signal. Steering vectors are used to represent this phase shift over all array elements and are thus composed of complex phase shifts. Please note that this concept is valid for clusters incoming (or departing) from any direction and for arbitrary arrays. As mmW use cases are expected to be mainly focused on arrays with tens or hundreds of antennas, the code has been optimized for such scenarios, partially degrading the performance when a small number of antennas (e.g., one at both transmitter and receiver) is used.

Our profiling highlights that the computations related to steering vectors and their combination are the most time-consuming part of the generation of an instance of the matrix representing the 3GPP channel, as reported in the *Computations* entry in Fig. 2.8. For a channel with a single element (i.e., $\mathbf{H} \in \mathcal{C}^{1\times 1}$), the computation takes 79.42% of the time. This percentage increases

*The 3GPP specifications refer to subpaths as rays.

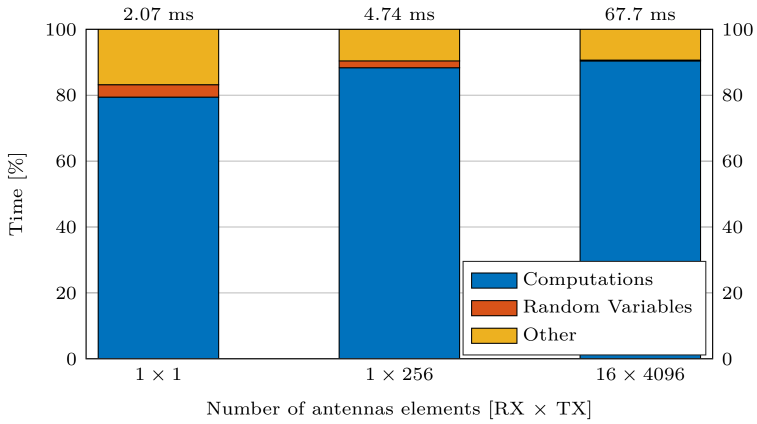


Fig. 2.8: Results from the profiling of the 3GPP channel model described in 3GPP TR 38.901, for different square antenna arrays at the transmitter and the receiver. We report the percentage of different tasks related to the channel matrix generation, and the absolute execution time above each bar. The *Computations* term only includes operations related to steering vectors, whose complexity increases with the channel matrix's size. As the size of the channel matrix increases, operations with complexity proportional to its size dominate over the overall generation time.

up to 90.38% for the largest antenna array configuration we consider (i.e., 16 × 4096).* The generation of random variables, such as the cluster powers, the delays, the sub-paths' angles, the phase shifts, and the AoA/AoD coupling, is instead negligible, particularly when large arrays are considered (0.27% for the 16 × 4096 configuration). On the contrary, the code overhead, composed of sub-routine calls and all other operations, is significant and does not considerably depend on the array size.

As the *Computations* entry is related to the generation and combination of the steering vectors, it is proportional to the number of clusters and subpaths that are generated and combined: the richer the channel, the slower this computation. For example, as reported in Fig. 2.8, a channel with a single entry would take up to 1.64 ms to perform the computations for a total of 2.07 ms, while for the largest antenna configuration, yielding the largest channel matrix, computations take 61.19 ms for a total of 67.7 ms.

2.3.3 Channel Simplification Results

As shown by the analysis in Sec. 2.3.1, reducing the number of clusters and subpaths can be beneficial in terms of simulation time. Nevertheless, changes in the channel model may affect or even compromise its reliability, depending on the application. In this section, we analyze the effects of the simplification on the network statistics obtained from the network simulator described in Sec. 2.3.2. The models with the original number of clusters and sub-paths were considered as baselines with which to compare the effects of the simulations, i.e., we do not provide in this thesis a direct comparison with ground truth measurements. Thanks to the

*Notice that the dependence between the number of antenna elements and the *Computations* entries is not linear, as MATLAB introduces optimizations for large matrices.

flexibility of our simulator, it was possible to perform the tests on different configurations of the 3GPP channel model while keeping the same settings for the cellular scenario. For this study, a 3GPP-compliant Urban Macro (UMa) downlink scenario is considered [24]. Similar results can also be obtained for other scenarios.

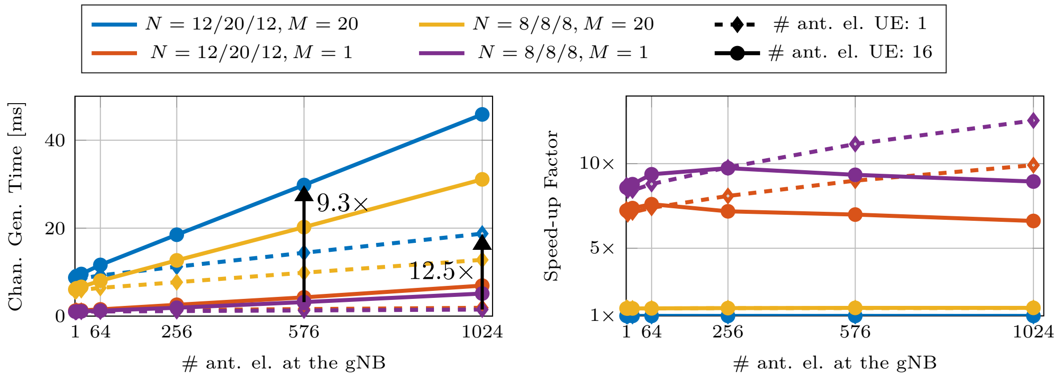
The 3GPP channel model differentiates among three states of the channel, namely LoS, NLoS, and Outdoor-to-Indoor (O2I). Different channel states correspond to a different number N of clusters, whereas the number of sub-paths per cluster $M = 20$ is kept fixed for all propagation conditions. As per [24], $N_{LoS} = 12$ clusters are present in LoS, $N_{NLoS} = 20$ in NLoS, and $N_{O2I} = 12$ in O2I channel conditions, in short $N = 12/20/12$.

We followed two complementary simplification strategies: on one hand, reducing the number of clusters, and on the other hand, reducing the number of sub-paths per cluster. Indeed, it was possible to vary the latter from $M = 20$ to $M = 1$, corresponding to a cluster with only the main path and no sub-paths. On the contrary, in [24], the azimuth and elevation angle spreads are specified only for some specific cluster configurations, and cannot be trivially interpolated to extract the parameters for the configurations that are excluded from the model. Therefore, this limits the extension of our simplification to clusters. Specifically, the channel model was complete enough to allow only a maximum reduction down to $N_{LoS} = 8$, $N_{NLoS} = 8$, $N_{O2I} = 8$, besides the default one.

Considering the different contributions to the computational complexity discussed in Sec. 2.3.2, the speed-up factor should be proportional to the reduction of the overall number of clusters and/or sub-paths. However, several additional aspects need to be taken into account, making the dependence on the number of clusters and sub-paths not necessarily linear. Particularly, decreasing N for one channel state will contribute proportionally to the number of users that are in that propagation condition. In the considered scenario, which follows the specifications in [24], 80% of the users, being indoor, are in O2I conditions, making N_{O2I} the most significant term to reduce. Moreover, depending on the implementation and on the initial access policy, one may need to consider only the users who are connected to a Next Generation Node Base (gNB), adding a further layer of complexity to these considerations. In our simulator, the attachment is purely based on the combination of pathloss and shadow fading, and the channel is computed only for the users that successfully connect to a gNB.

We evaluated the various (N, M) configurations for different array sizes, two for the User Equipments (UEs) and five for the gNBs, to test our approach in multiple settings. Array sizes were chosen following typical values found in the literature, and scenarios with both single- and multi-antenna UEs were tested. In Figs. 2.9a and 2.9b, the generation time and speed-up factors with respect to the baseline configuration $N = 12/20/12$ are shown. The generation time of a single channel matrix was reduced by a factor up to 12× for a single-antenna UE, going from 18.75 ms to 1.49 ms. Note that, according to the aforementioned considerations on the channel state distribution, the speed-up factor is not necessarily proportional to the reduction of the number of clusters.

We evaluated the effects of the simplifications on (i) the narrowband SINR, given its relation with channel capacity; (ii) the wideband Signal-to-Interference Ratio (SIR); and (iii) the



(a) Computation time

(b) Performance gain

Fig. 2.9: Computation time required to generate an instance of the channel matrix (a) and performance gain introduced by the simplification (b), as a function of the number of antenna elements at the gNB, for different configurations at the UE and different combinations of simplification parameters for the channel.

distribution of the singular values of the channel matrix, to show how spatial multiplexing is affected by our channel simplification. Defining $P_{\text{rx},t,r}$, P_N and $I_{\text{tot},t,r}$ as the powers of the received signal, the noise and the interfering signals, respectively, the narrow-band SINR, computed after the optimal Singular Value Decomposition (SVD)-based SISO beamforming, can be expressed as

$$\Gamma_{t,r} = \frac{P_{\text{rx},t,r}}{P_N + I_{\text{tot},t,r}}, \tag{2.12}$$

where $P_{\text{rx},t,r} = P_{\text{tx},t}\,\boldsymbol{w}_{t,r}^T \boldsymbol{H}_{t,r} \boldsymbol{w}_{r,t}$, with $P_{\text{tx},t}$ the transmit power of device t, $\boldsymbol{w}_{i,j}$ the beamforming vector used by device i to communicate with device j ; $P_N = N_0 BF$, with N_0 the noise power spectral density, B the communication bandwidth, and $F = 10^{\frac{\text{NF}}{10}}$ the noise figure of the receiver; $I_{\text{tot},t,r} = \sum_{m \neq t} P_{\text{tx},m}\,\boldsymbol{w}_{m,*}^T \boldsymbol{H}_{m,r} \boldsymbol{w}_{r,t}$, with $\boldsymbol{w}_{i,*}$ used with abuse of notation to indicate the beamforming vector used by device i to transmit towards a connected device, or $\boldsymbol{0}$ if i is not transmitting.

The wide-band SIR is defined as

$$\xi(f) = \frac{\left|H_{\text{rx},t,r}(f)\right|^2}{\left|\sum_{i=1}^{N_{\text{interf}}} H_{\text{interf},i}(f)\right|^2}, \tag{2.13}$$

where $H_{\text{rx},t,r}(f)$ is the receiver's channel frequency response and $H_{\text{interf},i}(f)$ are the channel frequency responses from the N_{interf} interfering base stations to the receiver.

For the wideband case, following [86], we consider two metrics that measure the impact of fading on the performance of the system. The Level Crossing Frequency (LCF) is defined as the fraction of Orthogonal Frequency Division Multiplexing (OFDM) subcarriers* in which the SIR $\xi(f)$ (as a function of frequency) crosses a given threshold ξ_{th} in the upward (or equivalently

*In our scenario, we consider a total bandwidth of 100 MHz, with subcarrier spacing equal to 60 kHz, as specified by the 3GPP for calibration at 30 GHz [24].

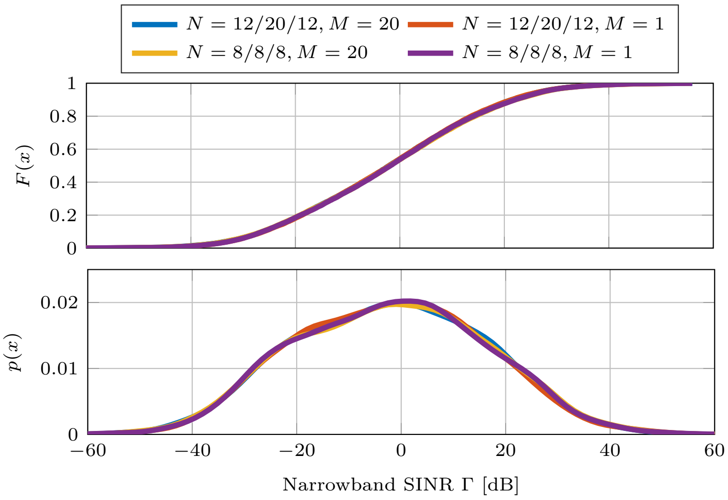


Fig. 2.10: Cumulative Distribution and Probability Density Functions of the narrowband SINR Γ of a scenario composed of UEs and gNBs with 16 and 64 antenna elements, respectively.

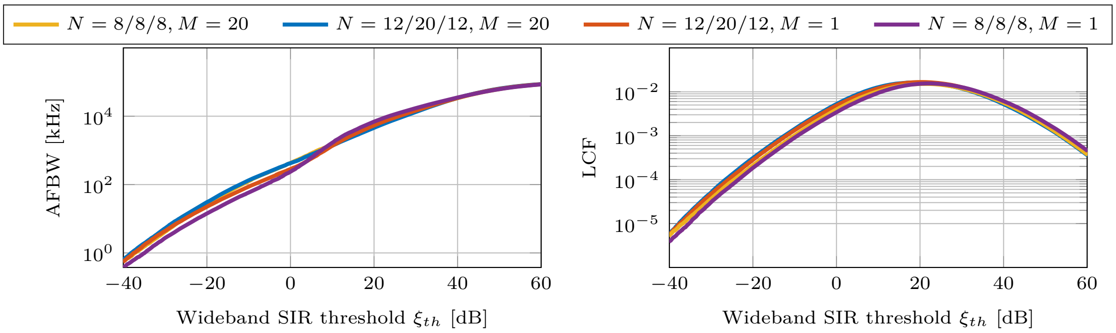
downward) direction. The Average Fading Bandwidth (AFBW) is defined as the average width (in kHz) of contiguous chunks of the overall bandwidth for which the envelope of $\xi(f)$ stays below a given threshold ξ_{th}.

Results show that narrow-band statistics are not affected by the channel simplification (Fig. 2.10), whereas the wide-band ones are only slightly affected by it. It is interesting to notice how, considering $N = 12/20/12$, $M = 20$ as the baseline, removing clusters almost does not affect the AFBW (Fig. 2.11a) while, on the contrary, the removal of sub-paths does not significantly affect the LCF (Fig. 2.11b). From Fig. 2.12 it can be noted that the mean ratio of the singular values of the channel matrices, while being the most diverging metric shown, still does not significantly differ from the baseline. The first singular value of the baseline model, however, is 14% smaller than that of the simplified channel. In any case, differences between the baseline and the most simplified channel (i.e., $N = 8/8/8, M = 1$) only have a minor effect on most metrics while speeding up the simulation by a factor of 10.

Thus, as shown by these results, reducing the number of clusters and sub-paths to the minimum allowed by the parameters found in [24] does not significantly change the system performance, while obtaining significant reduction of the computations. Unfortunately, it is not possible to push the simplification even further, while following the constraints of the parameters in the 3GPP specifications.

2.4 Simplification of an RT-based Channel Model

The accuracy of the MoI-based ray-traced channels – especially when considering the QD model – comes at a high computational cost, which may limit the scalability of the simulations, especially when considering a very large number of devices. In this perspective, the main objective of this



(a) Average Fading Bandwidth (AFBW) vs ξ_{th} for a scenario with UEs equipped with 16 antenna elements, and gNBs with 64 antenna elements.

(b) Level Crossing Frequency (LCF) vs ξ_{th} for a scenario with UEs equipped with 16 antenna elements, and gNBs with 64 antenna elements.

Fig. 2.11: Wideband results for the 3GPP TR 38.901 channel simplifications.

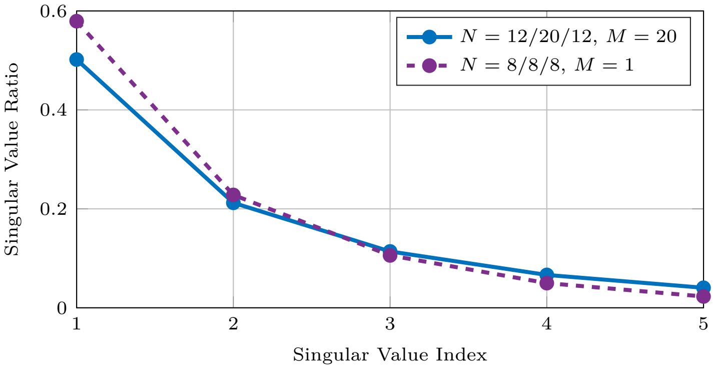


Fig. 2.12: Mean Singular Value Ratio of channel matrices for a scenario with UEs equipped with 16 antenna elements, and gNBs with 64 antenna elements. Only the baseline and the most extreme simplification are shown.

work is to evaluate how channel simplifications affect the results of link-level and network-level simulations while speeding up the overall simulation runtime. In this section, we present two techniques that were designed with this objective in mind [32]:

- *Maximum Reflection Order Reduction.* The overall computational complexity of a simulation of N TX/RX nodes lasting t time steps in a scenario composed of T triangles when considering up to R reflections is $\mathcal{O}\big(tN^2RT^{R+1}\big)$. In this approach, t and N are simulation parameters set by the user, T is determined by the CAD model of the simulated environment, while the maximum reflection order R for the ray tracing depends on the channel model. Understanding how different values of R affect the lower- and higher-layer performance metrics of the network with respect to the model complexity is the core of the first simplification strategy proposed in this work.

- *MPC Thresholding.* The second technique aims at reducing the number of rays between a pair of nodes. Specifically, given a set of M rays connecting two nodes, we propose and evaluate a selection criterion that decreases the number of MPCs to $M' < M$, to reduce the overall simulation time. This operation is applied on a time-step basis.

The rationale behind both strategies is to decrease the number of MPCs by removing the least significant ones, i.e., those with the lowest power, as they are expected to provide a limited contribution to the overall received signal strength.

Maximum Reflection Order Reduction Each reflection of the MPCs on a surface is associated to a partial power loss and an increased path length, translating into a higher path loss. Namely, from (2.11), the path gain for a ray reflected on r surfaces is

$$\boldsymbol{PG}_{\text{dB}} = 20\log_{10}\left(\frac{\lambda_c}{4\pi\sum_{i=1}^{r}\ell_i}\right) - \sum_{i=1}^{r}\boldsymbol{RL}_{i,\text{dB}}, \tag{2.14}$$

where ℓ_i is the length of the segment associated with the i-th reflection. The summation is decomposed into two terms to underline the different contributions: both the path length and the reflection losses degrade the path gain when the reflection order increases. Therefore, it is reasonable to assume that MPCs that bounce across multiple scattering surfaces have a low contribution to the overall received power, and can be omitted from the RT computations. Setting the maximum reflection order to $R' < R$, the RT complexity is decreased to $\mathcal{O}\left(tN^2R'T^{R'+1}\right) < \mathcal{O}\left(tN^2RT^{R+1}\right)$ with significant savings in terms of computation time, given the super-exponential dependency of the complexity on R.

MPC Thresholding Besides the reflection order, there are other elements that contribute to reducing the MPC path gain. For example, even if R is small, in large scenarios surfaces that are located far from the TX and RX nodes are associated to MPCs with longer path lengths (i.e., the first term in (2.14)). As the path gain from these scatterers is much smaller than that from close-by reflecting surfaces, it is possible to prune them from the list of MPCs to compute. At the same time, it is possible that some MPCs can be pruned regardless of the distance between the signal source and the detector, if the path gain falls below a minimum threshold as a result of an excessive reflection loss (i.e., the second term in (2.14)), e.g., when the signal propagates in NLoS.

For these MPCs, the path gain plays a key role and can thus be used as an indicator to perform the selection of the most significant rays. Specifically, the selection is performed at each time step considering a threshold γ_{th} in dB units, according to the following rules:

- the strongest ray is identified, regardless of whether it is obstructed, and the corresponding path gain $PG_{\text{strong, dB}}$ is computed;
- all the other MPCs are identified. Note that $PG_{i,\text{dB}} < PG_{\text{strong, dB}}$ for every MPC i.
- the selection is carried out discarding every MPC i such that $PG_{i,\text{dB}} - PG_{\text{strong, dB}} < \gamma_{\text{th}}$.

While being more general than the previous approach, as it identifies directly the weakest, least significant rays, this strategy requires the computation of all the geometric paths in order

Tab. 2.2: Characteristics of the simulated scenarios. Other important simulation parameters are the bandwidth $B = 400$ MHz, and the noise figure NF $= 9$ dB. All nodes of a given scenario transmit with the same power P_{tx} and carrier frequency f_c.

	Indoor1	**L-Room**	**Parking Lot**
Time steps t	3 133	3 831	13 579
LoS	✓	✓	✓
NLoS	✗	✓	✓
Environment	Indoor	Indoor	Outdoor
RX speed	1.2 m/s	1.2 m/s	(1.2, 4.17) m/s
Interferer	✗	✓	✓
T	12	16	755
P_{TX}	20 dBm	20 dBm	30 dBm
f_c	60 GHz	60 GHz	28 GHz

to obtain the path gain list. However, this method, which removes $(M - M')$ rays, eases the load on the subsequent RT operations, i.e., the obstruction check, reducing by a factor of $\frac{M-M'}{M}$ the complexity of each time step. In fact, following the same logic as in Sec. 2.2.3 for every ray of reflection order r, after the r geometrical operations required to compute the path of the ray, none of the $(r+1)T$ obstruction checks are performed if the ray is discarded. Updating (2.4) with $r \geq 0$ instead of $r + (r+1)T$ operations per ray, the complexity can be reduced to $\mathcal{O}\big(tN^2RT^R\big)$ and thus by a factor up to T. Note that, whereas this approach achieves a constant factor improvement, T can be in the order of tens to thousands, depending on the details included in the CAD file and on the adopted triangulation, thus dominating the complexity expression.

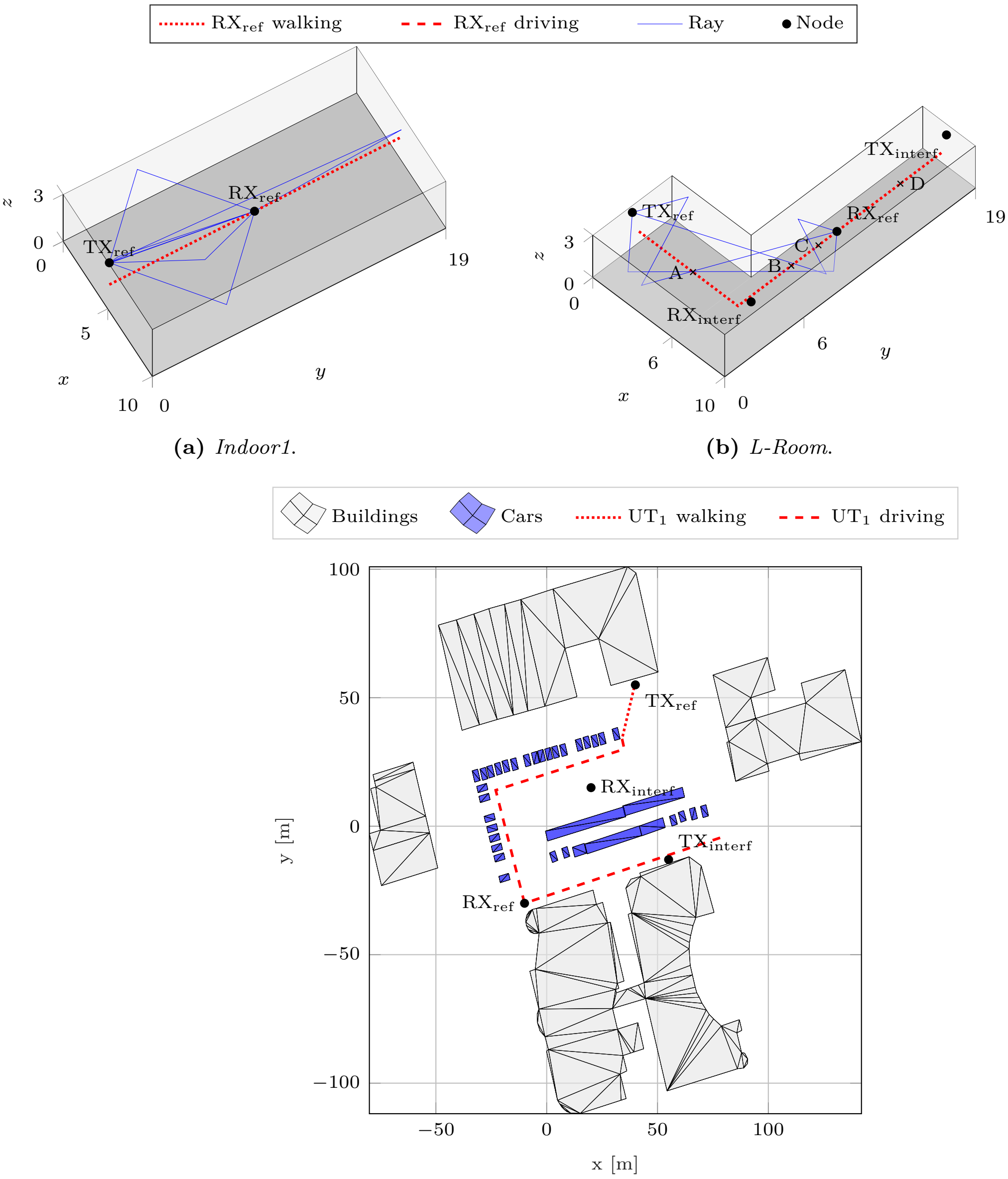
Absolute thresholding can also be used to limit the number of extremely weak rays similarly to the previous technique. This approach can be useful when considering high values for R, low values for γ_{th}, and especially when using the QD model. In this case, setting a conservative threshold Γ_{th}, every MPC i such that $PG_i < \Gamma_{\text{th}}$ is discarded.

The complexity of the RT can be significantly reduced thanks to the removal of MPCs and to the reduction of the maximum reflection order R. On the other hand, these simplifications degrade the accuracy of the simulation results at the different levels of the network stack. In the remainder of this thesis, we will quantify this trade-off for three realistic propagation scenarios. The overall end-to-end network performance and the runtime of the simplified RT settings will be compared with those of the complete, non-simplified channel traces.

2.4.1 Simulation Scenarios

This section reports the details of an extensive performance evaluation aimed at understanding the impact that the simplifications introduced in Sec. 2.4 have at different layers of the protocol stack. We first describe the scenarios and tools used for the performance evaluation (Sec. 2.4.1), then the link and higher layer performance (Secs. 2.4.2 and 2.4.3, respectively), and conclude with the computational performance given by the simplifications (Sec. 2.4.4) and guidelines for the most efficient design configurations (Sec. 2.4.5).

Three representative scenarios with distinctive features have been selected to make the performance evaluation as general as possible. Their main characteristics are hereby described and summarized in Table 2.2. Without loss of generality, only downlink channels are considered.



(a) *Indoor1.*

(b) *L-Room.*

(c) *Parking Lot.*

Fig. 2.13: Visual representations of our simulation scenarios. Distances are measured in meters.

1. *Indoor1*: The most basic scenario, with a rectangular room (see Fig. 2.13a) of size 10 m × 19 m × 3 m. The TX is positioned close to the ceiling at $(5, 0.1, 2.9)$ m. The RX, at height 1.5 m, moves away from the TX at a speed of 1.2 m/s along a straight line. This scenario was deliberately designed to be simple, to analyze the propagation characteristics simulated by the RT focusing on the received power pattern when different simplifications are used;

2. *L-Room*: An L-shaped hallway (see Fig. 2.13b). A static TX, placed at $(0.2, 3, 2.5)$ m, transmits to the reference RX that moves away from it at a speed of 1.2 m/s across the corridor. The shape of the room is such that the RX is in NLoS condition for a significant portion of the path. In order to analyze the impact of interference on the network performance, a second TX, placed at $(8, 18.8, 2.5)$ m and acting as interferer, communicates with an RX at $(9, 3, 1.5)$ m. Furthermore, the shape of the room plays an important role when comparing the proposed simplification techniques, as it may create blind spots where little or no signal is received;

3. *Parking Lot*: The only outdoor scenario, representing a parking area of about 120 m×70 m enclosed by buildings (see Fig. 2.13c). The reference TX, located at $(40, 55, 3)$ m, transmits from an access point placed at 3 m height on a building to the RX, which moves initially at a speed of 1.2 m/s, and then starts driving at a speed of 4.17 m/s to exit the parking lot. Moreover, an interfering TX at $(55, -13, 3)$ m communicates with its RX placed at the center of the parking lot at $(20, 15, 1.5)$ m. By far the largest scenario, it makes it possible to analyze the effect of the simplifications in terms of time savings when the CAD file contains a large number T of triangles, and parked vehicles allow for an investigation of both LoS and NLoS conditions. Moreover, the reference RX moves at a much higher speed than in the previous scenarios, as in a basic vehicular scenario.

The statistical quantities used by the QD model mentioned in Sec. 2.2.4, namely the reflection loss $\mathbf{RL}$, the rician factor $\mathbf{K}$, the inter-arrival times $\boldsymbol{\tau}$, the power-delay decay constant $\boldsymbol{\gamma}$, and the power-delay decay standard deviation $\mathbf{S}$, for the *Indoor1* [35] and *Parking Lot* scenarios, are obtained from detailed measurement campaigns.[*]

For each scenario, the RT and QD model softwares described in Sec. 2.2.3 have been used to generate the channel instances at 60 or 28 GHz for the specified devices and mobility patterns sampled every 5 ms. These traces were then integrated in a custom MATLAB simulator [32, 87] to evaluate metrics at the link layer (e.g., the SINR), and with the mmW module [19] of the ns-3 network simulator [88] to investigate the performance of the full protocol stack.

Ns-3 configuration For ns-3, we extended the channel model implementation described in [43] to account for a generic channel matrix computed, in this case, as expressed in (2.1)[†]. In the performance evaluation, this channel model has been combined with the 3GPP-like

[*] Material libraries for the environments presented in this thesis (and more) can be found in [57]

[†] The implementation can be found at `https://github.com/signetlabdei/qd-channel`.

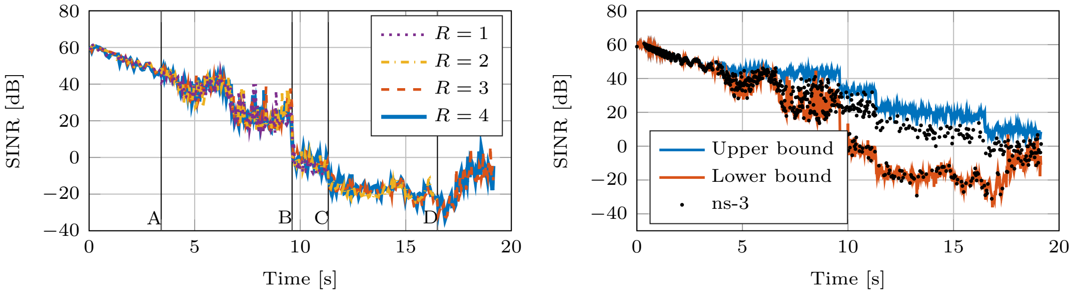
protocol stack of the 5G mmW module for ns-3 [19], which features physical and MAC layers with an OFDM-based frame structure, dynamic Time Division Duplexing (TDD), Adaptive Modulation and Coding (AMC), and several scheduler implementations. The channel model implementation influences the protocol stack performance through an error model that maps the SINR to the capacity of the physical layer. Besides, the UEs and base stations protocol stacks are completed by 3GPP Radio Link Control (RLC) and Packet Data Convergence Protocol (PDCP) layers, together with a realistic control plane based on the Radio Resource Control (RRC) layer which supports mobility-related procedures [89]. We consider two configurations for the uniform planar antenna arrays: large arrays, comprising 8×8 elements for the TXs and 4×4 elements for the RXs, and small arrays, comprising 2×2 elements for both TXs and RXs. All the arrays feature omni-directional elements spaced by $\lambda/2$. The planes on which all planar arrays lie are parallel to the y-z plane with a fixed orientation throughout the simulation. The beamforming is based on the SVD of the channel matrix $\boldsymbol{H}_{t,r}$, i.e., the beamforming vector $\boldsymbol{w}_{t,r}$ is the eigenvector associated to the largest eigenvalue of $\boldsymbol{H}_{t,r}$ [90]. Finally, thanks to the integration with ns-3, it is possible to equip the UEs with the TCP/IP stack and applications which connect to remote servers in the Internet.

The results shown in the following sections are benchmarked, unless stated otherwise, with the most complete and accurate baseline possible, where very conservative simplifications are introduced which produce negligible differences with respect to the measurements [35], allowing us to consider these baselines as equivalent to the actual measurements. Specifically, we set a maximum reflection order $R = 3$ for the *Parking Lot* scenario, and $R = 4$ for the others (to consider several reflections of the MPCs), a relative threshold $\gamma_{th} = -\infty$ (thus implying that no MPCs are discarded, regardless of their power level at the detector), a conservative absolute threshold $\Gamma_{\text{th}} = -200$ dB, a large antenna array configuration, only deterministic rays (i.e., no QD model), and a User Datagram Protocol (UDP) stream with an offered traffic equal to 800 Mbps.

2.4.2 Link-Level Performance Results

The first step towards proper protocol design is gaining a deep understanding of how the proposed ray-tracing simplifications impact the link-level performance of the network, neglecting, at this stage, the effects at the upper layers. In this perspective, we are interested in investigating how the strategies described in Sec. 2.4 result in different SINR regimes.

SINR evolution From Fig. 2.14a, which plots the temporal evolution of the SINR experienced when the RX moves in the *L-Room* scenario along the path described in Fig. 2.13b, we see that the impact of R is certainly non-negligible: the trend of the SINR visibly changes when progressively reducing the number of reflections per MPC. Moreover, we see that the SINR evolves consistently with the mobility of the RX. The SINR indeed drops by more than 30 dB when the RX loses the LoS (position B in Fig. 2.13b), while the SINR degradation experienced at time $t = 3.4$ s (position A in Fig. 2.13b) is due to the interference from $\text{TX}_{\text{interf}}$. Rapid



(a) SINR vs. time for different values of R, with $\gamma_{\text{th}} = -\infty$ dB.

(b) SINR vs. time. Solid lines represent upper and lower bounds, while dots show the SINR extracted from a single ns-3 run.

Fig. 2.14: Evolution of the SINR experienced when the test RX moves in the *L-Room* scenario along the path described in Fig. 2.13b.

fluctuations within the SINR trace are then due to the fact that different MPCs travel different paths. At 60 GHz, where the wavelength is as short as $\lambda = 5$ mm, even small variations of the path length between the direct ray and the reflected ones from the back wall (behind the TX), side walls, ceiling, and floor of the room, may result in strong fading. Fig. 2.14a also shows that the impact of R is particularly evident when the RX operates in NLoS: in this region, in fact, the received power drops to zero when first-order ($R = 1$) and second-order ($R = 2$) reflections are removed (positions C and D in Fig. 2.13b, respectively).

Impact of interference For completeness, in Fig. 2.14b we compare the metrics from the MATLAB (solid lines) and ns-3 (dots) simulations. The lower bound (red line) assumes an always-on interferer, while the upper bound (blue line) assumes an interference-free channel. On the other hand, ns-3 models a realistic transmission pattern for the primary and interferer links, which could occupy the channel in overlapping, partially overlapping, or non-overlapping time intervals. Therefore, for each time interval, the SINR generated by the ns-3 simulations is lower and upper bounded by the two other curves, with the ns-3 SINR much closer to the upper bound when the two transmitters use non-overlapping slots for communications with their receivers. We observe that the SINR, after steadily decreasing until time 17 s, starts increasing again. This happens because the $\text{TX}_{\text{interf}}$ is pointing its beam towards its own receiver $\text{RX}_{\text{interf}}$, and as RX_{ref} approaches $\text{TX}_{\text{interf}}$ it becomes less and less aligned with the interfering beam, which makes the interfering power actually decrease as a result.

SINR distribution Fig. 2.15 further investigates the impact of parameter R on the SINR, reporting its CDF for the three scenarios described in Sec. 2.4.1 as a function of R. In the *L-Room* scenario, the shape of the CDF of the SINR changes significantly when varying the value of R. This has eventually an impact on the capacity and packet error rate simulated at the physical layer. Specifically, reducing the number of MPCs of the channel may imply that the rays are not able to reach the RX position with sufficiently high power, thus resulting in

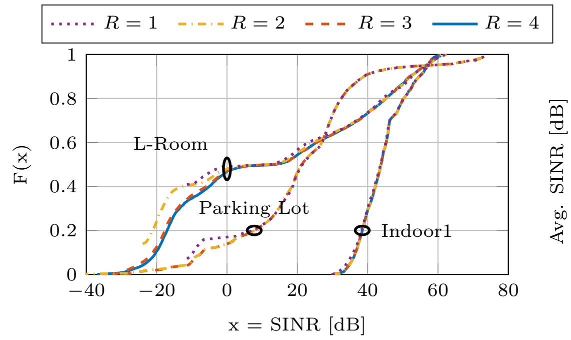


Fig. 2.15: Cumulative Distribution Function of the SINR in different scenarios vs. R, with $\gamma_{\text{th}} = -\infty$ dB.

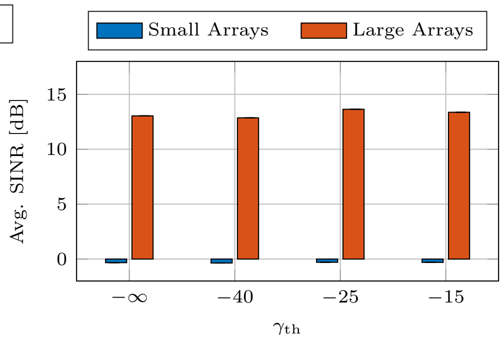
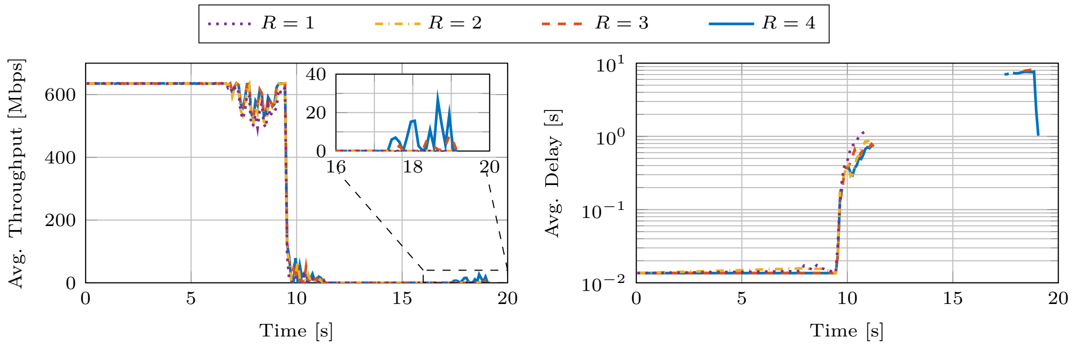


Fig. 2.16: Average SINR vs. γ_{th} as a function of the antenna architecture in the *L-Room* scenario, with $R = 4$.



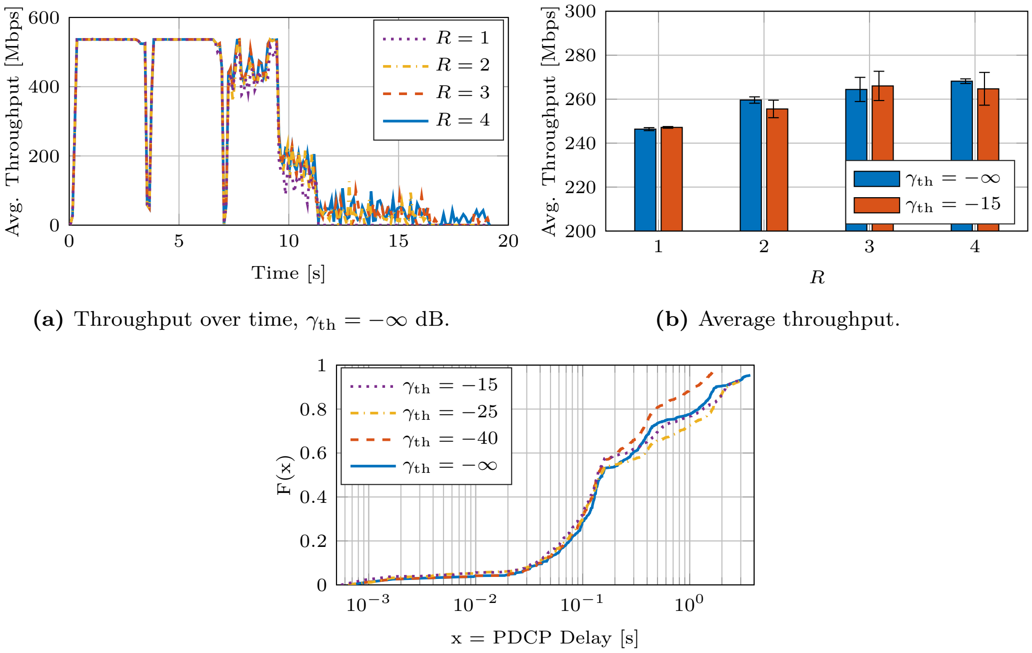
(a) Average throughput.

(b) Average delay.

Fig. 2.17: End-to-end performance vs. R for the *L-Room* scenario with an offered UDP CBR traffic of 800 Mbps.

a complete outage: for example, the CDF terminates at SINR $\simeq -10$ dB for $R = 1$ and at SINR $\simeq -23$ dB for $R = 2$. On the other hand, both the *Indoor1* and the *Parking Lot* scenarios are able to preserve the LoS for the whole duration of the simulation, thereby making it possible for the signal to propagate with a minor impact on the received power even when limiting the number of reflections R per MPC. Notice that the 20 dB gap of SINR between the *Parking Lot* and *Indoor1* configurations is due to the larger distance between the TX and the RX, and to the reflecting surfaces (e.g., buildings) in the outdoor scenario.

Impact of array size on the SINR Finally, in Fig. 2.16 we plot the SINR vs. the relative threshold γ_{th} as a function of the antenna size at the TX and the RX. As expected, the SINR increases when increasing the number of antenna elements, which increases the beamforming gain. Also, while the impact of R severely affects the SINR in the *L-Room* scenario, increasing the relative threshold γ_{th} to reduce the number of MPCs to be processed by the RT results in negligible deterioration at the link level, while speeding up the simulation, as will be discussed in Sec. 2.4.4.



(a) Throughput over time, $\gamma_{\text{th}} = -\infty$ dB.

(b) Average throughput.

(c) PDCP delay, $R = 4$.

Fig. 2.18: End-to-end performance vs. R and γ_{th} for the *L-Room* scenario with full-buffer TCP traffic.

2.4.3 End-to-End Performance Results

Many of the conclusions we derived from the link-level performance in Sec. 2.4.2 can be extended to the end-to-end metrics, i.e., throughput and delay at the PDCP layer. Statistics have been collected at this layer since they can easily profile both UDP and Transmission Control Protocol (TCP) traffic and are very close to the application layer performance, without the addition of extra delays due to the specific architecture of the simulation scenario. In this section we study three types of traffic, namely full-buffer TCP traffic and UDP traffic with a Constant Bitrate (CBR) of 100 Mbps and 800 Mbps. The latter was chosen to be the default for the results shown in this section, unless stated differently. Both throughput and delay are averaged over 100 ms windows.

Throughput and delay evolution with UDP traffic Fig. 2.17 reports end-to-end metrics over time for the *L-Room* scenario as a function of the maximum number of reflections R. First of all, we notice that the 800 Mbps data rate for UDP was chosen to saturate the channel capacity, as the physical layer only supports 630 Mbps of peak rate. The interference starts to impact the UDP performance at 7 s, followed by a rapid performance degradation when the direct ray is lost in point B (see Fig. 2.13), at about 9.65 s. Between point B and point C (11.33 s), the signal is still strong enough to allow for some transmissions, resulting, however, in a rapid increase of the delay of received packets due to buffering and retransmissions.

When $\mathrm{RX}_{\mathrm{ref}}$ gets closer to $\mathrm{TX}_{\mathrm{interf}}$ (i.e., around 17 s), as already observed (see the discussion about Fig. 2.14) the misalignment of the interfering signal results in a slight increase of the SINR, which allows to transmit the enqueued packets producing some non-zero throughput and a reduction of the delay (even though the transmission conditions remain quite severe and the performance is rather poor). We also observe that this effect is visible only for $R \geq 3$, since more simplified models have too few reflections, thus failing to reach the final part of the corridor.

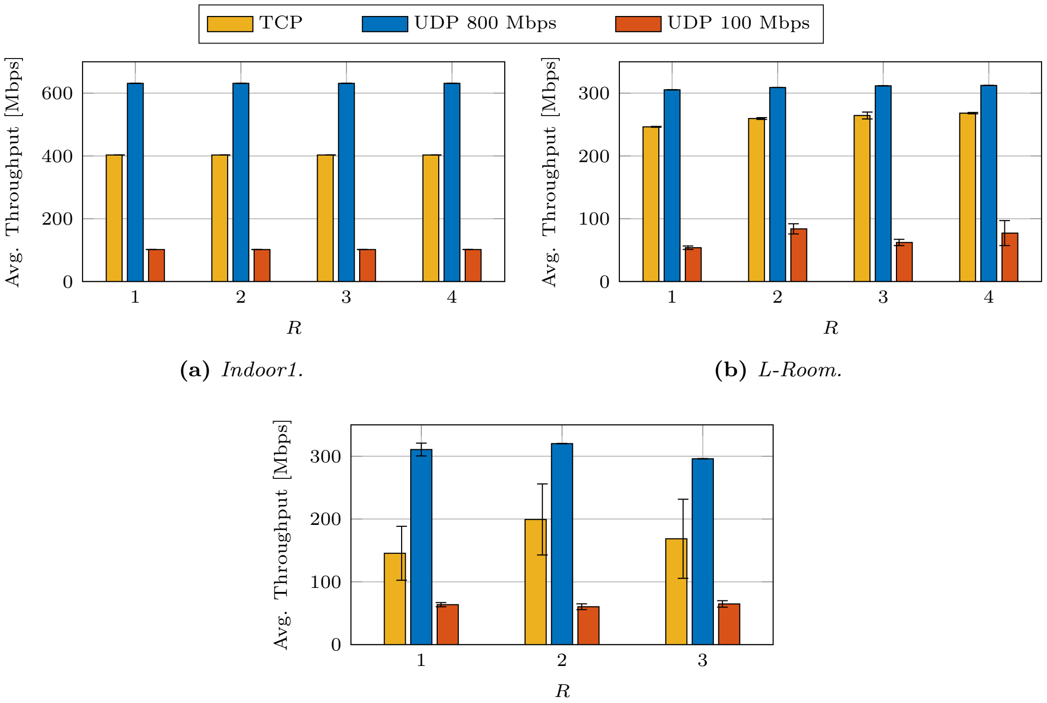
Throughput and delay analysis with TCP traffic Fig. 2.18a shows the corresponding simulations using a full-buffer TCP traffic stream, which reaches a peak rate of 536 Mbps. As expected, sudden jumps in the channel quality lead to sudden performance drops in TCP. This is the case in point A at 3.415 s (see Fig. 2.13), where the strong first-order rays of $\mathrm{TX}_{\mathrm{interf}}$ are received by $\mathrm{RX}_{\mathrm{ref}}$, at the beginning of the interfering regime at about 7 s, when the direct ray is lost in point B at 9.65 s, and finally when the strong first-order reflections are lost in point C at 11.33 s.

For these reasons, Fig. 2.18b highlights a slightly positive correlation between the average throughput and the maximum reflection order R, although only a 9% increase is observed with $\gamma_{\mathrm{th}} = -\infty$ dB, going from 246 Mbps for $R = 1$ to 268 Mbps for $R = 4$. In general, instead, delay statistics do not show a clear trend in the reflection order, nor in the relative threshold, probably due to the extra complexity created by retransmissions and queues. An example is shown in Fig. 2.18c where most statistics follow a very similar trend, separating only towards extreme values of delay, corresponding to the portion of the scenario after point B, i.e., when the direct ray is lost. Notice that the CDFs for the delay do not reach 1 since windows where no packets were received were considered to have infinite average delay.

Unlike for the UDP case at 800 Mbps, TCP decreases the congestion window when strong interference affects the communication for both the reference and the interfering streams. For this reason, packets sent during the interfering regime do not always collide with each other. The reduced interference greatly increases the perceived SINR, as explained in Sec. 2.4.2 for Fig. 2.14b, thus triggering transmissions even after point B for $R \geq 2$. Although not shown here, similar conclusions can be drawn for UDP traffic at 100 Mbps, which is able to transmit after point B as well, sending data at a rate that depends on the small scale fading affecting the communication.

Throughput analysis for different scenarios and antenna array configurations The average throughput for different configurations is shown in Fig. 2.19, including the 95% Confidence Interval (CI). The *Indoor1* scenario shows virtually no variations across different values of R for all the three types of traffic considered. Minor variations can only be observed for the *Parking Lot* and *L-Room* scenarios. For the latter, the NLoS regime sets apart simulations with $R \leq 2$ from those with $R \geq 3$, not being able to exploit the last part of the path with lower interference and thus showing slightly lower performance.

Similar results are shown in Fig. 2.20, where two sets of antenna configurations are considered. When smaller antenna arrays, and thus smaller antenna gains, are simulated, the average per-

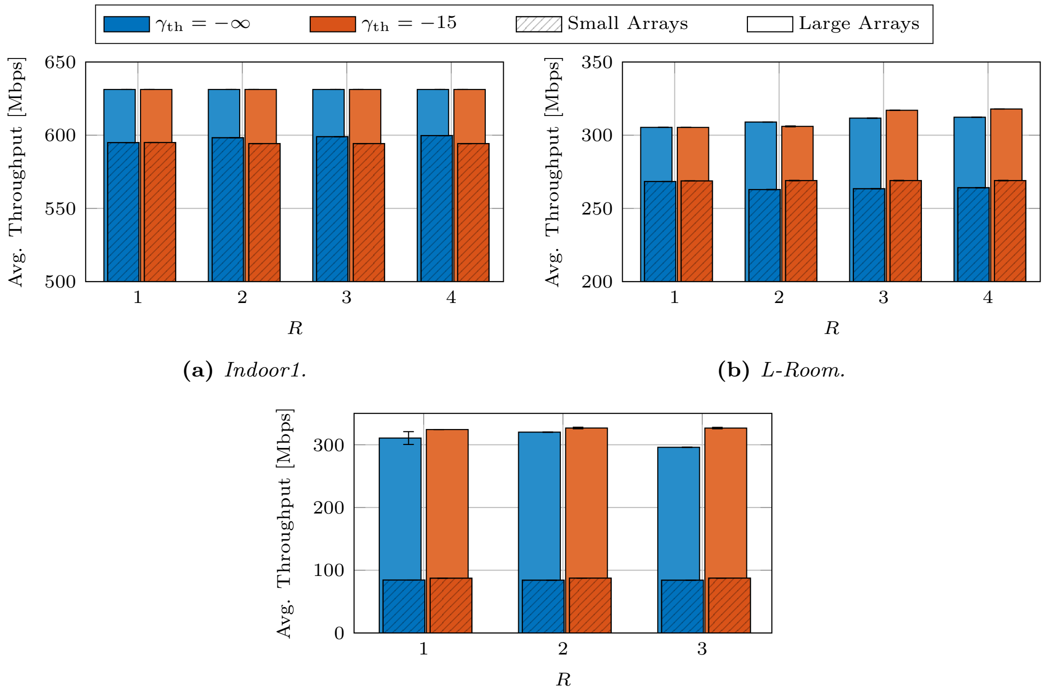


(a) *Indoor1.*

(b) *L-Room.*

(c) *Parking Lot.*

Fig. 2.19: Average throughput considering $\gamma_{\text{th}} = -\infty$ dB.



(a) *Indoor1.*

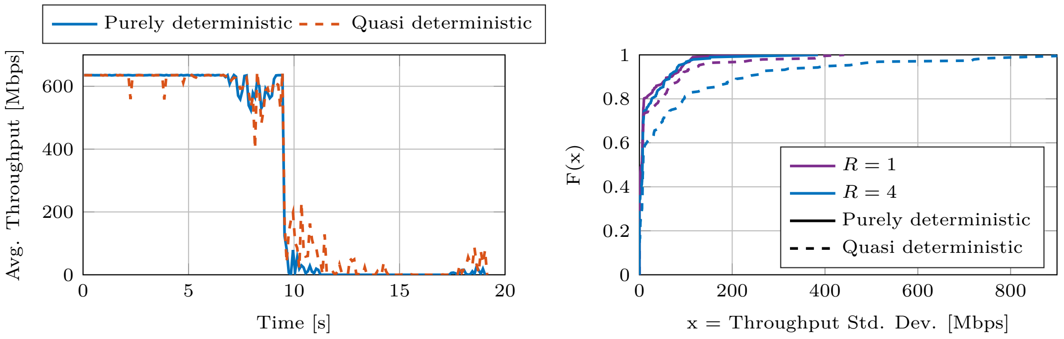
(b) *L-Room.*

(c) *Parking Lot.*

Fig. 2.20: Average throughput for the 800 Mbps UDP traffic considering different antenna architectures.

formance of the system decreases for all scenarios. The largest performance hit is experienced by the *Parking Lot* scenario, since the stronger path loss experienced as a result of the larger propagation distances involved in the outdoor scenario can be mitigated by the antenna gain. The performance drop observed in Fig. 2.20 can be also due to stronger interference. Smaller arrays, in fact, are not able to create narrow beams, making $\text{TX}_{\text{interf}}$ interfere more strongly with RX_{ref}. Moreover, we see that, even though a larger beam pattern would capture more rays than it would be possible in the case of narrower beams (that typically only absorb the strongest rays), both small and large antenna arrays show a minor impact of the reflection order R on the system performance, e.g., for the L-Room case, the throughput changes as little as 4% when R is reduced from 4 (our baseline) to 1. In fact the reflected rays do not contribute much to the overall received power budget, and can be pruned with minimal performance degradation. Small throughput variations are also experienced when increasing γ_{th} from $-\infty$ (our baseline) to -15 dB, except when $R \geq 2$ in the *L-Room* and *Parking Lot* scenarios, i.e., when reflected rays may still carry significant power.

Throughput analysis with and without QD model Finally, a comparison between a purely-deterministic and a quasi-deterministic channel with the QD model described in Sec. 2.2.4 is shown in Fig. 2.21. Results are given for $R = 4$ (to consider as many detectable reflections of the MPCs as possible) and $\gamma_{th} = -\infty$ (thus implying that no MPCs are discarded,



(a) Throughput over time with $R = 4$ and $\gamma_{\text{th}} = -\infty$ dB.

(b) Throughput standard deviation for $\gamma_{\text{th}} = -\infty$ dB.

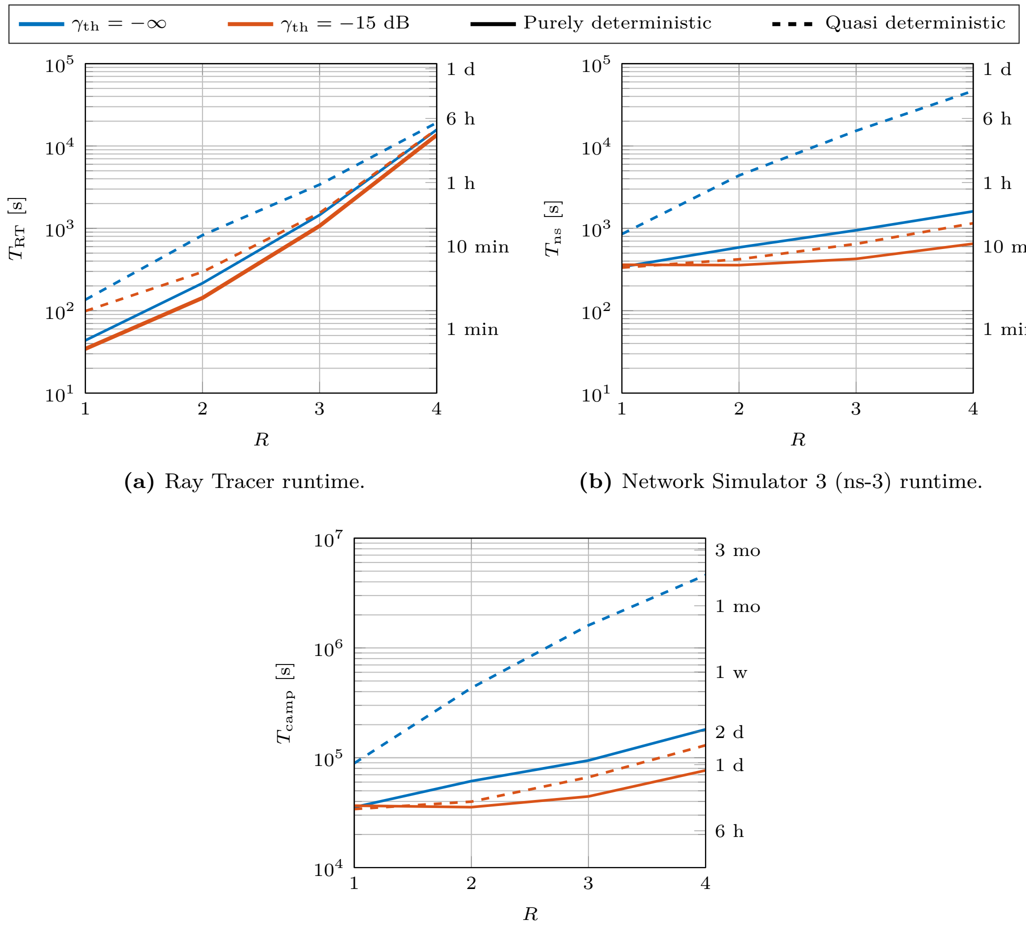
Fig. 2.21: Throughput performance for the *L-Room* scenario with a purely deterministic and quasi deterministic channel model, with a UDP CBR traffic of 800 Mbps.

regardless of their power level at the detector), to analyze the most complete and accurate baseline possible, where very conservative simplifications are introduced in the ray tracing computation. In general, from Fig. 2.21a it is possible to notice that the added random rays from the QD model tend to (i) increase the average received power and (ii) increase the frequency and amplitude of power fluctuations due to small scale fading, which is considered independent across subsequent time steps of 5 ms at a speed of 1.2 m/s. These fluctuations can also affect the end-to-end performance, making it significantly less stable.

To further study these random fluctuations, the CDF of the standard deviation of the throughput over 100 ms windows has been computed. To do so, we first computed the average throughput over 5 ms sub-windows, i.e., the sampling period chosen for the ray-traced channel, and subsequently the standard deviation over 20 consecutive sub-windows. This approach captures the deviation of the throughput over short time intervals, where it can be considered roughly constant. Computing the standard deviation over the whole simulation, in fact, would yield a misleading metric, given the extreme differences over the almost 20 s long scenario. Fig. 2.21b shows how an increasing number of rays tends to increase the standard deviation of the throughput due to an increased small scale fading, especially when a QD model with random diffuse components is considered: the gap is as large as 10% when transitioning from $R = 1$ to $R = 4$. This effect should be taken into account when evaluating the performance of protocols for mmW communications which adapt to the channel conditions, e.g., TCP [7].

2.4.4 Computational Performance

The simulation techniques proposed in Sec. 2.4 offer a trade-off between the simulation speedup given by the lower complexity, and a corresponding loss of accuracy. Secs. 2.4.2 and 2.4.3 analyzed in depth the impact of the simplifications on the network metrics at two distinct levels. Here, we compare the proposed simplifications from a computational complexity point of view, and then draw guidelines on the optimal combination of parameters that maximizes



(a) Ray Tracer runtime.

(b) Network Simulator 3 (ns-3) runtime.

(c) Total simulation campaign runtime.

Fig. 2.22: Simulation runtime vs. R and γ_{th} for the *L-Room* scenario. The total simulation campaign runtime (c) accounts for an RT simulation (a) and 100 sequential ns-3 simulator runs (b). A purely-deterministic channel and a quasi-deterministic channel are considered.

the accuracy.

Simulation time For completeness, we need to distinguish the different contributions to the total simulation campaign runtime T_{camp} required by a campaign of network simulations. The first is the RT runtime T_{RT}, required by the RT to generate the MPCs for the channel matrix (Fig. 2.22a). The second contribution, T_{ns}, is due to the network simulator (either MATLAB or ns-3 in this work), which includes the computation of the channel matrix with the RT data and what can be considered as simulation overhead (Fig. 2.22b). Usually, a simulation campaign aiming at proving a new result requires running simulations multiple times with different random seeds (i.e., Monte Carlo analysis), for a total number of simulations in the order of thousands. Running such simulations over tens of parallel processes is equivalent to running only hundreds of simulations sequentially, thus requiring a total of about $T_{\text{camp}} = T_{\text{RT}} + 100\, T_{\text{ns}}$ to first obtain the RT traces, and then re-use them for the whole simulation campaign (Fig. 2.22c).

Fig. 2.22 shows the RT, the ns-3, and the corresponding total simulation campaign runtime, for the *L-Room* scenario. The figure compares the impact on the computational complexity of the simplification introduced by the reduction of the maximum order of reflection R. We consider the two extreme values of γ_{th}, i.e., -15 dB and $-\infty$ dB, and compare a purely-deterministic channel with a quasi-deterministic channel that includes the random diffuse components introduced by the QD model. First, it can be observed that R has the greatest impact on the runtime: the RT runtime T_{RT} increases by more than 2 orders of magnitude when increasing R, and the ns-3 runtime T_{ns} experiences a similar effect. The impact of the QD model is clearly visible in Fig. 2.22b and Fig. 2.22c. Nevertheless, in this case increasing γ_{th} to -15 dB can effectively reduce the gap between the runtime with and without QD model, for every reflection order. However, whether and how this combination of parameters, while speeding up the ray tracer, affects the accuracy of the simulations will be discussed in the following paragraphs.

Accuracy metric To summarize the conclusions from Secs. 2.4.2 and 2.4.3 quantitatively, we compare the network performance of the simplified models and the baseline considering the Normalized Root Mean Square Error (nRMSE), computed as [87]

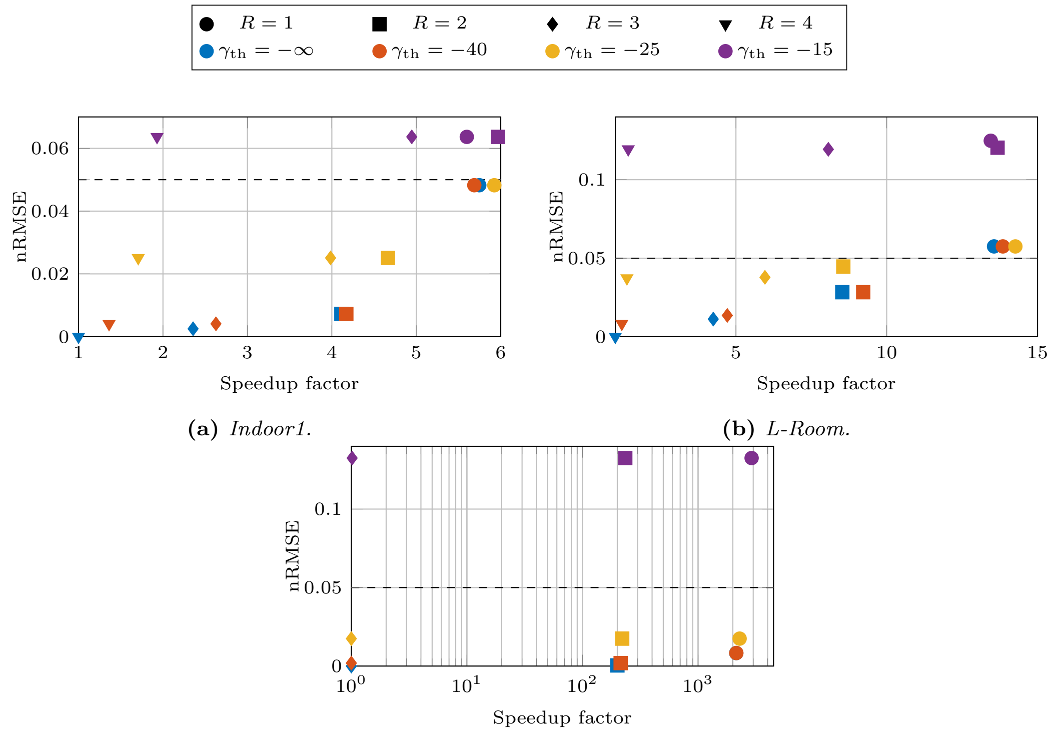
$$\text{nRMSE} = \frac{\text{RMSE}}{\sigma_{\hat{x}}} = \frac{\sqrt{\frac{1}{N}\sum_{n=1}^{N}\left(x_n - \hat{x}_n\right)^2}}{\sigma_{\hat{x}}}, \tag{2.15}$$

where x is the metric with the configuration of interest, $\hat{x}$ is the baseline metric, and $\sigma_{\hat{x}}$ represents the standard deviation of the baseline metric. This metric evaluates the distance between each baseline-simplified pair of a given simulated metric in the time domain. As in [87], we compare it with a speedup metric, defined as the factor by which the overall simulation T_{camp} runtime is reduced compared to the baseline. For the remainder of this section, we will consider 0.05 as the maximum acceptable value for the nRMSE, meaning that we deem acceptable an RMSE equal to 5% of the standard deviation of the considered metric.

Link-Level Performance The variations of the SINR due to the simplifications are shown in Fig. 2.23. In general we notice that markers with the same shape (same R) tend to increase with increasing steepness as the relative threshold increases, thus showing diminishing returns for the largest value of γ_{th}.

For the *Indoor1* scenario (Fig. 2.23a), significant deviation from the baseline occurs with $R = 1$, and with $\gamma_{\text{th}} = -15$ dB. Nevertheless, for all the considered cases, the nRMSE is smaller than 0.07, confirming what was anticipated in Sec. 2.4.2, i.e., that only minor changes take place even with the most aggressive simplifications. Within the maximum accepted nRMSE it is possible to accelerate the simulator up to a factor of almost 6 with an nRMSE equal to 0.048 choosing $R = 1$, $\gamma_{\text{th}} = -25$ dB.

Good performance is also obtained in the *L-Room* scenario (Fig. 2.23b). In this case, choosing $\gamma_{\text{th}} < -15$ dB and $R > 1$ makes it possible for the nRMSE to remain below 0.05, but an overall

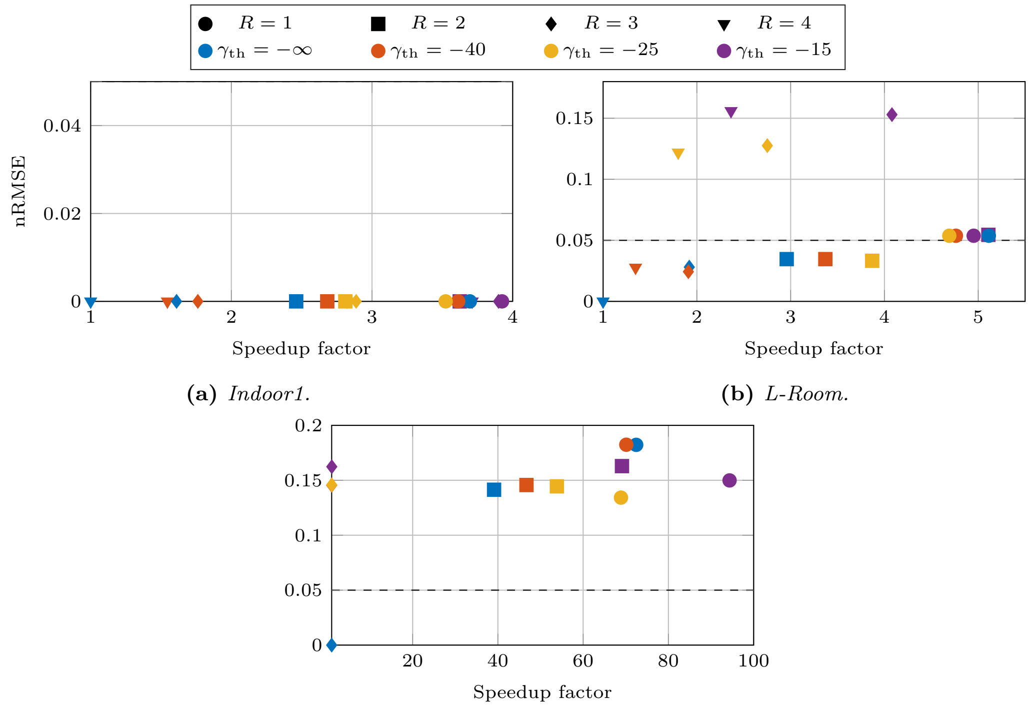


(a) *Indoor1.*

(b) *L-Room.*

(c) *Parking Lot.*

Fig. 2.23: Trade-off between the SINR performance and the speedup obtained with the different simplification parameters for the three scenarios. The dashed black line at nRMSE=0.05 represents the maximum acceptable value for the nRMSE. As in Sec. 2.4.2, results and runtimes for the link-level MATLAB simulator have been considered.



(a) *Indoor1.*

(b) *L-Room.*

(c) *Parking Lot.*

Fig. 2.24: Trade-off between the throughput performance and the speedup obtained with the different simplification parameters for the three scenarios. As in Sec. 2.4.3, for the throughput ns-3 has been considered.

speedup of a factor of 9.2 is obtained with $R = 2$ and $\gamma_{\text{th}} = -40$ dB. In this case, using $R = 1$ even with $\gamma_{\text{th}} = -25$ dB might still be acceptable, with an nRMSE of only 0.057 but a speedup factor equal to 14.3.

Finally, good performance is also obtained in the *Parking Lot* scenario (Fig. 2.23c), with a behavior extremely similar to the *L-Room* scenario but with significantly higher speedup factors due to the higher complexity of this scenario. In fact, for $\gamma_{\text{th}} < -15$ dB the nRMSE stays below the 0.05 threshold while the campaign runs orders of magnitude faster. Specifically, choosing $R = 1$ with $\gamma_{\text{th}} = -25$ dB yields an nRMSE of 0.017 with a speedup factor of more than 2000.

End-to-End Performance Fig. 2.24 reports the nRMSE of the throughput vs. the speedup for end-to-end simulations. Similarly to what happened for the link-level performance, γ_{th} shows diminishing returns especially for the *L-Room* and *Parking Lot* scenarios.

Fig. 2.24a shows that virtually no variation occurs when introducing simplifications in the *Indoor1* scenario. This suggests that setting $R = 1$ and $\gamma_{\text{th}} = -15$ dB can speed up the simulation by a factor of almost 4 with negligible accuracy loss with respect to the baseline configuration.

Good results are obtained also for the *L-Room* scenario in Fig. 2.24b, and with $R = 2$ and

$\gamma_{\text{th}} = -25$ dB it is possible to gain a 3.9 speedup factor with an nRMSE of 0.033, or even a 5.1 speedup if an nRMSE of 0.054 is still accepted using $R = 1$.

Conversely, the end-to-end simulations for the *Parking Lot* scenario are much more severely affected by the simplifications introduced in this thesis. In fact, Fig. 2.24c shows that the proposed simplifications are not able to achieve a significant speed-up without sacrificing the fidelity of the results, unlike in the previous scenarios.

2.4.5 Design Guidelines

Without focusing on the exact numbers, that are specific to the scenarios and the simulation tools employed in this work, it is still possible to draw some general guidelines for an efficient use of RT simulations. In particular, we do not only focus on link-level metrics, as generally considered in literature studies, but also identify which combination(s) of simplifications affect the system-level performance (which typically drives network design choices).

Scenario. The simulation scenario plays a key role in the simplification choice. Specifically, when considering indoor LoS simulations, the secondary rays can be neglected with good approximation and very significant time savings. When considering also NLoS conditions in indoor scenarios, instead, less flexibility should be expected, although it is still possible to considerably reduce the runtime with a minor accuracy loss. Finally, outdoor scenarios should be treated carefully, as aggressive simplifications may have detrimental effects on the fidelity of the simulations. Nevertheless, a working point can usually be identified which offers a significant speedup.

Simplification Strategy. Although their effect can vary substantially depending on the considered scenario, some general considerations can be drawn regarding the simplification techniques. The link-level metrics, such as the SINR, benefit, in terms of runtime, more from a reduction of the maximum reflection order than from an increase of the MPC threshold. Conversely, full-stack metrics, such as end-to-end throughput and delay, are more transparent to ray tracer's simplifications. In any case, an aggressive thresholding policy leads to a performance degradation that is not justified by a corresponding speedup improvement. Finally, end-to-end results require a balanced use of both simplification techniques to achieve an optimal working point. In particular, for the *L-Room* scenario we showed that considering more than two reflections, i.e., $R = 3$ or $R = 4$, would not decrease the nRMSE significantly, while in turn reducing the speedup gain by a factor of 2. At the same time, for the *Parking Lot* scenario, it may be fundamental to decrease γ_{th} below -25 dB to maintain a good trade-off between accuracy and speedup, even though an nRMSE below the 0.05 threshold could be guaranteed only setting $\gamma_{\text{th}} = -\infty$. Conversely, for the *Indoor1* scenario even the most aggressive simplifications would not virtually affect the RT accuracy. In light of this, setting $R = 2$ and a relative threshold $\gamma_{\text{th}} \in [-40, -25]$ dB consistently yields a good balance of accuracy and speedup in almost all cases, and for this reason it is the suggested configuration. Moreover, end-to-end evaluations of protocols that adapt to the channel behavior should consider combining the ray tracing process with a QD model.

2.5 Blockage Modeling

One of the main outputs of this work revolves around the ability to introduce and study blockage in ray-traced scenarios. To do so, we started designing and building a software tool able to perform this exact task which we call *Blockage Manager*.

The software should support different geometries and mobility patterns for specific blockers, allowing both deterministic blockage scenarios and randomized campaigns, e.g., for future large-scale studies. Clearly, multiple arbitrary blockers should be supported at once, together with their effects on the wireless channel, meaning that blockers might also introduce diffraction, refraction, diffusion, and even further reflections.

The availability and our knowledge of the NIST-UniPD open source `qd-realization` software [91] suggests that traces from this software should natively be supported by the *Blockage Manager* software. Furthermore, to simplify the user interaction with the software, minimal amount of code should be required to set up a simulation.

The software assumes that channel traces have already been obtained, and thus post-processes them based on the desired blockage environment. For the time being, the software is a work in progress and the code is thus currently undisclosed. We will make sure to publish a public version of it as soon as we have a stable and usable version, together with a related publication.

2.5.1 Framework Overview

From our work on `qd-realization`, we learned some important lessons. Specifically:

1. MATLAB is expensive and thus not meant for open-source projects;
2. A solid testing framework is key to robustly extend and update a complex and potentially large software project;
3. Modularity and a consistent code quality is key for maintenance and upgradeability.

Regarding Item 1, we decided to build the *Blockage Manager* software from scratch based on Python 3, one of the most common programming languages for open-source projects. While it may not represent the best performing option, it is extremely convenient, simple, and flexible, and solutions exist to run scientific python code faster. In general, we prioritized flexibility and maintainability, given the research-oriented objectives of this software.

Regarding Item 2, we understood that MATLAB lacks a robust and flexible native testing framework, and it is especially hard to continuously test a MATLAB-based software. Instead, there exist several widely-used testing frameworks for Python, among which we chose `pytest`. Being easy to use, flexible, and widely supported, we found it to work well for our purposes. We built a unit testing framework, currently consisting of over 100 tests covering 100% of the code base. Furthermore, we set up a Continuous Integration and Developement (CI/CD) pipeline based on *CircleCI*[*], which allows us to fully test our code base every time we upload updates to *GitHub*.

Finally, Item 3 was addressed by creating solid and flexible modules using the object-oriented paradigm.

The different modules of the *Blockage Manager* software are described in the following paragraphs.

Geometry After looking for existing Python geometry modules, we found that none existed with the desired features. Specifically, since the software is based on ray-traced channels, it needs to handle ray geometry, and thus needs at least the notions of `Point`, `Vector`, `Line`, and `Segment`. These objects need to interact, while distances, projections, and collisions need to be computed against other common objects (e.g., canonical 3D shapes, triangles, etc.). All of this, of course, has to be implemented in a user-friendly manner to minimize code writing and maximize clarity. Optimization will then be beneficial to improve the computational performance.

Ray This module organizes the information provided by a ray-tracer, such as delay, path gain, phase, and the actual path (i.e., the vertices of the ray-traced path). It also offers a simple interface to consistently compute AoDs and AoAs. Notice that, for the correct import and usage of the *Blockage Manager* software, extttqd-realization should also export visualization files, namely, `MpcCoordinates`.

Scenario We defined a `Scenario` interface, which subclasses have to implement. This was done in an attempt to generalize the definition of a scenario, in principle allowing for the possibility to support different ray-tracing formats in the future. The interface defines common methods to import/export traces in the target format, as well as to access and update sets of rays between nodes. The `Scenario` interface was extended into `QdRealizationScenario`, which is specifically able to handle channel traces for multiple users and timesteps.

Notice that `QdRealizationScenario` supports the version of extttqd-realization used in [92], not the current master branch. Specifically, JSON outputs are currently not supported, but could be easily included in the future.

Obstacles We also defined a common `Obstacle` interface to handle obstructions, diffraction, and other effects that a generic obstacle will impose over `Rays` from the imported `Scenario`.

We already implemented a simple obstacle, namely `SphereObstacle`, which incorporates a fixed transmission loss affecting the path gain (while the phase is left unaffected).

Mobility Models A user can move an `Obstacle` during a simulation by using *MobilityModel*s. Each obstacle will then update its position based on such model, thereby providing an accurate and temporally-correlated mobility and making the channel temporally

[*] https://circleci.com/

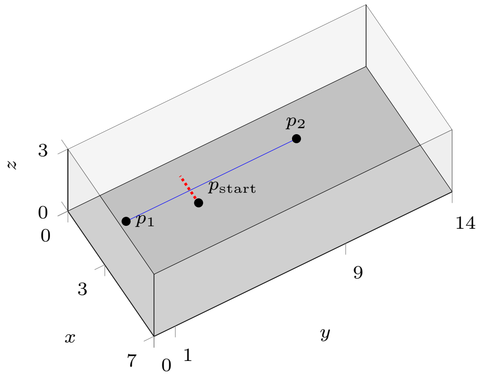


Fig. 2.25: Visual representation of the blockage evaluation scenario, inspired by [84]. The blue line represents the direct ray between the two nodes, the dotted red line represents the path taken by the obstacle.

consistent. The `MobilityModel` interface was extended into a number of common models, such as the `ConstantPositionMobilityModel`, the `ConstantVelocityMobilityModel`, the `ConstantAccelerationMobilityModel`, the `WaypointMobilityModel` (the `Obstacle` will move linearly between a list of given `Points` at the given speed and will optionally remain stationary at the arrival point for a given pause duration), and the `RandomWaypointMobilityModel` (similar to `WaypointMobilityModel`, but waypoints, velocities, and pauses are randomly generated with distributions given by the user).

Environment The `Environment` class handles:

1. A `Scenario` object;
2. A list of `Obstacles`.

It represents the core of the *Blockage Manager* software, where `Obstacles` interact with pre-computed `Rays`.

2.5.2 Example

We tested our framework in a scenario inspired by [84].

Specifically, we created an empty 14×7×3 m room with static nodes in positions $p_1 = (1, 3, 1.6)$ and $p_2 = (9, 3, 1.6)$, as shown in Fig. 2.25. We simulated the channel `qd-realization` [57] considering reflection orders 0 (only direct ray), 1 (up to first-order reflections), and 2 (up to second-order reflections).

We then imported the channel trace into the *Blockage Manager* software, configuring a `SphereObstacle` moving from $p_{\text{start}} = (4, 3.7, 1.6)$ downwards at 0.3 m/s, and sampled the channel every 3.4 ms for 1500 samples, for a total of about 5 s of simulation time. Given the simplicity of the proposed simulation, we considered a sphere with a diameter of 30 cm at the same height of both the transmitter and the receiver, in an attempt to emulate the size of the

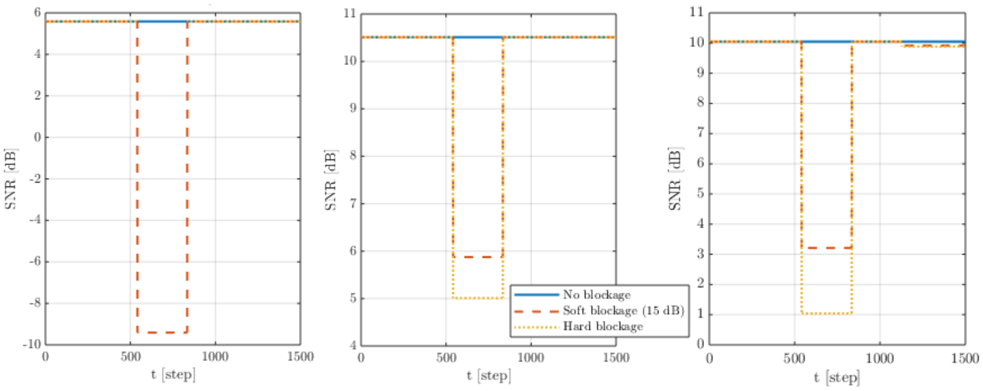


Fig. 2.26: Results of our first simple simulation campaigns.

human body. We also set the obstacle's transmission loss to 15 dB (soft blockage) and $+\infty$ dB (hard blockage) to show the flexibility of the described framework and compare the results.

The channel traces are processed with a simple link-layer simulator considering a single omnidirectional antenna for each node, with a noise figure of 9 dB at the receiver, and a transmitted power of 20 dBm at 60 GHz over a bandwidth of 2.16 GHz.

Simulation results are shown in Fig. 2.26. From left to right, we consider a baseline scenario with reflection order equal to 0, 1, and 2. For each scenario, we show the temporal evolution of the received power considering no blockage, a soft blocker, and a hard blocker.

When no blockage is considered (blue line), the received power, and thus the Signal-to-Noise Ratio (SNR), is constant throughout the whole simulation time. Since a different number of rays is considered in each scenario, the baseline received power differs in the three proposed scenarios.

For the leftmost scenario in Fig. 2.26, when only the direct ray is present, a hard blocker disrupts the connection for about 300 time steps, i.e., the time required for the sphere, i.e., the blocker, to pass through the direct ray between the two nodes. In fact, the yellow dotted line representing the SNR in case of a hard blocker disappears just after $t = 500$, since $\text{SNR}_{\text{dB}} = -\infty$. On the other hand, a soft blocker introduces a 15 dB attenuation on obstructed rays, resulting in just a 15 dB decrease of the SNR.

Considering also first reflections from the side walls, the floor, and the ceiling, the effect of the blockage event is not as strong as in the previous cases, as shown in the central scenario in Fig. 2.26. In fact, the hard blocker does remove the direct ray, but the reflected paths are still able to reach the user, resulting in an overall attenuation of only 5.5 dB. Similarly, the soft blocker increases the path loss of the direct ray by 15 dB, but combined with the reflected rays results in an overall 4.5 dB attenuation.

Finally, the rightmost scenario in Fig. 2.26 includes also second order reflections, which were found in [92] to be sufficient to accurately model most mmWave scenarios. The effects are similar to the previous scenario in which only first order reflections were considered. The additional reflections, though, introduce an additional source of channel variability, decreasing the received power by 9 and 7 dB, respectively for the hard and the soft blockers. Furthermore, toward the end of the simulation, one of the second order reflections is blocked, resulting in a lower power loss.

Notice the unnatural sharp transitions between LoS and NLoS. Our current simplified model does not take into account smooth transitions, as it applies the transmission loss of a given obstacle whenever it is obstructing a ray.

2.6 Conclusions

Performance evaluation is a fundamental part of the design of 5G mmW networks. To that end, an accurate channel model allows researchers to generate reliable simulation results, that can qualitatively and quantitatively describe what can be expected when using real devices. In this chapter, we analyzed how widely used channel models can be improved to better describe the reality and optimized for better performance by simplifying them. The usage of large antenna arrays, however, requires an accurate representation of the spatial dimension of the channel, for example through RTs, which can model the propagation of the different multipath components of a mmW signal based on the geometry of the scenario.

Specifically, with respect to more detailed channel models, in Sec. 2.2.4 we introduced a mathematical formulation for a class of mmW channels, i.e., the QD models, that can closely simulate the propagation of rays in a specific environment calibrated on real-world channel measurements. We provided a step-by-step tutorial on how such models can be implemented, including the parameters and random distributions obtained from a NIST measurement campaign [35]. We then compared the results that can be obtained with an open source implementation of the model with the real measurement traces, showing significant improvements with respect to a fully deterministic channel model in mimicking reality. We want to highlight the fact that since our original publication on this topic [62], more measurement campaigns have been performed, thus releasing more material libraries for a set of new scenarios openly available [91].

We then proceeded to the analysis of a stochastic SCM: the 3GPP TR 38.901. The thorough profiling partially reported in this thesis highlights how such a complex channel may require a very significant computational overhead, thus possible simplifications were suggested and studied to understand their effects on common simulation scenarios. Unfortunately, the simplification was constrained by the structure of the 3GPP model itself, as discussed in Sec. 2.3.3 and [32], although the proposed approach showed a significant performance improvement with little to no behavioral changes.

Transferring this knowledge to (quasi-) deterministic channel models, the impact of different simplifications on both the physical layer and the end-to-end network performance has been numerically evaluated for different scenarios, applications, and antenna array configurations. Notably, after introducing the RT method based on the MoI, and the parameters influencing its computational complexity, we discussed two strategies which aim at avoiding computations for MPCs which do not contribute significantly to the overall received power. The first limits the maximum reflection order, while the second removes MPCs with a path gain which is much smaller than that of the strongest ray. We then showed that the optimal trade-off is achieved when neglecting third-order (and beyond) reflections, and multipath components with path gain lower than -40 dB compared to the strongest ray, further discussed in [87, 92]. We believe that the insights that resulted from the extensive profiling and performance evaluation can guide researchers in designing accurate, yet scalable, simulations of mmW networks.

Finally, we gave an overview of our *Blockage Manager* software, which will allow us and fellow researchers to better study the effect of blockage on mmW communications. Although the work is still in its early stages, we believe that it will be able to help us produce novel and interesting research.

Strong of the knowledge we obtained from these experiments, we propose some promising future research directions.

Regarding improvements to mmW channel models:

1. Better analyze peculiar, although relevant, channel states, e.g., human blockage, self-blockage, user terminal rotation. Not only some of them lack standard or even popular models, but their effects on end-to-end performance has not been fully analyzed.

2. Always analyze and contain the additional channel complexity introduced by the added features.

3. Study network performance of a scenario with and without obstacles, to fully assess their impact on communication and test whether the current state of the art is able to withstand these non-ideal conditions. Obstacle randomization, in terms of quantity, type, shape, and mobility will help us creating large campaigns with a richer and more realistic simulation environment.

Secondly, on improving the scalability of mmW simulations:

1. Better analyze the impact of simplifications on directionality. Most of the proposed methods rely on reducing the multipath components of the channel, thereby preventing the model from being used for the evaluation of techniques that, in reality, exploit channel sparsity to form multiple simultaneous beams in independent angular directions (e.g., hybrid beamforming);

2. Investigate optimization techniques that can automatically tune the simplification parameters to obtain the maximum speedup while minimizing accuracy degradation;

3. Compare end-to-end measurements in actual mmW deployments with full-stack simulations at different degrees of simplification.

3
Antenna Array Modeling and Optimization

3.1 Introduction

Massive UPAs operating in the mmW frequency range will be adopted in the 5th generation of mobile networks (5G) as the key enablers to meet the challenging requirements of the new standard. Large antenna arrays can compensate for the propagation and penetration losses at such high frequencies thanks to beamforming techniques, synthesizing 3D beams that can focus the transmitted power towards specific users [21], increasing the antenna gain and thus increasing the received power. Furthermore, beamforming can help exploit the unique propagation characteristics of the mmW channel, such as spatial sparsity, to reduce the interference among users and improve the channel coherence time [93].

On the other hand, directional communications make it inherently more difficult to handle mobile scenarios, making it necessary to keep the alignment between the transmitter and receiver beams [9], introducing more overhead. Directivity is even more critical when experiencing sudden blockage events. In this case, the pair of communicating devices should either store backup links or find a new connection on the fly, increasing the overhead and complexity of beam management operations.

The combination of these phenomena may seriously affect the quality experienced by mobile users. In this context, it is necessary to implement a detailed performance evaluation to identify and address the weakest points of the technology across the full communication stack. Cellular networks are extremely complex systems, highly correlated with hidden factors that may impact the overall performance at different scales. Usually, the different parts of the system, i.e., the antenna system, the RF components, the cellular protocol stack, etc., are designed using a block-level approach, developing each block independently of the others. However, this methodology may lead to undesired behaviors and even sub-optimal performance, since the possible side effects among the components may have a strong impact on the overall system. This problem can be solved using system-level simulation tools, which represent an accurate and cheap solution to evaluate the overall performance before the actual deployment and to adjust the design of each component accordingly. Indeed, simulation enables large scale evaluations, allowing the user to decide the degree of abstraction required by the desired analysis [20].

These ambitious goals require a novel approach to antenna design, optimization, and simulation. Antenna arrays can no longer be designed and optimized without considering the network topology: in addition to the common antenna design goals, such as decreasing the side-lobe level or maximizing the directivity, more global, network-oriented requirements need to be taken into account. Such requirements dramatically increase the complexity of the optimization problem, as it moves from the bare electromagnetic to the network domain. As both antenna prototyping and network deployment tests are prohibitively expensive for both academia and most industry, electromagnetic and network simulators are often employed. In [25], the accurate modeling of antennas in network simulators was proved to be decisive, further confirming that design and optimization need to carried out jointly.

Heuristic simulation-based optimization is generally not feasible, as such detailed simulators are both time and computationally expensive. Indeed, the large number of iterations needed by optimization algorithms prevents the use of simulations requiring hours (or even days) of running time.

The remainder of this chapter will be structured as follows: in Sec. 3.2 we briefly describe relevant works related to antenna optimization and modeling, followed by an introduction of antenna array modeling in Sec. 3.3.

In Sec. 3.4 we propose and evaluate an ML framework that can mimic a given simulator and allows us to approximate any network optimization objective in a reduced amount of time. Two antenna optimizations will be proposed in Secs. 3.5 and 3.6.

Then, in Sec. 3.7 we will showcase an open-source and publicly available* framework for full-stack 5G NR-compliant simulations, based on the popular open-source ns-3. While other 5G NR simulators already exist, to the best of our knowledge we propose a simulator that has unique features, such as (i) a ray-tracing based channel model for mobile users, improving the spatio-temporal coherence over the previous stochastic channel models [43], (ii) a flexible antenna module, comprising of multiple parametric antenna elements as well as a generic interface for phased antenna arrays, and (iii) a Beamforming (BF) module. This work builds upon a pre-existing full-stack 5G NR mmW simulator [19], a popular tool developed jointly by the University of Padova and NYU-Wireless. Thanks to the integration with ns-3, this simulator features a detailed implementation of the TCP/IP stack, together with several traffic and mobility models, allowing the community to analyze and compare full-stack behaviors of different physical and protocol setups.

Finally, we will draw our conclusions in Sec. 3.8.

*https://github.com/signetlabdei/ns3-mmwave-antenna

3.2 Antenna Optimization: Related Works

Recently, ML techniques have started to be applied as a tool to solve many kinds of problems. Also in the communication field, there exist many works adopting learning-oriented approaches to address complex transmission issues. Particular attention has been gathered by the new database proposed in [94], as it lays the premises for a common research ground.

One common application of ML is parameter estimation, where great results were achieved even where the most sophisticated classical techniques failed. An example can be found in [95], where the authors try to estimate the downlink channel starting from samples of the uplink channel. While well-known signal processing techniques (e.g., the Wiener filter) were not able to perform a good estimate, the ML approach proposed by the authors yielded very good results.

Another common approach is the encoding of the channel representation through autoencoders [96]. Autoencoders are an unsupervised learning algorithm, and as such they do not need labeled data but can learn autonomously. The idea behind this technique is to train two Neural Networks (NNs), one performing the encoding of the input data, the second trying to decode it. The layer between the two should contain, in our case, a useful and extremely compressed representation of the channel. This can be applied at many levels, starting from the pure, physical channel model, to the entire transmitter-channel-receiver chain [97]. This allows obtaining either encoders/decoders, transmitter/receiver chains or channel models that have a much lower computational complexity.

ML has been successfully applied also at the network layer. Innovative ideas and proposals have challenged even the most resilient classical paradigms such as the ISO/OSI architecture [98]. These new approaches started showing their potential in the increasingly heterogeneous network scenarios, e.g., when facing the high data load and quality of experience required for video streaming [99].

Moreover, the authors in [100] use Deep NNs to optimize the allocation algorithm in a wireless resource management problem. The proposed concept is similar to the one described in our work, as a learning tool is used to approximate a complex input-output function. However, the authors also include the optimization step into the learning process and use many more training samples to accommodate the needs of their deep architecture. For our work, instead, it is crucial to use as few samples as possible as we aim to speed up the optimization process by approximating very slow simulators, making the data acquisition the main bottleneck.

In the literature, many research activities have been focusing on the study of mmW mobile environments while in parallel a lot of works have studied in the past the problem of beamforming and antenna array optimization. However, there are no conclusive works focused on antenna optimization for mmW mobile scenarios.

At high frequencies, such as in the mmW bands, where strong attenuations are present, quantifying the actual antenna gain obtained due to the radiation pattern is fundamental to precisely evaluate any mmW system. In a previous work [25], realistic antenna radiation patterns have been studied and precisely characterized, motivated by the need to properly capture mmW propagation behaviors and understand the achievable performance in 5G cellular scenarios. As it is customary, antenna patterns were modeled as the superposition between the single element radiation pattern and the array factor. The latter term is a function used to characterize the effects of the entire array, while the former is used to quantify how the power of each antenna element is irradiated in all directions. The work shows how the single element radiation pattern affects the network performance, thus highlighting how optimization of this further parameter is not only possible but also required.

Finally, regarding the specific problem of thinned arrays, several works exist on their optimization at the antenna level. A reference for the general theory and results can be found in [101]. On the optimization side, [102] and [103] apply genetic algorithms to the activation mask of the array to further lower the side-lobe level. Nevertheless, as mentioned above, none of these works consider network metrics in the antenna design. Some works, such as [104], explicitly employ thinning for interference reduction, but they do not rely on simulated network statistics that allow achieving ad hoc optimizations, tailored to the network characteristics. This is due to the high complexity of the network simulators, which would be added to the already high computational load of the electromagnetic simulation. Here is where our framework can prove useful, providing an agile emulator that can be used to speed up complex optimization tasks, resulting in reasonable execution time.

3.3 Antenna and Beamforming Modeling

Antenna Array Model Phased antenna arrays can have extremely diverse geometries, from which their BF capabilities are derived. While it would be possible to create a generic class for arbitrary phased arrays, some geometries (e.g., uniform linear and planar arrays) are extremely popular and deserve specialized methods. For this reason, we created a generic interface for phased antenna arrays, specifying the polarized element field pattern, the locations of the elements (from which it is possible to compute the phase difference experienced by each antenna element for a transmitting or receiving signal), and the BF vector (the phase shifts and amplifications applied to every single element necessary to obtain the desired beam shape).

For this work, we considered the model described in the 3GPP specifications TR 38.901 [24]. The standard describes a uniform planar array, meaning that antenna elements are equal and are placed in an equally-spaced $M \times N$ rectangular lattice with vertical spacing d_V and horizontal spacing d_H, which form a panel. In our implementation we consider the simpler case of vertically polarized elements and only a single-panel configuration.

Antenna Element Model Phased antenna arrays are composed of multiple antenna elements capable of radiating and receiving electromagnetic signals. Every antenna element has a specific radiation and polarization pattern due to its specific design. Different antennas are needed in different contexts, e.g., directional elements can be used in multi-sector devices (e.g., gNBs), while quasi-isotropic antennas may be used for devices with no preferred communicating direction (e.g., UEs with a single-antenna panel).

A large number of antenna element designs exist in practice, leading us to creating a generic interface allowing users to add their own antenna models. In general, it is possible to create antenna elements with pattern measured from real devices to further increase the simulation accuracy. For this thesis, we implemented three of the most common antenna element models, with directivity pattern in dBi D_{dB} in the θ (inclination) and ϕ (azimuth) directions:

- *Isotropic antenna element*

  $$D_{\mathrm{dB}}(\theta, \phi) = 0$$

- *3GPP antenna element* [24]

  $$D_{\mathrm{v,dB}}(\theta) = -\min\left\{12\left(\frac{\theta - 90^\circ}{\theta_{3\mathrm{dB}}}\right)^2, SLA_V\right\}$$
  $$D_{\mathrm{h,dB}}(\phi) = -\min\left\{12\left(\frac{\phi}{\phi_{3\mathrm{dB}}}\right)^2, A_{\max}\right\}$$

  $$D_{\mathrm{dB}}(\theta, \phi) = G_{E,\max} - \min\left\{-(D_{\mathrm{v,dB}}(\theta) + D_{\mathrm{h,dB}}(\phi)), A_{\max}\right\}$$

  where the side-lobe attenuation in the vertical direction $SLA_V = 30$ dB, the maximum attenuation $A_{\max} = 30$ dB, the vertical and horizontal 3 dB beamwidths are respectively $\theta_{3\mathrm{dB}} = \phi_{3\mathrm{dB}} = 65^\circ$, and the maximum directional gain of the antenna element is $G_{E,\max} = 8$ dBi.

- *Cosine antenna element*

  $$D_{\mathrm{dB}}(\theta, \phi) = G_{\max} + 20\log_{10}\left(\cos^{\alpha_h}\left(\frac{\phi}{2}\right)\cos^{\alpha_v}\left(\frac{90^\circ - \theta}{2}\right)\right),$$

  where the exponents $\alpha_{h/v}$ can be computed from the beamwidths $\mathrm{BW}_{\mathrm{h/v}}$ as $\alpha_{h/v} = \frac{-3}{20\log_{10}\cos\frac{\mathrm{BW}_{\mathrm{h/v}}}{4}}$, and the maximum gain $G_{\max}$ can be computed from the directivity formula found in [105].

Beamforming Model Multiple BF architectures exist, which are commonly divided into three main categories, namely analog, digital, and hybrid. In analog architectures, a network of phase shifters is used to connect the antenna elements to a single RF chain, enabling a passive control of the beam by acting on the elements' phases. In digital architectures, instead, each antenna element is connected to an independent RF chain to provide digital control of the BF using baseband processing. The presence of multiple RF chains enables Multi-User Multiple Input, Multiple Output (MU-MIMO) operations, i.e., independent data streams can be transmitted and received simultaneously, possibly serving multiple users at the same time. Finally, hybrid architectures represent a middle ground between analog and digital approaches, in which the array is divided into multiple sections, each including multiple elements connected to an independent RF chain.

Although digital and hybrid architectures have the potential to achieve higher spectral efficiencies, several technological and economic issues still make analog BF a valuable choice, also due to its relatively low complexity. For this reason, in this work we consider the analog architecture and leave the study of the other two categories as future work.

Analog BF is achieved by controlling amplitude and phase shift of each antenna element of the phased array; this corresponds to assigning a complex number to each element, which is often identified as a *BF vector*. Several algorithms exist to compute such vectors, each affecting the directivity pattern in a unique way. Some try to maximize the gain in given directions, some try to suppress side lobes, some try to regulate the beamwidth, some others try to optimize the performance for a given channel estimate, and some others even try to also take into account the interference generated to other users.

In general, two main approaches exist: those based on a channel estimate, and those based on BF codebooks.

For the first approach, we implemented an algorithm originally proposed in [106] based on the Multiple Input, Multiple Output (MIMO) *Maximum Ratio Transmission* scheme, in which the optimal weight vectors correspond to the singular vectors associated with the largest singular values of the SVD of the estimated channel matrix. For a perfect channel estimate in interference-free environments, this method ensures optimal performance. Unfortunately, good channel estimates are hard and expensive to obtain, especially when dealing with large antenna arrays. The SVD decomposition is itself an expensive operation, and sending feedback information comprises a difficult trade-off between accuracy and overhead. For this work, we assume that the channel matrix is perfectly known, thus posing the BF algorithm in ideal conditions.

For the second approach, we implemented a generic interface for codebooks, allowing the user to create custom ones (for the sake of our evaluation, we used the tool available at [*]). We also implemented a file-based codebook, allowing to create complex codebooks using other custom and highly-specialized softwares, avoiding the computation of sophisticated algorithms in ns-3. As a first step, the implemented codebook-based BF computes the SINR for every pair of TX/RX BF vector, choosing the pair with the best performance. The advantages over the previous approach are many, in particular, no channel estimation nor complex matrix decomposition is performed and the only feedback needed is the index corresponding to the best performing BF codeword. On the other hand, exhaustive search among all possible codeword pairs may be inefficient, while reducing the search to a subset of codewords might yield suboptimal performance. We leave a more realistic and standard-compliant beam-management implementation and evaluation as future work.

3.4 Framework Description

The objective of the proposed framework is to speed up simulation-based optimization in the presence of slow simulators. Optimization based on simulated data requires several iterations,

[*] https://github.com/signetlabdei/codebook-file-generator

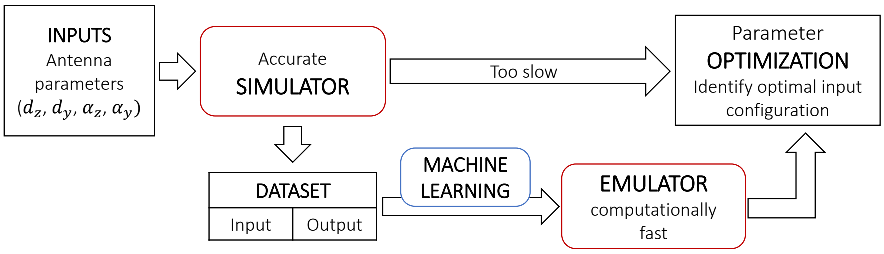


Fig. 3.1: Workflow of the proposed framework. The diagram highlights how the parameter optimization is achieved using an ML-based emulator.

each with a different input configuration, for the optimization strategy to steer toward the optimal value. The major constraint is the simulation time*, which makes a brute-force approach infeasible. The goal of our framework is to require a small number of simulations to learn the input-output relationship through ML algorithms, which are orders of magnitude faster to evaluate. A key advantage is that, after the preliminary database creation, the optimization of the selected antenna parameters can be achieved in a negligible amount of time, even when testing different optimization goals. In fact, we remark that once the emulator is trained, the optimization of multiple objective functions can be done instantaneously.

Although the idea is broadly applicable, our focus here is on antenna optimization over network-level metrics for mmW systems. We consider this use case as a testbed and we report the results of this particular optimization later in the chapter. The diagram in Fig. 3.1 shows how the parameter optimization can be achieved through the ML-based emulator, which only requires a single training phase done using a dataset of simulated data. More details on the different parts of the proposed workflow will be given in the subsequent sections.

In order to assess the validity of this framework, three main questions have to be answered:

Q.1 Is it possible to emulate a complex network simulator with a learning tool, and which learning tool can achieve the best emulation accuracy?

Q.2 How many train and test samples are needed for the chosen learning paradigm to converge and to be robustly evaluated?

Q.3 Does the achieved precision allow an optimization that is accurate enough to be useful?

The remainder of this section is devoted to addressing these problems.

3.4.1 Network Simulator

To test the framework, we need some data to learn from. A custom simulator was built in order to efficiently obtain results from such complex simulations. Simulation parameters are 3GPP standard compliant [24, 107], using the Urban Micro-cell (UMi) scenario with no O2I losses.

*Simulation times vary remarkably depending on the type of simulation and the accuracy required. It is not unlikely for a single run to require hours or even days.

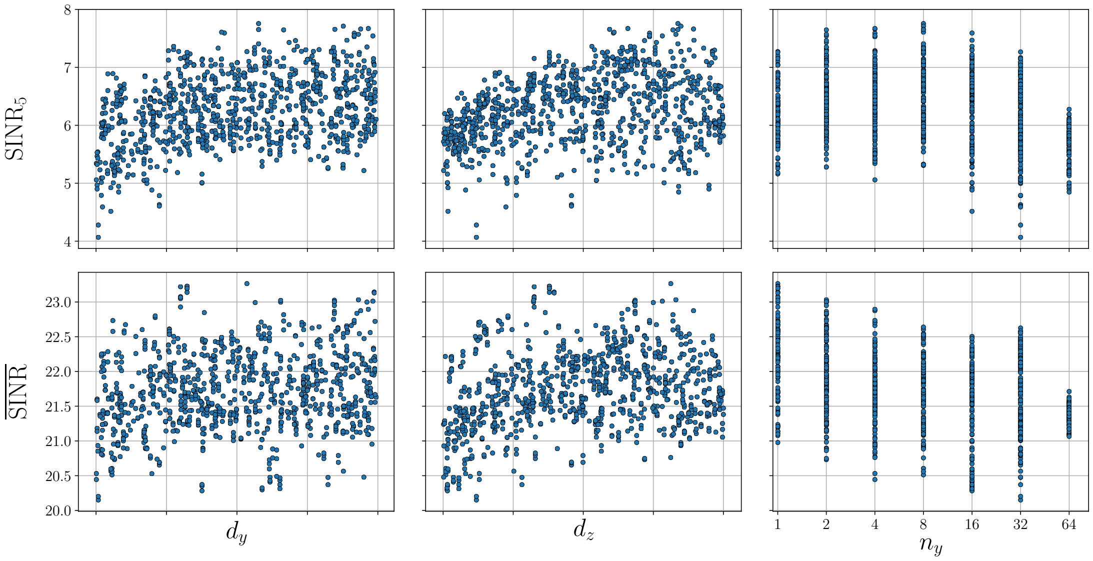


Fig. 3.2: Correlation between selected inputs and outputs.

3.4.2 Data Analysis and Machine Learning

The dataset was created with the simulator introduced in Sec. 3.4.1.

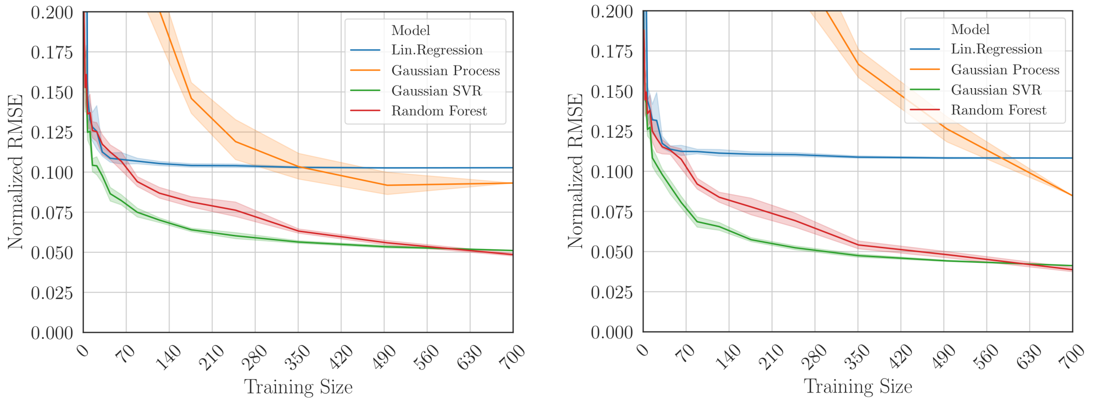
Given that our goal is to show the capabilities of the framework and not the optimization itself, the simulator has been simplified to obtain a good number of samples in a reasonable amount of time. It should be clear that such a rich database would correspondingly require more time when using a complex, thus more realistic simulation.

As usual in ML when dealing with new datasets, the initial phase is devoted to the analysis of the gathered data. A proper preprocessing, e.g., normalizing the inputs and removing the linearly correlated features, can boost the learning performance. The scatter plot showing the relation between the inputs and the outputs is reported in Fig. 3.2. Note that, even though the visual inspection of the data through different representations can help identify some hidden trends, its effectiveness is limited both by the high dimensionality of the problem and by the scarcity of available samples. Therefore, in general, it is not possible to rely on this kind of data analysis for optimization.

The objective of the learning algorithm is to learn the underlying function mapping the input antenna configuration to the output network metrics, for example

$$\begin{aligned} f : \mathbb{R}^n &\to \mathbb{R}^m \\ \mathbf{x} &\mapsto \mathbf{y} \end{aligned} \tag{3.1}$$

where $\mathbf{x}$ is the vector of the n input antenna parameters, f represents the simulator, computing the output network statistics from a given antenna configuration, and, finally, $\mathbf{y}$ is the vector of the m considered network metrics. Therefore, the learning algorithm (*emulator*) learns an



(a) SINR_5

(b) $\overline{\mathrm{SINR}}$

Fig. 3.3: Plots show the nRMSE as a function of the number of training samples. Multiple runs are performed, showing mean (line) and 95% confidence interval (shadowed area) for each algorithm.

approximation $\hat{f}$ of the *simulator*'s underlying function f, thus trying to mimic it.

Considering a scalar output y, the prediction or emulation error is then computed as the difference between the prediction of the emulator $\hat{y}$ and the corresponding simulator output y. In order to assess this, we define the nRMSE as

$$\mathrm{nRMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\frac{y_i - \hat{y}_i}{y_i}\right)^2}, \tag{3.2}$$

where N is the number of samples of the test set. This parameter allows for a fair comparison among metrics on different scales, as the normalization yields a percentage of standard error with respect to the simulated value. Note that SINR values are first converted to linear units.

An effective way to address questions Q.1 and Q.2 is reported in Fig. 3.3, related to the scenario described in Sec. 3.5, where the performance of the selected algorithms is evaluated for increasing training sizes. We recall that increasing the number of training samples is always beneficial for learning, improving both emulation accuracy and stability. However, it affects the dataset creation time, going against the purpose of the framework. From the comparison of Figs. 3.3a and 3.3b, it emerges that different emulation accuracies can be achieved for different metrics and that some learning algorithms predict a given metric better than others. This suggests that the choice of the technique should be made specifically for each metric. Eventually, this choice should also be made considering the number of available samples, as more complex algorithms, e.g., random forest, may outperform more basic ones when trained with enough samples.

Moreover, note that the performance of linear regression quickly saturate, while more complex algorithms achieve a lower error before converging. Saturation is expected even with the most powerful algorithms since data obtained from the simulator is inherently noisy (e.g., the number of Monte Carlo simulations is never infinite, thus statistics are not perfect). Instead, the reason

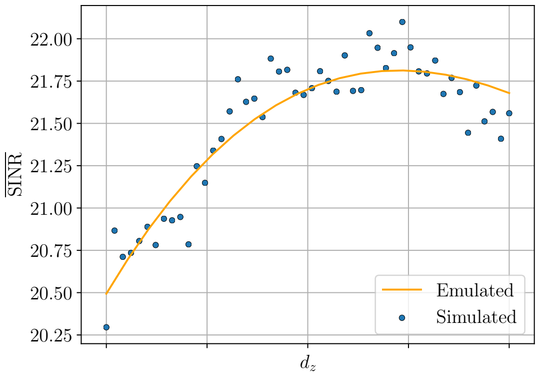


Fig. 3.4: Representation of a one-dimensional plot obtained by fixing all the array parameters except one. The plot makes it possible to visually compare the emulator fit with the simulator samples. In this case, an 8×8 array was used with $d_y = 0.5\lambda$ horizontal spacing, while the vertical spacing d_z is varying. The emulator is still trained in all 4 inputs simultaneously, justifying the suboptimal fit towards higher values of d_z.

why simpler algorithms tend to saturate earlier and with higher errors is because they are too simple to describe the inherent properties of the underlying function f. This concept can be easily seen in Fig. 3.4, where we visually compare the emulator fit with the simulator samples.

As expected, the nRMSE decreases as the training size increases, but at different rates for different algorithms. The trade-off between the number of samples and the emulation precision has to be taken into account when selecting the algorithm. The achieved nRMSE can be quite low, namely about 3.2% and 5.7% for $\overline{\mathrm{SINR}}$ and SINR_5, respectively. Finally, it can be observed that the two estimated metrics in Fig. 3.3 present different behaviors and performance for different outputs. Furthermore, in general, we observed that it is not possible to have a universally valid list of best algorithms, as this is very much dependent on the simulator, the scenario, and even the considered metric. As a basic approach, once the error achieves a target threshold, the emulator can be used for the optimization and the simulator can be stopped.

3.4.3 Optimization

The proposed framework is optimization agnostic, meaning that most standard numerical optimization techniques can be equally used. Clearly, the learned representation is just an approximation of the real-world performance: while the simulator tries to reproduce reality via random experiments, the emulator tries to approximate the input-output relationship of the simulator via a black-box approach, adding a level of abstraction that further distances it from the real world.

Since in general our models are not required to be differentiable, nor would we have an explicit derivative for most of them, gradient-based techniques are hardly usable. Some of the inputs could also be categorical or discrete (e.g., the number of antennas in each dimension). Furthermore, we are not posing any constraint on the convexity (or concavity) of the underlying function. For these reasons, gradient-based optimization algorithms would not even be desirable.

On the other hand, since a global optimum is typically desired, gradient-free global optimization algorithms exist that satisfy all these requirements (e.g., genetic algorithms or simulated annealing). Nowadays most scientific-oriented programming languages have optimization libraries, implementing several algorithms. As briefly explained in Sec. 3.4, Fig. 3.4 shows the noisiness of the training data. Thus, finding the maximum values over the raw data might not be the best choice, while numerically finding a global maximum over a smooth model might be a better choice, provided that the model is not underfitting. In the next section, we show the results obtained for the antenna optimization.

3.5 Regular UPA Optimization

In this section, we will discuss our results when trying to optimize the most common antenna array geometry, the regularly spaced rectangular UPA. We will first describe the simulation scenario in Sec. 3.5.1, and then discuss the related results in Sec. 3.5.2.

3.5.1 Scenario Description

A custom simulator was built in order to efficiently obtain results from such complex simulations. Simulation parameters are 3GPP standard compliant [24, 107], using the UMi scenario with no O2I losses.

The variable parameters of the scenarios are listed here:

- the antenna spacing d_z, d_y in the vertical and horizontal directions, respectively;
- the number of antenna elements $n_z, n_y \in \{1, 2, \dots, N\}$ in the vertical and horizontal directions, respectively. The total number of antenna elements is fixed to $N = 64$, in order to obtain a fair comparison between different configurations. Thus, n_z and n_y are the integer divisors of N and are deterministically related through $n_z = N/n_y$.

For each configuration, we collect network-level metrics such as the average SINR $\left(\overline{\mathrm{SINR}}\right)$ and the 5^{th} percentile of the SINR (SINR_5).

In this section, results are validated using a test set of 300 samples, while the training set is composed of 700 samples, that were proved to be large enough for this setup to obtain good testing accuracy (see Sec. 3.4.2). The test set size is kept fixed, to allow a simplified presentation of the framework while guaranteeing a proper result validation. Therefore, concerning question Q.2 from Sec. 3.4, only the training set size is taken into account.

Several learning techniques [108] have been analyzed and tested. However, only results for linear regression, Gaussian Process Regressions (GPRs), random forests and Support Vector Regressors (SVRs) are hereby reported.

- *Linear regression* is the most basic class of regression algorithms. Despite its simplicity, many versions and adaptations have been created, able to solve non-trivial problems. It is often considered as a baseline for more powerful algorithms.

- *GPRs* consider data as if it were sampled from a Gaussian stochastic process, trying to minimize the log-marginal-likelihood during the fit;
- *Random forests* are ensembles of decision trees, that approximate stepwise the target function;
- *SVRs* are derived from the Support Vector Machine (SVM) classification algorithm. Among all the typical kernels, the Gaussian one performed best and is used here.

One of the main advantages of linear regression is that, due to its simplicity, it is fast to train and easily interpretable, i.e., the analysis of the coefficients leads to some insights on the importance of the different inputs and their correlation. On the other hand, random forests and SVRs are black-box algorithms, meaning that results are hardly interpretable. Given their popularity, Neural Networks (NNs) have been tested as well. However, the lack of a large dataset has been found to be problematic for a stable convergence and they have thus been discarded from this study.

3.5.2 Results

The optimization phase shows the significant advantages of this framework. As previously stated, we remark that the proposed framework can be used for optimization in a wide set of scenarios, beyond that of cellular network design, used here as an example. As the optimization is done jointly on all the input parameters, the hyperspace where it operates can be extremely vast and complex. These features, along with the complexity of the search of the global maximum, require a very large number of evaluations. The gain of the framework can then be measured comparing the number of entries necessary for the database creation with the number of function evaluations needed by the optimization. This is because, due to the typical complexity of a simulator, the time required to obtain the database far exceeds that of the training and the optimization itself. In terms of time costs, the training itself is negligible and, once trained, the predictions are instantaneous.

Another aspect to take into account is that, although significant, the database creation in our framework is an overhead that is needed only once, as it does not depend on the optimization goal. The same emulator, providing almost instantaneous iterations, can be used with different optimization objectives, without requiring long simulation-based iterations.

For our example, given the data analysis initially done (partially shown in Fig. 3.2), we use as the objective function

$$\begin{aligned} &\text{maximize} \quad && \overline{\text{SINR}} \\ &\text{s.t.} && \text{SINR}_5 > 6 \text{ dB} \end{aligned} \tag{3.3}$$

where the constraint on the worst UEs (identified with SINR_5) has been introduced in order to guarantee some degree of fairness and coverage to all the UEs in the network.

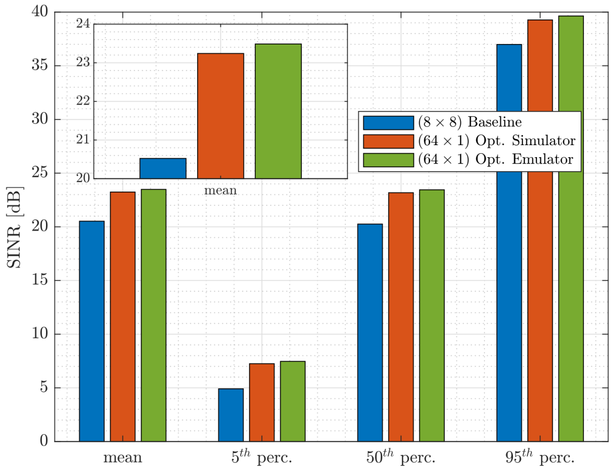


Fig. 3.5: Comparison among the network performance obtained with the baseline configuration (blue bar), with the optimal configuration identified using the simulator samples (orange bar) and using the emulator (green bar).

The optimization results obtained within the scenario described in the previous section are presented in Table 3.1 and Fig. 3.5. Results show that in the proposed scenario, a baseline setup consisting of 8×8 arrays with $\lambda/2$ spacing in both directions performs significantly worse than the optimized ones. The other two configurations represent the optimum obtained over the collected dataset (*Opt. Simulator*, made of 1000 randomly sampled points in the four-dimensional space described in Sec. 3.4) and the global optimum obtained using our framework (*Opt. Emulator*). They both identified a 64×1 configuration (vertical Uniform Linear Array (ULA)), but respectively with 0.825λ and 0.734λ spacing. Results show a ~ 3 dB improvement over the trivial baseline. Although in this case the results are really close (both inputs and outputs), two facts are important: firstly, we discussed in Sec. 3.4 that significantly fewer than 1000 samples would have been enough, a far lower number than required by a brute force optimization; secondly, as more inputs are considered, the input space will not be sampled enough to find a good setup, making emulation even more important.

Having computed 1000 samples while the optimization required more than 12000 function evaluations, we obtain a speedup factor of 12× with respect to brute force evaluation. A key advantage of our approach is the possibility of changing the objective functions of the optimizer, which would be easily and quickly done with the emulator, without having to retrain it.

Tab. 3.1: Numerical results shown in Fig. 3.5.

	$\overline{\text{SINR}}$	SINR_5	SINR_{50}	SINR_{95}
Baseline	20.52	4.91	20.26	36.99
Opt. Simulator	23.24	7.25	23.18	39.27
Opt. Emulator	23.49	7.47	23.45	39.64

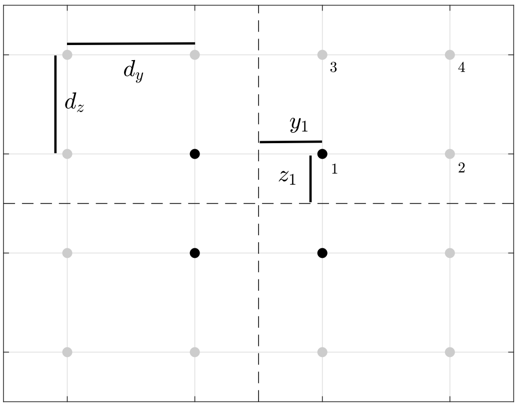


Fig. 3.6: Example of a generated array. Dashed lines separate the four quadrants, while black and gray dots represent respectively the activated antennas and the array lattice. The top-left quadrant is generated and then mirrored to the other three.

3.6 Thinned UPA Optimization

As in Sec. 3.5, a 3GPP compliant simulator was used to extract network-level metrics, such as SINR statistics, based on a Monte Carlo approach. Specific environment parameters follow the 3GPP standards [24, 107] based on the UMi scenario with no O2I losses.

The goal of this study is to understand whether irregular thinning is a desirable property in an array. In Sec. 3.6.1 we describe the adopted irregular thinning approach, while in Sec. 3.6.2 we list the parameters to be optimized. Finally, in Sec. 3.6.4 we will discuss the optimization results obtained from our framework.

3.6.1 Antenna Array Generation

To simplify both the implementation and the optimization, thinning is defined by means of an *activation mask* over a regular lattice of dummy antennas. Namely, a large antenna array lattice is created but only some of the antennas are turned on (see Fig. 3.6). Thus, all the antenna elements have approximately the same element pattern and thinned arrays are more easily parameterized.

The activation mask is randomly produced at each iteration of the Monte Carlo simulation as follows. First, the lattice is split into four quadrants. Then, starting from the center of the lattice, a *probability profile* $f(\Delta_y, \Delta_z) = f_y(\Delta_y) f_z(\Delta_z)$ is defined, where Δ_y and Δ_z are the distances of the antenna elements from the center of the lattice in the horizontal and vertical dimensions, respectively. Considering a single quadrant, each element $i = 1, \ldots, N_{quadrant}$ in position (y_i, z_i) is assigned a value $v_i = u_i \cdot f(y_i, z_i)$, where $\{u_i\}$ are i.i.d. uniform random variables defined in the interval $[0, 1]$. Finally, the elements with the largest values v_i are chosen and the sample quadrant is mirrored over the other three, to force a realistic symmetry. In this

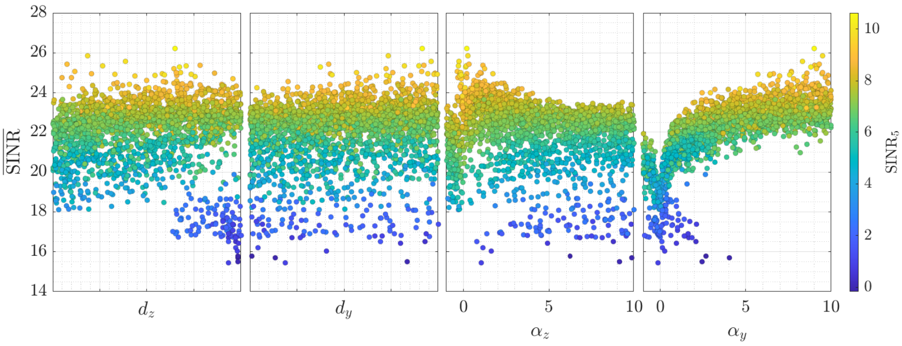


Fig. 3.7: Correlation plot of the four input parameters vs the output metric (y-axis) and the bound metric (color). From here it is already possible to see the importance of the antenna shape parameters for the system performance, giving a hint on the optimal parameters found by a good emulator.

thesis, a probability profile following an exponential decay $f_y(\Delta_y) = e^{-\alpha_y \Delta_y}$ (and analogously for f_z) is chosen.

3.6.2 Scenario Parameters

Specific values and ranges were chosen based on our previous experiences (Sec. 3.5), to optimize the positioning of 64 antenna elements over a given lattice. Results shown in Sec. 3.6.4 are based on a fixed lattice with 100×99 antenna elements spaced apart by d_y, d_z. Regarding the generation of the activation mask, the probability profile is parameterized by α_y, $\alpha_z \in [-1, 10]$. Values are chosen to allow a very wide range of possibilities, including extremely sparse ones. Please note that increasing values of α tend to push active antennas together towards the center, while negative values tend to push them towards the outer edges of the lattice.

3.6.3 Learning Methods

The problem we are facing is a numerical regression on synthetic, noisy data.

Several algorithms were tested, but only a selected subset will be hereby described. The performance of the different techniques is evaluated using 5-fold cross-validation, and, as we are interested in keeping the training set as small as possible, the comparison is made for different training set sizes. Thus, it is possible to know the accuracy of the emulation, based on the number of available samples.

- *Linear regression* is the most basic class of regression algorithms. Despite its simplicity, many versions and adaptations have been created, able to solve non-trivial problems. It is often considered as a baseline for more powerful algorithms. Adding a ridge regularization to the linear regression helps avoid overfitting the training data by imposing a penalty on the size of the weights.

- *Random forests* are ensembles of decision trees, that approximate stepwise the target function;

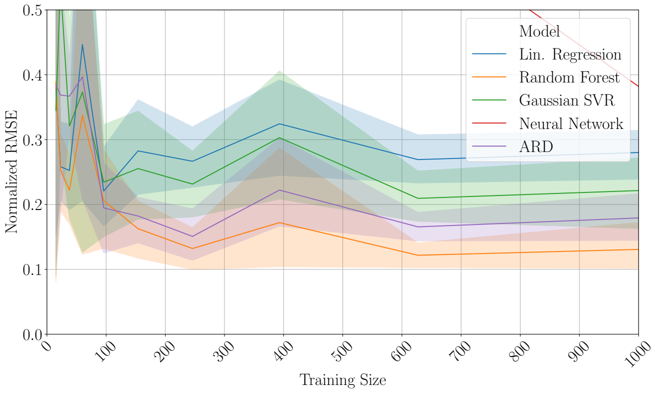


Fig. 3.8: Plot showing cross-validation scores on the nRMSE metric with increasing training size.

- *SVRs* are derived from the SVM classification algorithm. Among all the typical kernels, the Gaussian one performed best and is used here.
- *Automatic Relevance Determinations (ARDs)* directly derive from Bayesian Ridge Regression, but includes a sparsity assumption in the priors which stabilizes the weights;
- *Multi-Layer Perceptron (MLP)* is a well-known architecture that should be able to approximate any function. Nevertheless, MLPs generally require (i) long and computationally-demanding hyperparameter tuning and (ii) large datasets.

The performance is evaluated using the nRMSE metric, as in Sec. 3.5. Results are reported in Fig. 3.8.

3.6.4 Optimization Results

The algorithms described is Sec. 3.6.3 were evaluated for increasing dataset sizes. During the creation of the dataset, monitoring the learning performance as the number of available samples increases can help find a plateau of the learning process, allowing to stop the simulations when the required precision is achieved. In fact, the prediction performance is fundamentally limited by the noise in the given dataset, mainly caused by the limited number of Monte Carlo iterations. As the prediction residuals are symmetrically distributed around zero, this should not affect the generalization performance of the model. Based on the comparison of the described algorithms in Fig. 3.8, we decided to use *Random Forests* for our emulator as they give the best results even with as few as 500 data points.

The objective function chosen for this problem optimizes the average performance of the given network (mean SINR) while imposing a minimum coverage level, corresponding to a lower bound to the 5^{th} percentile of the SINR as follows

$$\begin{aligned} &\text{maximize} && \overline{\text{SINR}} \\ &\text{subject to} && \text{SINR}_5 > 6\text{ dB}, \end{aligned} \tag{3.4}$$

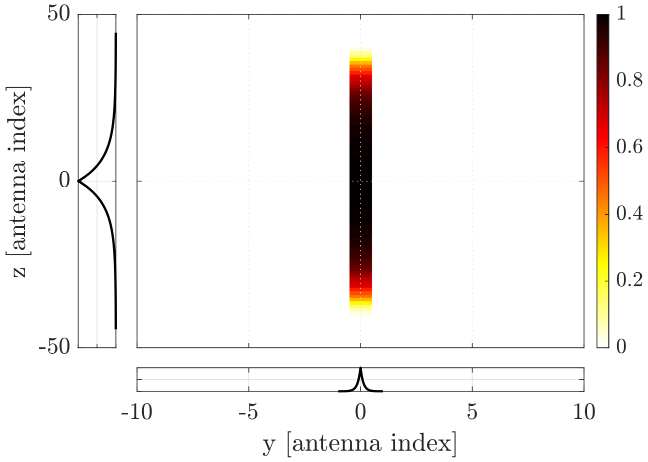


Fig. 3.9: Visual representation of the activation probability of any given element from the lattice using the optimal parameters α_y^*, α_z^*. Elements outside the central column are never activated, indicating that a vertically-shaped antenna is optimal.

and it is the same one proposed in (3.3) in Sec. 3.5.

Given the description from Sec. 3.6.1, it should be noted that the outcome of this optimization problem does not yield the best possible antenna for the given scenario, but rather a family of antennas following a probability distribution obtained for the optimal parameters α_y^*, α_z^*, together with the optimal lattice spacing d_z^*, d_y^*.

For comparison, we consider as the baseline antenna an 8×8 UPA with $d_z = d_y = 0.5\lambda$ spacing. Also, we compare the results with the optimal antenna previously found in Sec. 3.5, given by a vertical linear array of 64 elements, with $d_z = 0.796\lambda$.

The ML-based optimization framework suggests as the optimal parameters $\alpha_z^* = 9.02$, $\alpha_y^* = 0.20$, $d_z^* = 0.761$, $d_y^* = 0.866$. To better understand what these parameters suggest, Fig. 3.9 shows the probability that any given antenna in the lattice is active, together with the corresponding probability profiles. It can be easily noted that the activation probability indicates that vertical antennas tend to perform better than any other configuration, similarly to what was found in our previous work where the 64×1 configuration was identified as optimal.

As these results do not identify a specific antenna, but rather a family of antennas, in Fig. 3.10 we show a comparison between (i) two specific antennas used as references, (ii) antennas generated using non-optimized (i.e., randomly selected) input parameters, and (iii) antennas generated using the optimal ones. Note that, while the antennas from the optimal family do not perform equally, they always achieve significantly better performance with respect to the baseline and to the other configurations and closely approach the optimal antenna found in Sec. 3.5, often improving over the SINR_5 although not over the $\overline{\text{SINR}}$. Though the input parameters optimized by the framework do not directly identify a specific antenna configuration, they allow to drastically reduce the search space to a much narrower area, that can be further explored using more traditional, time-consuming techniques.

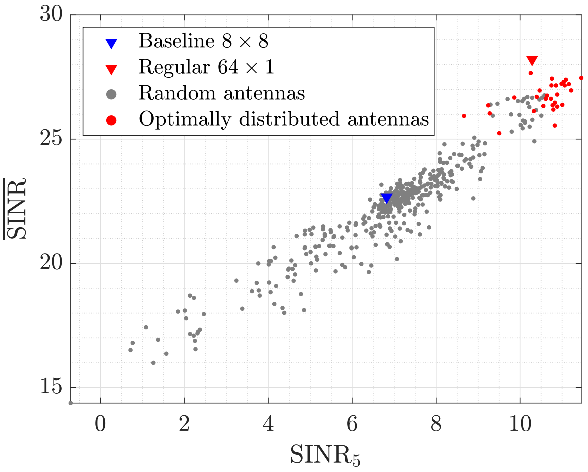


Fig. 3.10: Performance comparison between different classes of antennas. Gray dots are obtained choosing 300 antennas generated from non-optimized input configurations from the input space described in Sec. 3.6.2. Red dots, instead, show 30 antennas, all generated with the optimal input configuration. Blue and red triangles represent the baseline and the optimal antenna found in Sec. 3.5, respectively.

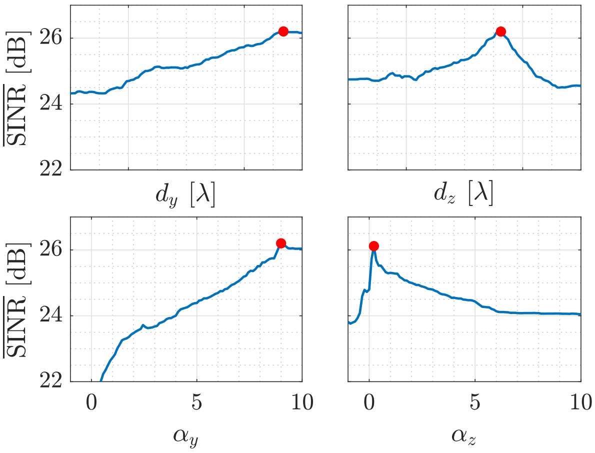


Fig. 3.11: Visualization of the input configuration around the optimal value. Slices of the input configuration are taken to show how the different parameters affect the performance. It is clear that the antenna shape parameters α_y, α_z play a more important role than the lattice-related parameters d_y, d_z.

Finally, we can study the sensitivity of our optimal point with respect to the four input parameters in Fig. 3.11. As expected, α_y^* is chosen to be large, forcing the antenna to be vertical. Instead, while a large value of α_z^* would push the elements towards the center to make it less sparse, it would also tend to make the antenna more rhomboidal. The optimization was thus able to find the largest possible value for the vertical sparsity that still allowed all antennas to be strictly vertical. Given the preference for a vertical antenna, the horizontal spacing d_y^* is the parameter with the least effects on the network performance. The vertical spacing d_z^* is instead similar to the one previously found.

3.7 Antenna Array Modeling on ns-3

Typically, antenna and Beamforming (BF) design is carried out as an independent task, by means of real-world experiments or link-level simulations, without considering it as part of the overall system optimization. However, the solutions obtained with this approach may not be able to achieve optimal system-level performance, because they are designed without considering the interactions between the antenna systems and the higher layers of the protocol stack.

To go beyond this standalone block-level design perspective, new tools able to properly consider all the relevant aspects of the cellular system are required. For instance, the authors of [109] verified the importance of carrying out system-level simulations for the design of an 8×2 hybrid beamformer, since the cross effects between the different system blocks may have a strong impact on the overall performance. In [110], the authors present novel antenna array and BF solutions for mmW MIMO systems based on lens antennas, and evaluate the end-to-end performance through system level simulations based on ray-tracing. Also, in [111] the authors investigate the possibility of co-designing the antennas and the RF blocks in the front-end using a system-level platform. Although these works tackle antenna and/or BF design with a system-level approach, they make use of closed-source software or unavailable tools, specifically developed for a single application.

In this section, we propose new models for the end-to-end performance evaluation of antenna and BF designs targeted for mmW cellular systems. Thanks to the integration with ns-3, these models allow users to evaluate the impact of novel antenna and BF solutions on the end-to-end system behavior. Sec. 3.3 describes the antenna array model, Sec. 3.3 describes the antenna element model, while Sec. 3.3 describes the BF model.

3.7.1 ns-3 Simulation Setup

We carried out a simulation campaign to evaluate the performance of different antenna and BF configurations. To this aim, we used the ns-3 mmW module extended with the proposed modeling framework. The scenario we considered is similar to the *Parking Lot* scenario described in Sec. 2.4.1, and is also reported in Fig. 3.12. It models a parking lot with multiple cars (between 1.2 and 2.25 m high) and buildings. Two mmW Base Stations (BSs) providing cellular coverage are placed on the front face of two buildings at a height of 3 m and are oriented with a

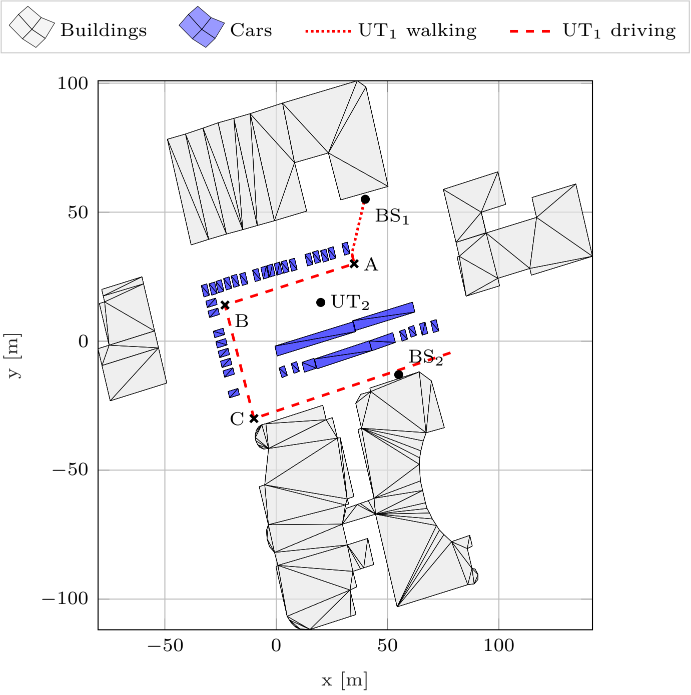


Fig. 3.12: Reference scenario.

bearing angle in the direction normal to the wall and with a downtilt of 12° with respect to the horizon. Two users, UT$_1$ and UT$_2$, both at a height of 1.5 m, are connected to the respective BS. During the simulation, UT$_1$ leaves the main building walking at 1.2 m/s up to point A and then starts driving towards the exit of the parking lot at 4.2 m/s, while UT$_2$ stands still at the center of the scenario. The channel was ray-traced every 5 ms considering up to 2$^{\text{nd}}$ order specular reflections and diffuse scattering, but ignoring diffraction effects. More details on the ray-traced scenario can be found in [92]. The same type of BF schemes is used by all nodes of the scenario. We assume perfect channel knowledge for SVD BF, computed for every received and transmitted packet. Instead, to assess the impact of realistic mobility on this type of scenario, codebook-based BF is only updated to find the best codeword pair for each TX/RX node pair every {10, 100, 1000} ms.

Codebooks have been generated ensuring that adjacent beams cross at 3 dB below the maximum directivity and with no tapering across antennas. The system operates at 28 GHz with a bandwidth of 400 MHz, and is configured with NR numerology index 2. The downlink traffic is generated by a remote server which transmits UDP packets to the users at a constant rate. Table 3.2 summarizes the parameters used in our evaluation.

To evaluate the communication performance, we considered both link-level and end-to-end metrics, including SINR and SNR experienced by UT$_1$, respectively showing the performance with and without the interference from the second cell, and APP layer throughput.

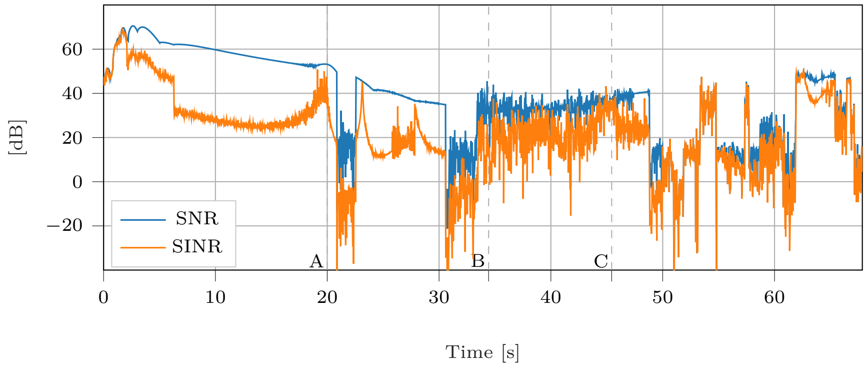


Fig. 3.13: Temporal evolution of the signal quality experienced by UT$_1$.

3.7.2 Simulation Results

In this section, we present and comment the results obtained. Unless explicitly stated, we consider the baseline simulation to have 4×2 arrays for the UEs, 3GPP antenna elements for the BSs, and codebook-based BF with 100 ms beam alignment, in addition to the parameters shown in Table 3.2.

In Fig. 3.13, we reported the temporal evolution of the SNR and SINR experienced by UT$_1$. During the first part of the simulation, the SNR stays always above 50 dB and decreases as the user walks away, but the presence of interference strongly affects the channel quality, as shown by the behavior of the SINR. At time instant A, the user starts driving towards the exit of the parking lot. Shortly after 20 s and 30 s, some of the parked cars temporarily block the line of sight, making the channel quality suddenly drop. From time instant B to time instant C, both the SNR and the SINR show an oscillating behavior caused by the presence of multiple reflections with similar path losses from the surrounding cars. The last part of the simulation is characterized by multiple blockage events due to the cars parked in the bottom part of the parking lot. During this phase, the SNR and SINR exhibit similar behavior since the user is no longer subject to the inference caused by the communication between BS$_2$ and UT$_2$.

Tab. 3.2: Simulation parameters.

Frequency	28 GHz
Bandwidth	400 MHz
Channel sampling period	5 ms
NR numerology index	2
Transmission power	30 dBm
Noise figure	9 dB
BS array size	8×2
UE array size	{1×2, 4×2, 4×2}
BS element pattern	{Isotropic, 3GPP, Cosine}
UE element pattern	Isotropic
BF algorithm	{SVD, Codebook}
Codebook BF period	{10, 100, 1000} ms
APP packet size	1490 bytes
Inter-packet interval	$\{10, 1000\}$ μs
RLC mode	Acknowledge Mode (AM)

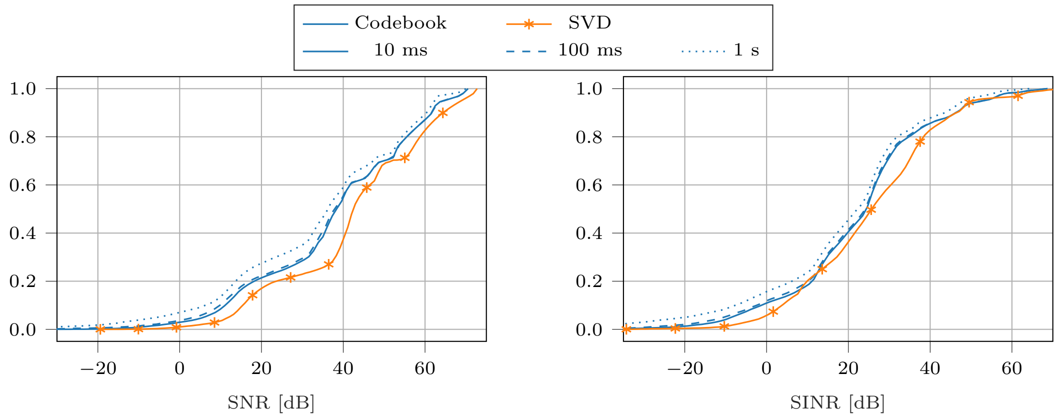


Fig. 3.14: Comparison of the SNR/SINR CDFs for different BF schemes.

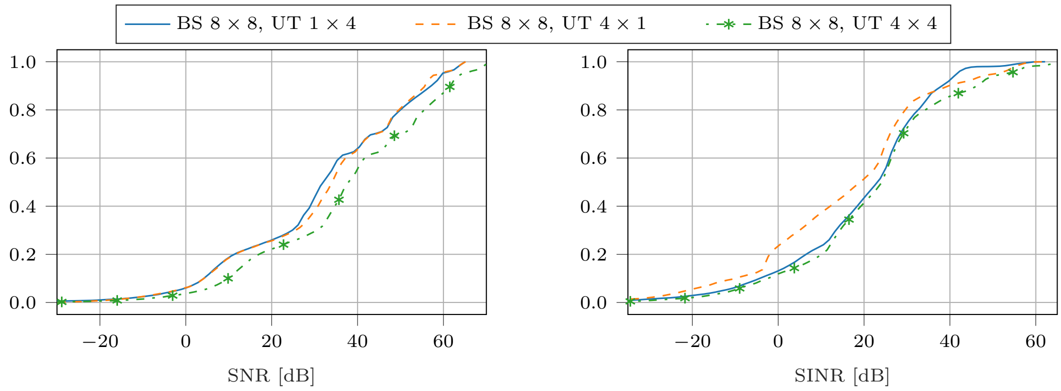


Fig. 3.15: Comparison of the SNR/SINR CDFs for different phased antenna array configurations at the UT side.

Fig. 3.14 shows the CDFs of SNR and SINR experienced by UT_1 with different BF configurations. We can notice that the SVD approach guarantees the best performance in terms of SNR, as supported by the theory, but not always when considering the SINR, i.e., when interference is considered. Since SVD BF does not account for interference when computing the BF vectors, while codebook BF does so when probing the different codeword pairs, the performance gap between the two approaches is reduced and SVD may even be suboptimal, as shown in Sec. 3.7.2. Moreover, it can be seen that the value of the refresh rate used to update the weight vectors affects the behavior of the codebook-based algorithm, providing better performance for more frequent updates. Due to the geometry of the environment and the mobility, diminishing returns are clearly visible when reducing the beam alignment period from 100 ms to 10 ms making the extra overhead unnecessary.

Fig. 3.15 shows a comparison between different array sizes for the UEs. Clearly, the most complex configuration represented by a 4×2 array is able to achieve the highest performance for both SNR and SINR. This is due to the higher antenna gain obtained with the larger antenna

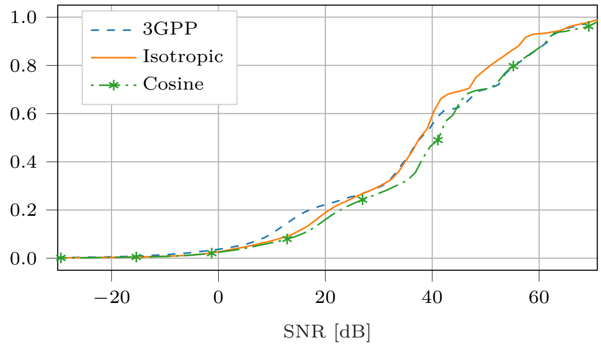


Fig. 3.16: Comparison of the SNR CDFs for different antenna element patterns.

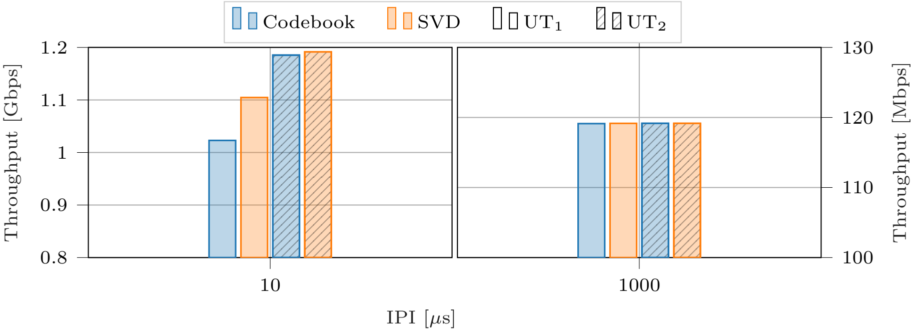


Fig. 3.17: APP-layer throughput for different inter-packet intervals.

array, but also to the reduced interference due to the higher directivity. On the other hand, considering vertical 4-element ULAs results in a very similar performance in the interference-free scenario, but vastly different performance when considering the interfering cell. In fact, a vertical array is only able to produce directivity with cylindrical symmetry around the vertical axis. Being both BSs at the same height, a good BF codeword able to improve the received power will also be likely to increase the downlink interference from the second cell. On the other hand, when orienting the linear array horizontally, the cylindrical symmetry will also rotate over the horizontal axis. In this case, the geometry of the environment and the positioning of the BSs make it less likely to incur strong interference.

Fig. 3.16 evaluates the impact of the element radiation pattern on the SNR experienced by the user. Isotropic elements radiate equal power in all directions, and therefore provide a low directional gain, but are able to cover a wide area. On the contrary, elements characterized by the 3GPP pattern have high directivity but small beamwidth, which implies that the transmitted power is focused in a small portion of the space. The best performance is achieved with the cosine pattern set to have a 3 dB beamwidth of 120°, thus obtaining a maximum gain $G_{\max} =$ 5.7 dBi, as this represents a good compromise between directivity and beamwidth.

Fig. 3.17 shows the average throughput achieved by UT$_1$ and UT$_2$ at the APP layer. With an inter-packet interval of 10 μs, the network is highly loaded and the scarcity of radio resources may prevent the recovery of the lost packets, e.g., by means of MAC and RLC layer retransmissions. In this situation, the choice of the BF algorithm may have a strong impact on the end-to-end performance, especially in the presence of user mobility. Indeed, as shown in Fig. 3.17, the higher channel gain provided by the SVD-based algorithm allows UT_1 to achieve higher throughput, while there is no benefit for UT_2 since it stays in the same position during the entire simulation. Instead, with a higher inter-packet interval, the codebook-based algorithm achieves the same performance as the SVD, since the recovery mechanisms at the MAC and RLC layers are able to compensate for the lower channel quality.

3.8 Conclusions

In this chapter, an innovative framework has been presented that makes the joint optimization of multiple parameters a reality, needing just a fraction of the time that is currently required when directly employing a simulator [112, 113]. As simulators are generally computationally complex and time-consuming, the key idea is to bypass them using a fast emulator, obtained through ML techniques. After a long, initial database creation, any objective function can be optimized in a matter of minutes or even seconds. The effectiveness of this methodology has been proved using a network simulator, which requires a long time to compute the network metrics for specific antenna configurations, thus representing the perfect testbed for our framework.

In the first part of this work, our framework found a vertical ULA to be the optimal configuration, confirmed by empirical results on a large database, showing a ~ 3 dB improvement over the baseline. Furthermore, we discussed about how our methodology can obtain this optimization over 10 times faster with respect to common simple optimization techniques. A key advantage of our approach is the possibility of changing the objective functions of the optimizer, which would be easily and quickly done with the emulator, without having to retrain it.

In the second part of this work, thanks to the antenna parameterization chosen in this study, our framework was able to explore much more complex configurations than regularly spaced planar arrays. Returning an optimized family of antennas rather than a specific configuration successfully reduces the search space of possible configurations, making it possible to further refine it with more precise simulations. Overall, in both studies based on a 3GPP-compliant UMi scenario with static users, the optimizer suggests that vertical linear arrays are the optimal configuration.

We then presented a modeling framework for the end-to-end evaluation of 5G mmW cellular networks which is compliant with the 3GPP NR specifications [114]. Our work extends the capabilities of the ns-3 mmW module presented in [19] by providing (i) a ray-tracing based channel model for mobile users, which improves the spatio-temporal coherence over the previous stochastic channel [43], (ii) a flexible antenna module, comprising multiple parametric antenna elements as well as a generic interface for phased antenna arrays, and (iii) a BF module supporting different algorithms for the computation of the optimal BF vectors. Using this framework, we evaluated the performance achieved by different antenna configurations and BF schemes in a realistic simulation scenario. Our results show that inaccurate antenna and BF designs may provide sub-optimal channel gains and affect the performance of the higher layers.

Future research directions for the optimization framework call for further studies on how to reduce the number of required training samples, in order to further reduce the dataset creation overhead. Moreover, a second aspect would be to increase the accuracy of the emulators, possibly resorting to more complex ML techniques. The range of applicability of the framework, concerning both the complexity of the involved simulator and the number of parameters to be optimized, is left for future studies with different problems. Finally, understanding how to automatize the whole process of data acquisition, formulating a stopping criterion related to the emulator's accuracy, periodically re-train the emulator, and optimize the objective function would create a more usable software, while also solving non-trivial problems that would need to be heuristically solved by hand otherwise.

Regarding the antenna models in ns-3, the proposed framework can be extended in multiple ways, e.g., with better support to antenna polarization and rotation, with a more realistic beam-management implementation to account for the overhead introduced by beam search operations, and it should be validated with real-world measurements to ensure the correctness and credibility of the model. We want to highlight the fact that our model has been integrated in the official release of the ns-3 software starting from version ns-3.34.

Clearly, merging the two main proposals together could also be of interest, i.e., optimizing the antenna parameters based on end-to-end metrics extracted from full-stack simulations, using our antenna module implemented in ns-3.

4
WiGig MAC Scheduling

4.1 Introduction

Wi-Fi is nowadays present in many devices and is common in households, offices, public institutions, and transportation. Over more than 20 years, many amendments have been made to the original standard, updating both the PHY and MAC layers to provide higher bit-rate, robustness, and Quality of Service (QoS).

Indoor Wi-Fi networks have had a key role in the digital revolution of the last two decades, as wireless technologies paved the way toward the design of applications for work settings (e.g., smart metering, remote control) and house entertainment (e.g., AR, VR, XR). From a technical point of view, these new applications also changed the infrastructure requirements, with higher required data rate, lower delay thresholds, and brand new classes of QoS constraints.

To face these challenges, moving to the mmW spectrum has proven to be a valuable alternative to the widespread sub-6 GHz spectrum used by legacy wireless architectures, given the abundant bandwidth available in the former frequency range. On the downside, given the higher carrier frequency, mmW transmission suffers from an increased propagation loss, as well as deeper diffraction shadows, and higher penetration and reflection losses, making communication more difficult and less stable.

On the other hand, these characteristics allow for extreme spatial reuse, e.g., transmissions in different rooms will hardly interfere with each other unlike in legacy Wi-Fi. Moreover, the short wavelength makes it possible to use antenna arrays with tens of elements packed in a small area, making it possible to counteract the increased path loss by focusing the radiated power into directive *beams*, thus increasing the overall antenna gain. While this further reduces interference even where users share the same area and improves spatial reuse, it also creates the problem of directional deafness, worsens the hidden node problem, and makes mobility more complex to handle.

In an effort to create a common playground for researchers and manufacturers, the IEEE devised specific amendments to update the PHY and MAC layers in what is known as WiGig, first with 802.11ad [16] in 2012 and now with 802.11ay [17]. By taking advantage of techniques such as channel bonding and MIMO, and by introducing novel features to the protocol stack, the latest standard can provide data rates over 100 Gbps [115].

In particular, WiGig standards introduced a new contention-free strategy to access the transmission medium at specific time intervals, referred to as Service Periods (SPs). A Station (STA) can request SPs to the Personal Basic Service Set (PBSS) Central Point/Access Point (PCP/AP) asking for a specific duration and periodicity. A detailed overview of such procedure will be later described in Sec. 4.3.3.

This new access strategy can be useful for applications with stringent QoS requirements, i.e., throughput, delay, and jitter, which may be heavily affected by legacy, contention-based channel access mechanisms. Moreover, applications such as video streaming or VR can generate periodic traffic, whose performance with contention-based channel access can degrade, given the uncertain availability of resources from one time interval to another. Fortunately, WiGig provides specific scheduling mechanisms to directly support periodic applications with tight QoS constraints.

Regarding the practical design, handling multiple periodic traffic streams can be problematic, especially when traffic flows with different periodicities coexist. In this case, it is necessary to anticipate collisions among different periodic allocations and either adjust them or, in the worst case, reject new incompatible requests. Furthermore, upon receiving a new request, the scheduler needs to decide whether to rearrange the previously allocated resources to improve fairness and efficiency, or to maintain the original schedule and then best accommodate the new request, in order not to perturb the pre-existing streams but potentially reaching a suboptimal resource allocation. Moreover, SPs are subject to a number of constraints, described in Sec. 4.5, which need to be accounted for when designing and optimizing scheduling algorithms.

Considering all these aspects, in this chapter we propose two main contributions:

1. An End-to-End (E2E) framework to manage distinct traffic flows based on the requirements provided by the WiGig standards, taking care of the admission and scheduling of new allocation requests. To do so, we extend the module described in [116], which integrates into ns-3 the new features of 802.11ad, and publicly release the source code to the research community.

2. We address both admission control and resource allocation for multiple periodic traffic sources, following the constraints given by the WiGig standards. Specifically, we cast the periodic scheduling problem within the WiGig allocation framework and design a simple and efficient algorithm to check for the feasibility of a new request. We then propose a simple admission control algorithm with limited scheduling capabilities, as well as a more elaborate and optimized strategy to increase the admission rate and, possibly, the fairness among independent flows.

The remainder of this chapter will be structured as follows: in Sec. 4.2 we will report the relevant state of the art regarding IEEE 802.11 performance and scheduling, with a focus on WiGig standards; then, in Sec. 4.3 we will give a brief overview of the IEEE 802.11ad standard specifically, focusing especially on the scheduling aspect; after that, in Sec. 4.4 we will describe our work regarding the implementation in ns-3 of a flexible scheduling framework on top of the previously existing IEEE 802.11ad module [116, 117]; our simulation campaigns suggested us to better study the mathematics behind periodic scheduling in WiGig, described in Sec. 4.5; finally, Sec. 4.6 will conclude the chapter, also indicating some relevant research directions.

4.2 State of the Art

The optimization of Wi-Fi's MAC layer procedures has been investigated in the literature, even before WiGig standards were introduced. Most of these works, however, mainly focus on Contention-Based Access Periods (CBAPs) and do not consider the possibility of using SPs. Starting from 802.11ad, the possibility of allocating contention-free resources gained further momentum, considering also the directional characteristic of mmW channels. An attempt to prioritize the traffic injected in the network was made for IEEE 802.11e, where four Access Categories (ACs) were introduced. Based on which category they belong to, packets with higher priority use a shorter Arbitration Inter-Frame Space (AIFS) and thus they wait less before being transmitted. A study of 802.11e contention-based prioritization mechanisms was provided in [118].

A mathematical framework to analyze E2E metrics in 802.11-based systems was proposed in [119], to compare throughput and average packet delay in scenarios where the nodes are equipped with advanced antenna systems. It also accounts in detail for the characteristics of the Distributed Coordination Function (DCF), for which a theoretical performance analysis was carried out in Bianchi's seminal work on IEEE 802.11 performance analysis [120].

Likewise, the authors in [121] presented a detailed analytical model to assess the performance of CBAPs in 802.11ad, taking into account a directional channel model and the presence of scheduled SPs. Yet, the model lacked the details about how to schedule such SPs for certain types of relevant applications, such as periodic ones.

A seminal study on the use of Reinforcement Learning (RL) to solve the problem of jointly scheduling CBAPs and SPs in 802.11ad is in [122], where ns-3 was used to assess how the algorithm could decrease the Data Transmission Interval (DTI) occupancy while guaranteeing state-of-the-art QoS performance.

In general, in the context of WiGig networks, little work has been done on the scheduling of contention-free time resources. Moreover, to the best of our knowledge, little to no work in the literature faces the problem of periodic scheduling with all the constraints introduced by WiGig standards. The authors of [123, 124], for example, study the case where all SPs are allocated at the beginning of each Beacon Interval (BI), while the rest of the interval is left for a single CBAP. In [125] they propose an accurate mathematical analysis of the performance of a realistic Variable Bit Rate (VBR) traffic source in the presence of channel errors when using a periodic resource allocation scheme. How to schedule multiple allocations at once, however, has not been detailed.

On the other hand, the problem of periodic scheduling has been studied for real-time computation and task scheduling, where the goal is to complete some tasks before a certain deadline while minimizing resource utilization. For example, the authors of [126] proposed a scheduling algorithm to dynamically assign priorities, capable of achieving full processor utilization. In [127], the authors tried to schedule safety-critical periodic tasks with precedence constraints, distributed over multi-processor systems using an adapted deadline-first approach, while the authors of [128] used simulated annealing to optimize a similar problem. Finally, [129] found a low-overhead optimal solution (from a resource utilization point of view) assuming that tasks have a fixed resource requirement.

All these approaches, however, cannot be directly used in WiGig systems, either because they are not compliant with the constraints imposed by the resource allocation procedures (i.e., granularity of the allocation periods, BI boundaries), or because they cannot exploit the flexibility of the WiGig standards (e.g., the dynamic allocation of T_{blk}). Part of this work contributes to fill this gap by proposing admission control and scheduling algorithms that account for the specific features of mmW WLANs.

Other works in the literature consider different aspects of the DTI. For example, [130] derives the theoretical maximum throughput for CBAPs when two-level MAC frame aggregation is used. Beamforming is also considered in [131], which proposes a joint optimization of beamwidth selection and scheduling to maximize the effective network throughput, while other works, though not specifically concerning IEEE 802.11ad, deal with transmission scheduling for mmW communications [132].

4.2.1 Available Research Tools

Although commercial devices supporting the IEEE 802.11ad standard are currently available, manufacturers do not provide tools to access low-level functionalities, although some alternatives exist (e.g., [133]) and some work has been done to that regard [134, 135, 136, 137, 138, 139, 140, 141, 142]. Ultimately, it is more flexible, timely, and cost-effective, albeit being arguably less realistic, to simulate the behavior of such devices.

In particular, significant effort has already been done implementing the IEEE 802.11ad standard into ns-3 [117], a popular open-source network-level simulator. The last release of the simulator also supports quasi-deterministic channel modeling based on ray-tracing, making simulations extremely accurate and realistic at the cost of a long preliminary channel generation phase, although some works already tried to improve this aspect [87]. While the implementation already covers most parts of the standard, it is still missing the scheduling mechanisms necessary for this project. The authors of [117] are also working on the implementation of the IEEE 802.11ay amendment [143], making their work even more valuable.

With these powerful tools, it will be possible to further advance the state of the art, create a comprehensive performance evaluation of available algorithms and further improve upon them once the weak points are clearly identified.

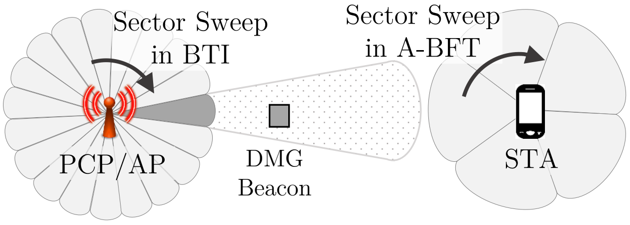


Fig. 4.1: Graphical representation of sector structure in IEEE 802.11ad.

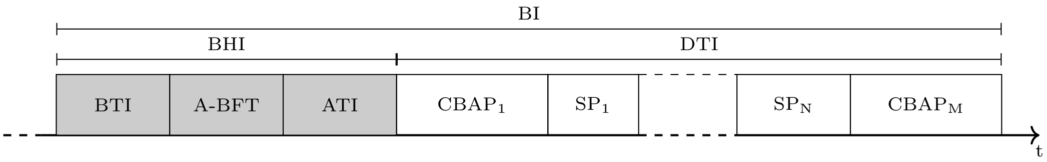


Fig. 4.2: Representation of a Beacon Interval (BI).

4.3 IEEE 802.11ad Overview

To introduce the main concepts and nomenclature of IEEE 802.11ad, in this section we provide a short summary of the standard [144], while referring to other sources for more details [2].

Being a mmW-based standard, directional communication with all the added overhead and the possibility of spatially multiplexing users are included in the amendment. To simplify beam management, both the PCP/AP and the STAs divide their surrounding space into sectors as shown in Fig. 4.1. STAs and PCP/AP will then need to keep beam alignment, which increases the signaling overhead.

Fig. 4.2 shows that in IEEE 802.11ad time is divided in BIs, the unit time interval used by the devices to organize association, beamforming, and data transmission procedures, of about 100 ms. Each BI is further divided into Beacon Header Interval (BHI) and DTI, briefly described in the following sections.

4.3.1 Beacon Header Interval

The PCP/AP does most of the managing, such as beaconing, beamforming training, and scheduling, during the BHI. This period can last hundreds of microseconds up to a few milliseconds, and is further divided into three subintervals: Beacon Transmission Interval (BTI), Association-BeamForming Training (A-BFT), and Announcement Transmission Interval (ATI).

The BTI is used to send Directional Multi-Gigabit (DMG) Beacons to announce the network, give the basic synchronization and BI structure information, start the beamforming training with new stations, and, if needed, do some basic traffic management. Beacons are sent over the different sectors, covering all possible directions to maximize coverage for untrained STAs.

After receiving a DMG Beacon during the BTI, new STAs can use the A-BFT to complete the basic beamforming training by sending Sector Sweep (SSW) frames in different sectors.

Beam alignment is completed once the PCP/AP responds with an SSW Feedback.

Finally, advanced scheduling mechanisms setup and further network management can be done during the optional ATI.

4.3.2 Data Transmission Interval

The DTI is mainly used for the actual data transmission, but it can also be used to improve communication links and for further scheduling. The DTI comprises Contention-Based Access Periods (CBAPs) and Service Periods (SPs), which can appear in arbitrary combinations and are scheduled during the BHI.

Transmission in CBAP follows the rule of Enhanced Distributed Channel Access (EDCA), slightly modified to account for directional transmission, in which STAs compete with each other in order to transmit their data.

Instead, SPs are scheduled contention-free intervals that are dedicated to exclusive transmission between a pair of STAs* to guarantee QoS. The standard also allows for spatial sharing, meaning that multiple pairs of STAs with low cross-interference can be scheduled in the same SPs. This, however, comes at the cost of increased overhead since interference measurements must periodically take place.

4.3.3 Scheduling in IEEE 802.11ad

IEEE 802.11ad allows for great flexibility in the scheduling of radio resources, but we will hereby describe only some of these possibilities in their simplest form.

We want to stress the fact that, unlike in traditional contention-based medium access, scheduled SPs guarantee QoS. ACs introduced in 802.11e, in fact, only allow for stochastic traffic prioritization according to the DiffServ paradigm, which ceases to work in congested networks. For this reason, allocated traffic is especially important for those applications with strict QoS constraints. Instead, more realistic applications, such as data transfer or asynchronous bursty traffic, can simply rely on CBAP.

As shown in Fig. 4.3, a STA can set up an allocation by sending an Add Traffic Stream (ADDTS) Request frame to the PCP/AP during the DTI and embedding a DMG Traffic Specification (TSPEC) element. The DMG TSPEC element is created by the requesting STA and comprises information such as the allocation period, the minimum and maximum allocation duration, and the pseudo-static flag, which allows for persistent allocations over multiple consecutive BIs, among others.

Based on its admission policy, the PCP/AP will either reject or accept the request, immediately notifying the requesting STA via an ADDTS Response. If accepted, the allocation is made effective by including it in the Extended Schedule Element (ESE) transmitted in the next DMG Beacons, which will contain details such as the effectively allocated period duration and the SP start time. In this way, STAs not involved in the communication will not create interference

*A PCP/AP also *contains* a STA, i.e., *a logical entity that is a singly addressable instance of a MAC and PHY interface to the wireless medium* [144].

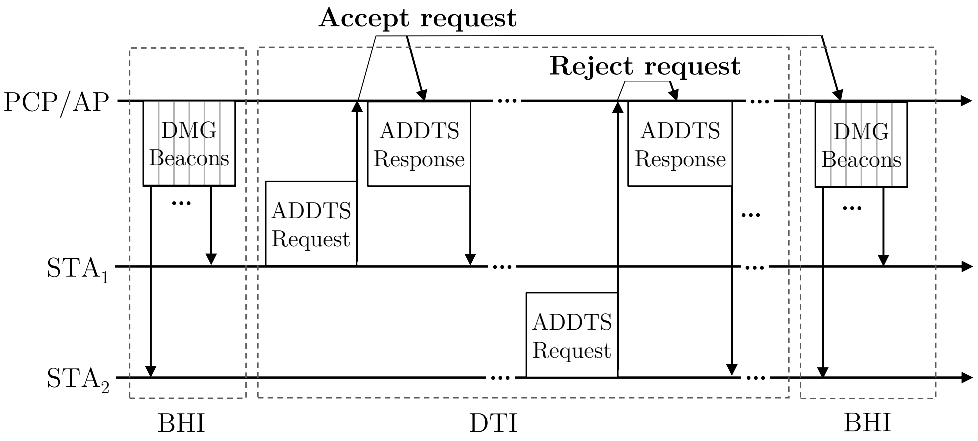


Fig. 4.3: Representation of ADDTS scheduling in IEEE 802.11ad.

and will be able to switch to power-saving mode. Otherwise, the PCP/AP can either reject or propose a change in the DMG TSPEC. A STA can later update the DMG TSPEC by sending another ADDTS Request with the updated element and follow again the same procedure.

Allocating the right duration to SPs is clearly a trade-off between QoS traffic, which needs resources to fulfill the minimum requirements imposed by the application, and elastic traffic, which still needs resources even though with less stringent requirements. Since resource availability, as well as channel quality, are time-varying, the standard supports SP extension and truncation services, which let the stations keep transmitting and/or relinquishing the unused occupied channel. Still, these features bring extra overhead and should thus be used carefully.

As mentioned in Sec. 4.2, a mathematical model for preliminary allocation of SP for VBR flows is presented in [125], which helps determine how to set the TSPEC parameters to meet QoS requirements while minimizing the amount of allocated time. Unfortunately, SPs are assumed to be placed at the beginning of the DTI, which is not possible in general for applications with tight delay constraints. For example, for virtual or augmented reality services, latencies should be below 20 ms to avoid motion sickness.

4.4 WiGig Scheduling Framework on ns-3

In this section, we describe the design choices and assumptions necessary to implement our scheduling framework on top of the 802.11ad ns-3 module [116], with a focus on MAC layer mechanisms.

Our work [145] mainly focused on the design and implementation of a generic scheduling interfaces, called `DmgWifiScheduler`, that implements the scheduling features for the MAC entity of the PCP/AP. Starting from this class, we extended it to create the `PeriodicWifiScheduler`, a simple scheduler for the allocation of periodic resources. Moreover, to study how a contention-based-only approach affects the overall QoS, we also created the `CbapOnlyWifiScheduler`, forc-

Tab. 4.1: Simulation parameters

MCS	4 (fixed)	APP period (T_{APP})	T
Max A-MSDU size	7 935 B	Packet size	1 448 B
Max A-MPDU size	262 143 B	Traffic direction	Uplink
BI duration (T_{BI})	T	Simulation duration	10 s
SP period (T_{SP})	T	Independent runs	30
Network protocols	IPv4/UDP	T	102.4 ms

ing STAs to transmit only over CBAP by allocating the entire DTI as such.

Even though the performance evaluation, presented in Sec. 4.4.2, considers only allocations with the same period and application requirements for all STAs, scheduled starting from the beginning of the DTI back to back as long as they fit, it is crucial to elaborate on the design choices that lead to this framework.

Thus, `PeriodicWifiScheduler` includes the following assumptions:

- Only SP allocations with period equal to an integer fraction of a BI are supported, while the standards also support periods multiple of the BI.
- If the period is $t = T_{\mathrm{BI}}/p$, the request is accepted only if the available time in the DTI can accommodate exactly p SPs, commonly referred to as allocation blocks, each distanced by t. For example, if $p = 4$, the number of blocks per BI must be exactly 4.
- A STA can send an ADDTS Request to reduce the duration of the allocation, while the increase is not supported as it possibly requires a major reorganization of the DTI.
- Once an allocation is accepted, the SPs duration and blocks starting time cannot be changed by the scheduler, even if the DTI structure changes as a consequence of subsequent requests from other STAs.
- All the time that is not reserved by SPs will be allocated as CBAP.

These constraints allowed us to validate our results in a clear setting with firm requirements.

4.4.1 Simulation Setup

The network scenario consists of a single PCP/AP in the center of a room, surrounded by STAs with perfect channel conditions, with simulation parameters listed in Table 4.1.

To emulate periodic traffic, we implemented a *periodic application* that generates periodic packet bursts, whose size and period can be set as a parameter of the application, with every single packet being of size 1 448 B. Traffic is generated by the STAs and sent to the PCP/AP.

Since we expect CBAP-only scheduling to yield good performance when a small amount of traffic is sent over the network, and the SP scheduling to show its full potential for highly loaded networks, we defined the *normalized offered traffic* which we refer to as η. By varying η in $(0, 1]$, we control the traffic injected in the network, equally distributed among the number of stations.

For instance, in a scenario with $N = 4$ STAs transmitting using Modulation and Coding Scheme (MCS) 4 with a nominal PHY rate of $R_4 = 1\,155$ Mbps, for $\eta = 0.5$ the aggregate average offered traffic should be $\eta R_4 = 577.5$ Mbps, and thus each STA will generate about $\eta R_4/N \approx 144$ Mbps.

Note that with $\eta = 1$, the offered PHY rate would be exactly 1 155 Mbps, thus overloading the network. In fact, a portion of each BI is always reserved for the BHI where STAs are not allowed to transmit information, reducing the overall network capacity. On the other hand, $\eta = 0$ would translate in no traffic injected into the network. For this reason, in Sec. 4.4.2 we will show results for traffic loads $\eta \in [0.01, 0.9]$.

In all our simulations, the period of all periodic applications T_{APP}, the period of all scheduled SPs T_{SP}, and the duration of the BI T_{BI} are all the same, and thus simply noted as $T = 102.4$ ms. Based on the value of η, the number of packets making up a burst is constant as well, and they are all generated at the beginning of each application period.

The duration of each SP is computed based on the MCS and the application rate for the full transmission burst to fit exactly into the SP. The minimum and maximum duration fields in the ADDTS Request are thus equal, meaning that the request is either accepted by the PCP/AP guaranteeing the exact amount of resources necessary to serve its application, or rejected, and the rejected STA will remain silent for all the simulation.

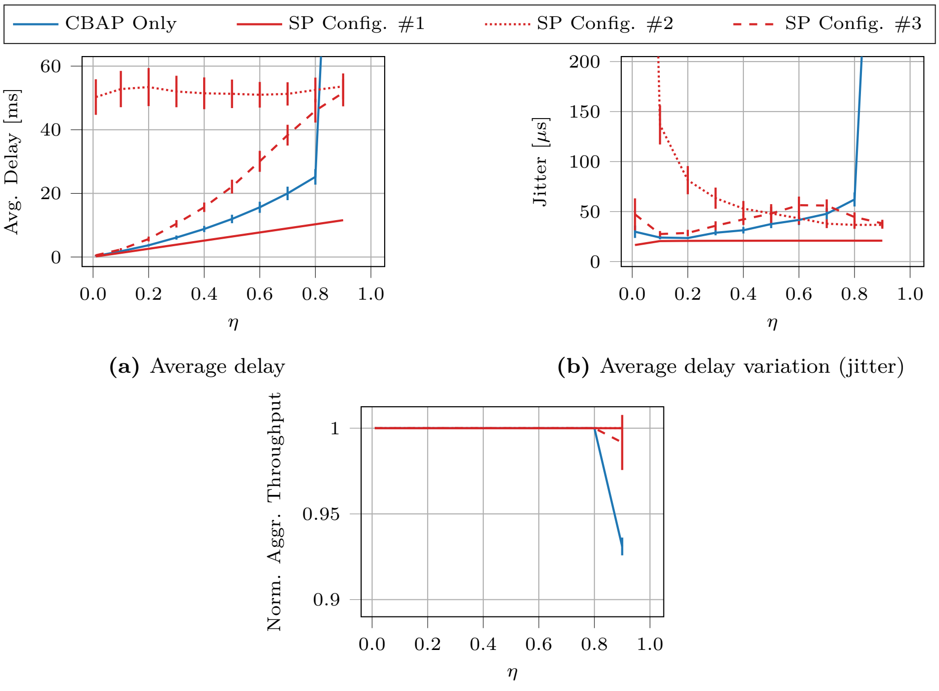
If the ADDTS Response for a given STA is accepted, its application will start randomly over a period T, and thus, by default, will not be aligned with the beginning of its assigned SPs. To fully take advantage of the scheduling concept, however, application and SPs should be aligned to yield the best possible performance. To do so, the APP layer has to be aware that the transmission will happen over a WiGig network as well as the details of the scheduled SPs, requiring some information exchange with the MAC layer. This might be possible for some types of applications running on specific hardware, e.g., VR headsets and, in general, for high-end hardware running applications that require tight delay constraints. For this reason, we defined a *smart mode* which, if activated, makes the application start at the beginning of the first allocated SP, thus assuming a cross-layer interaction and alignment. Nonetheless, this does not take into account applications with non-deterministic periods, which could lose the alignment in the following SPs.

We compare the performance of four scheduling configurations, namely:

- *CBAP-only*: all STAs transmit during the CBAP.
- *SP Config. #1*: the *smart start* mode is enabled. STAs are also allowed to transmit in the CBAP if necessary.
- *SP Config. #2*: *smart start* is disabled and STAs cannot transmit in the CBAP.
- *SP Config. #3*: *smart start* is disabled and STAs are allowed to transmit in the CBAP if necessary.

The performance evaluation of the proposed scheduling schemes has been carried out in three distinct scenarios.

- *First scenario*: four STAs transmit at different values of η using a deterministic application with period T.



(a) Average delay

(b) Average delay variation (jitter)

(c) Normalized aggregated throughput

Fig. 4.4: Performance of the different scheduling configurations with a bursty application with deterministic period $T = 102.4$ ms.

- *Second scenario*: all applications offer the same APP layer rate of $R = 50$, 100, 200 Mbps with a deterministic period of T, varying number of STAs up to 10.

- *Third scenario*: four STAs transmit a heavy traffic load ($\eta = 0.75$) using applications with random period. Periods are independently sampled one after the other $\mathbf{T}_i = \mathcal{N}(\mu, \sigma^2)$, where $\mu = T$ and $\sigma = \rho T$, calling ρ the *period deviation ratio*. Thus, for a given STA, bursts will occur at times $\mathbf{t}_k = t_0 + \sum_{i=1}^{k} \mathbf{T}_i$.

4.4.2 Simulation Results

In this section, we evaluate the performance of the different configurations considering a number of packet-based Key Performance Indicators (KPIs). First of all, the *average delay* takes into account only successfully received packets. For some relevant scenarios we also show the packet *jitter* [146], defined as the average absolute delay variation among successive packets. The *aggregated throughput* is also considered as a metric for network utilization, sometimes normalized by the amount of aggregated offered traffic. Finally, all metrics also show the 95% confidence intervals computed as $1.96\frac{\sigma_{\text{runs}}}{\sqrt{N_{\text{runs}}}}$.

First Scenario Fig. 4.4 shows the results for the first proposed scenario, where we compare the four scheduling configurations against traffic load, considering a deterministic application, as described in Sec. 4.4.1.

In Fig. 4.4a we show the *average delay* for this scenario. Note that an increasing η directly translates into an increased burst size, since more packets have to be delivered in a given period T, thus increasing the achievable average delay.

When the scheduling of SPs is not allowed, CBAP-only offers almost ideal delay performance for low traffic loads, which however degrades for higher loads and even becomes unstable for $\eta > 0.8$.

Instead, SP configuration #1, i.e., using *smart start*, is clearly the optimal strategy and represents a lower bound for all other configurations, since packets are sent immediately and back-to-back.

SP configuration #2, where *smart start* is not used and STAs with scheduled SPs are *not* allowed to access the CBAP, shows an almost constant average delay of about 51.2 ms $= T/2$. It can be proven that an application with period T with a uniformly distributed start time, which can only transmit during an SP of the same periodicity T and with a duration equal to what is needed to transmit the packet burst, has an expected average delay of exactly $T/2$, irrespective of the traffic load or the number of transmitting nodes. In fact, application bursts will happen either (i) sometime during the ongoing SP, so that the next SP will also be needed to finish sending the whole burst causing a large increase of the average delay, or (ii) outside an SP, thus needing to wait for the start of the next SP but being able to send the whole burst at once.

Finally, for SP configuration #3, where *smart start* is not used but STAs with scheduled SPs are allowed to also access the CBAP, the performance is lower bounded by the CBAP-only scheduling and upper bounded by SP configuration #2. In fact, application bursts can either start during an ongoing SP or a CBAP and thus have to be split among different SPs or CBAPs. For low traffic loads η, traffic will mostly be sent during the CBAP, mimicking the CBAP-only scheduler's performance. Instead, considering node k, as η increases, SPs allocated for nodes $\neq k$ will prevent it from freely transmitting over the whole BI, forcing it to either wait for its next SP or to concur with an increasingly busy and shorter CBAP, getting closer to the behavior (and the performance) of SP configuration #2. Contrary to what happens for the CBAP-only scheduler, though, the PCP/AP has a way to control the traffic flow by rejecting ADDTS Requests, preventing the traffic from becoming unstable even for higher loads, at the cost of possibly denying some STAs to transmit.

In Fig. 4.4b we show the *jitter* performance for the first scenario. Again, as expected, CBAP-only scheduling shows an increasing jitter with an increasingly loaded network and becomes unstable for $\eta > 0.8$, while SP configuration #1 shows constant jitter irrespective of the traffic load, always lower than any other scheduling schemes.

Similar to what happened for the average delay, SP configuration #2 has to account for two opposing trends. Note that bursts starting during the CBAP will have extremely low jitter since they will be sent entirely during the next SP. On one hand, a lower η translates to shorter SPs, making it more likely for application bursts to start during a CBAP. Bursts starting during a CBAP will be sent entirely during the next SP, resulting in a low jitter, while those starting during an SP will have to be split among two consecutive SPs, making one packet increase the jitter significantly. On the other hand, a lower η also reduces the burst size and, conversely, the number of packets composing the burst, making the single packet with higher delay variation weigh more in the average and thus affecting the jitter. This second effect appears to be predominant and thus the jitter decreases as the traffic load increases.

SP configuration #3 shows higher jitter than CBAP-only for lower values of η, since other nodes' SPs possibly interfere with the transmission of a full uninterrupted burst, while higher values of η show a decreasing jitter. This suggests that as the CBAP is reduced to leave space for the allocated SPs, nodes will be forced to use it less in favor of their allocated SPs, where transmissions are ensured and more stable but at the cost of a higher delay.

Finally, we show the *aggregated throughput* normalized by the offered traffic in Fig. 4.4c. Clearly, all SP configurations can fully allocate the BI, resulting in unit normalized throughput. The only exception to this is SP configuration #3: allowing allocated users to also exploit the CBAP resources might prevent new users from transmitting in a timely fashion. In fact, for high traffic loads, not only is the CBAP greatly reduced, but allocated STAs also contend for those resources, starving new users who might want to transmit non-QoS traffic or, as it happens in this case, send an ADDTS Request to schedule additional SPs, an event that clearly cannot happen when allocated users do not exploit CBAP resources.

Instead, the CBAP-only scheduler can only withstand the traffic demand for $\eta \leq 0.8$, then, as also noted for other metrics, the Wi-Fi contention mechanism loses its effectiveness making the traffic unstable and starting to lose packets.

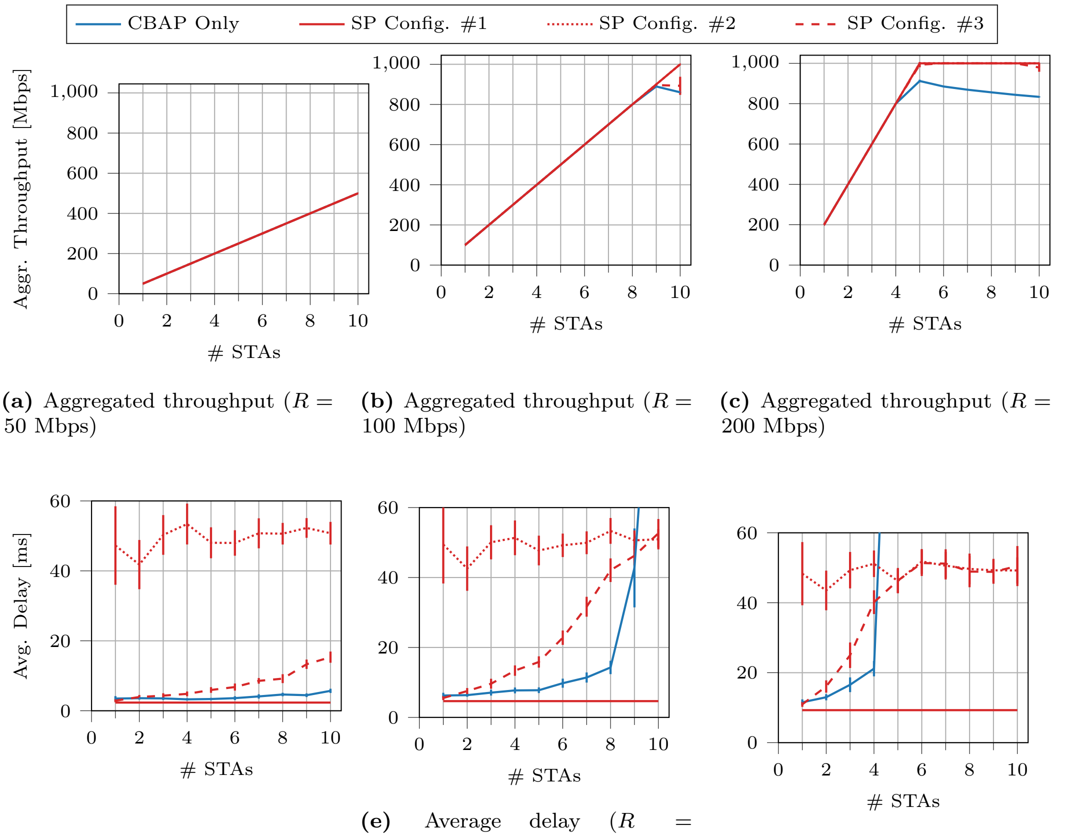
Second scenario In Fig. 4.5 we show the performance for the second scenario, where a fixed application rate was considered with a varying number of users.

Clearly, since MCS 4 was used with a PHY rate of 1 155 Mbps, for rates $R = 50$ and 100 Mbps, all scheduled SP allocations were able to meet the offered data rate (see Figs. 4.5a and 4.5b). Only SP configuration #3 was not fully able to support the full 1 Gpbs as previously discussed for the first scenario. Furthermore, also the CBAP-only case was unable to meet the aggregate demand since 1 Gbps of offered traffic or more corresponds to $\eta > 0.8$ and, as suggested by the results shown for the first scenario, is thus unstable.

Regarding the average delay performance shown in Figs. 4.5d to 4.5f, similar results to the first scenario can be observed.

Using only the CBAP yields good performance for low traffic loads, which in this case corresponds to a lower number of users, while it remains unstable for high traffic loads.

Instead, while SP configuration #1 is the lower bound achievable by any configuration consistently across all cases, SP configuration #2 is the upper bound for all SP configurations. As the offered traffic load increases, i.e., as more STAs transmit with higher application rates R, SP configuration #3 tends to have the same performance as SP configuration #2, as less CBAP is available.



(a) Aggregated throughput ($R =$ 50 Mbps)

(b) Aggregated throughput ($R =$ 100 Mbps)

(c) Aggregated throughput ($R =$ 200 Mbps)

(d) Average delay ($R = 50$ Mbps)

(e) Average delay ($R =$ 100 Mbps)

(f) Average delay ($R = 200$ Mbps)

Fig. 4.5: Performance of the four scheduling configurations using a bursty application with period equal to $T = 102.4$ ms, and an offered rate R for each user.

Third scenario Fig. 4.6 shows the average delay for the third proposed scenario, where we compare the four scheduling strategies considering a load of $\eta = 0.75$ and an application with random period against its *period deviation ratio* ρ, as described in Sec. 4.4.1. Note that $\rho = 0$ coincides with a deterministic application.

As expected, the CBAP-only case is not affected by the random periodicity of the application.

Similar to the first scenario, SP configuration #3 shows worse performance than the CBAP-only scheduler, as users can only transmit in their own SPs or during the CBAP. Since the applications are not synchronized with the SPs to begin with, also in this case the performance is not affected by the random periodicity of the application.

On the other hand, SP configuration #1, appears to be optimal only for almost-deterministic applications, i.e., for extremely low values of ρ. In fact, *smart start* only synchronizes the application with the first allocated SP, meaning that if an application has a random period, bursts starting from the second one will be out of sync. Since we allowed STAs to use the CBAP for SP configuration #1, as ρ increases, performance gets worse reaching the same average delay as SP configuration #3, where cross-layer alignment is not enabled.

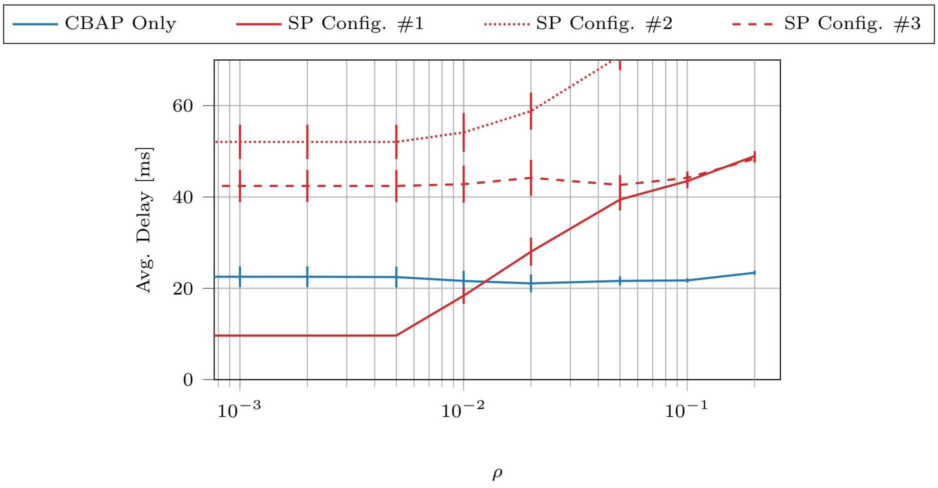


Fig. 4.6: Average delay of the different scheduling configurations with a bursty application with normally distributed period, with mean equal to $T = 102.4$ ms and standard deviation equal to a fraction of its mean $\sigma = \rho T$.

Even worse, SP configuration #2 shows by far the worst behavior. Not only is its performance bad for the deterministic case, but since STAs are only allowed to transmit during their own SPs and the SP duration was computed to be exactly the time required to send the whole burst, the random periodicity of the application further worsens the performance. In fact, if one period is longer than T, part of an SP might never be used, although the average traffic will still require all SPs to be fully utilized. The more random the application, the more likely this event, possibly leaving more and more portions of SPs not utilized.

Results Overview To summarize, the simulation results show that when the network load is low, contention-based channel access is capable of yielding overall good performance, but as the amount of offered traffic increases, average delay and jitter are quickly affected. Instead, SP scheduling shows its full potential only when cross-layer information is exchanged between the APP and the MAC layer, allowing the application to synchronize with the scheduled SPs.

Furthermore, we showed that if (i) the application and the SP allocations share the same period T, (ii) STAs can only transmit during their own SPs, (iii) the SP duration coincides with the time required to transmit the burst, and (iv) the application start time is uniformly distributed over a period T, then the average delay is equal to $T/2$, irrespective of the burst size, the number of users, or the network traffic load.

On top of this, SP scheduling allows the PCP/AP to accept and reject incoming traffic flows, allowing better control of the network even in the most intensive traffic regimes, thus being able to ensure to a limited number of users the required amount of resources without making the transmission unstable, unlike contention-based access alone.

Finally, we showed that small amounts of randomness in the period duration can easily favor the simpler contention-based access over the more complex SP scheduling, but further studies

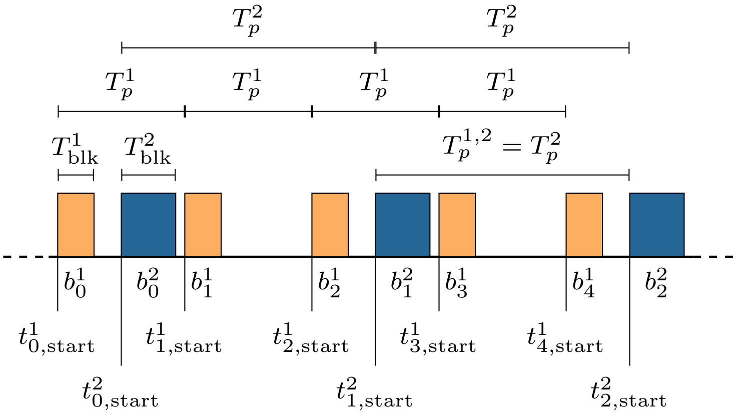


Fig. 4.7: Example of allocations A_1 (orange) and A_2 (blue), with $p_2 = 2p_1$.

need to be done as the setup was extremely simple.

4.5 Mathematical Framework Description

Based on [144], STAs can request the Access Point (AP) to reserve periodic transmission intervals by sending a control frame containing the required periodicity (p) and the minimum and maximum duration of each allocation ($[T_{\text{min}}, T_{\text{max}}]$). The AP advertises the allocated SPs to the STAs at each BI, specifying the starting time, duration, and periodicity of each block. The allocation needs to comply with a number of constraints:

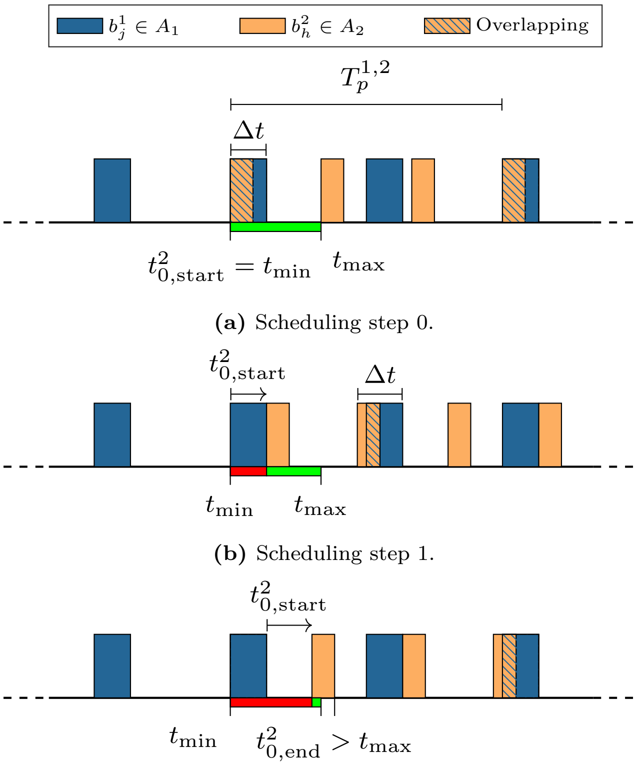
1. Periodicity (p) can only be an integer multiple ($p \in \mathbb{N}$) or an integer fraction ($p^{-1} \in \mathbb{N}$) of a BI (T_{BI}), thus the block periodicity interval will be $T_p = p\,T_{\text{BI}}$;

2. Allocation blocks cannot be scheduled across the BI boundaries;

3. The allocated block duration T_{blk} should fall in the range $[T_{\text{min}}, T_{\text{max}}]$ specified in the resource request.

A more detailed description of the constraints imposed by the standard can be found in [144] and [147].

Since this work is focused on allocation algorithms for periodic traffic sources, we neglect the CBAPs, which is present in each BI for asynchronous traffic. In addition, to compare the scheduling algorithms in challenging conditions, we assume that the allocated resources will be maintained indefinitely, so that the channel load increases progressively as new resource reservations are accepted. For the sake of simplicity and clarity, we also assume that the allocation blocks of a given accepted request are not fractioned into multiple disjoint intervals (i.e., each SP will consist of a single time interval of duration T_{blk}). Furthermore, for ease of analysis, we consider a *strict periodicity* constraint, which prevents the scheduler from changing



(a) Scheduling step 0.

(b) Scheduling step 1.

(c) Scheduling step 2.

Fig. 4.8: Feasibility check for an *infeasible* pair of allocations, where A_1 (blue) was a pre-existing allocation with $p_1 = \frac{1}{2}$, and the algorithm is checking whether a new allocation A_2 (orange) with $p_2 = \frac{1}{3}$ is compatible.

the starting time of already allocated blocks, while the block duration T_{blk} can be freely changed within the interval $[T_{\text{min}}, T_{\text{max}}]$.

We denote by $A_n = (t^n_{0,\text{start}}, T^n_p, T^n_{\text{blk}})$ the allocation for the n-th traffic stream, where $t^n_{0,\text{start}}$ is the starting epoch, T^n_p is the period, and T^n_{blk} is the allocated duration of each individual block. Therefore, the allocation consists of a sequence of blocks, where the k-th block of the n-th traffic stream takes the interval $b^n_k = \left(t^n_{k,\text{start}}, t^n_{k,\text{end}}\right)$, with

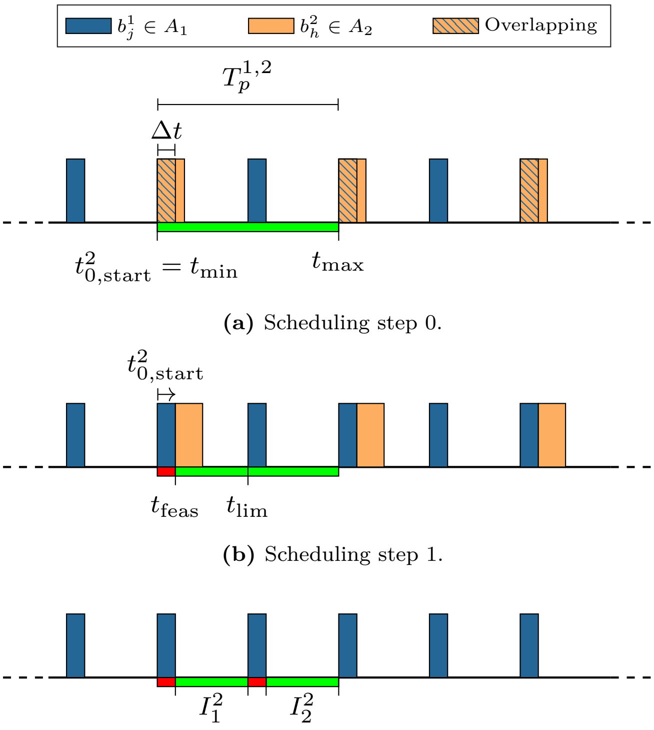
$$\begin{aligned} t^n_{k,\text{start}} &= t^n_{0,\text{start}} + kT^n_p \ ; \\ t^n_{k,\text{end}} &= t^n_{0,\text{start}} + kT^n_p + T^n_{\text{blk}} \ ; \end{aligned} \tag{4.1}$$

for $k = 0, 1, 2, \ldots$. A graphical example is shown in Fig. 4.7.

Following this definition we can say that, given N distinct allocations $A_1, \ldots, A_N$, they are jointly periodic over a period

$$T^{1,\ldots,N}_p = \text{lcm}\left(T^i_p, \ldots, T^N_p\right), \tag{4.2}$$

where lcm indicates the *least common multiple* of the periods. Note that, since all block periods



(a) Scheduling step 0.

(b) Scheduling step 1.

(c) Feasible intervals.

Fig. 4.9: Feasibility check for a *feasible* pair of allocations, where A_1 (blue) was a pre-existing allocation with $p_1 = \frac{1}{4}$, and the algorithm is checking whether a new allocation A_2 (orange) with $p_2 = \frac{1}{2}$ is compatible.

are integer multiples or fractions of T_{BI}, the lcm can always be properly defined [148] as

$$\text{lcm}\left(\frac{a}{b}, \frac{c}{d}\right) = \frac{\text{lcm}(a, c)}{\text{gcf}(b, d)}, \tag{4.3}$$

where gcf is the *greatest common factor*.

Given the periodicity of the allocation patterns, a new allocation A_N should start within a time interval T_p^N since the beginning of the BI. Moreover, a necessary requirement for admission is that in an interval of duration $T_P^{1,\dots,N}$, no block b_h^N overlaps with any block b_k^n, $n \in \{1, , \dots, N-1\}$, $\forall h, k \geq 0$.

The remainder of this section is structured as follows: in Sec. 4.5.1 we will illustrate an algorithm to efficiently check whether a new allocation is compatible with a pre-existing schedule, in Sec. 4.5.2 we will present a simple scheduling algorithm, and finally in Sec. 4.5.3 we will describe in detail a more complex algorithm that aims at minimizing the rejection of new allocations under the *strict periodicity* assumption.

Algorithm 4.1 Feasibility check under strong periodicity conditions (see Figs. 4.8 and 4.9).

```
function FeasibilityCheck($\{A_1, \ldots, A_{N-1}\}$ (fixed), $A_N$ (new allocation), $[t_{\min}, t_{\max}]$)
    Compute $T_p^{1,\ldots,N}$
    $t_{0,\text{start}}^N \leftarrow t_{\min}$                                                ▷ Fig. 4.8a
    while $t_{0,\text{end}}^N < t_{\max}$
        Check for collisions in $\left[t_{\min}, t_{\min} + T_p^{1,\ldots,N}\right)$
        if no collisions
            $t_{\text{feas}} \leftarrow t_{0,\text{start}}^N$ return $A_N$ is feasible with starting time $t_{\text{feas}}$    ▷ Fig. 4.9
        else
            Allocation block $h \in A_N$ collides with allocation block $k \in A_i$, for some $i \in 1, \ldots, N-1$
            $\Delta t \leftarrow t_{k,\text{end}}^i - t_{h,\text{start}}^N$
            $t_{0,\text{start}}^N \leftarrow t_{0,\text{start}}^N + \Delta t$
    return $A_N$ is not a feasible allocation                                    ▷ Fig. 4.8
```

4.5.1 Feasibility Check Algorithm

This feasibility check can be performed as described in Alg. 4.1, whose arguments consist of the existing allocations $A_1, \ldots, A_{N-1}$, the new request A_N and a search interval $[t_{\min}, t_{\max}]$. For reasons that will be clear later, we assume that the existing allocations cannot be changed, while for A_N only $t_{0,\text{start}}^N$ can be modified, keeping the period T_p^N and the block duration T_{blk}^N fixed. Based on these input values, the aim of the procedure is to find the earliest feasible starting time t_{feas}^N such that a block of duration T_{blk}^N fits in the search interval.

To do so, starting from $t_{\min}$, the algorithm progressively shifts the starting time by an interval Δt (described in Alg. 4.1) until either all feasibility conditions are met, or $t_{0,\text{end}}^N > t_{\max}$, in which case the allocation A_N with block duration T_{blk}^N is rejected.

A trivial example involves A_1, i.e., the first received allocation request from a STA. In this case, Alg. 4.1 is invoked with $t_{\min}$ set to the start time of the first BI following the reception of the request, and $t_{\max} = T_p^1$ to guarantee the periodicity. Since no previous allocated SPs exist, A_1 is immediately accepted with $t_{\text{feas}} = t_{\min}$. It is important to highlight that, however, by choosing specific combinations of input parameters, Alg. 4.1 can be used also by more advanced scheduling schemes, as explained later.

Given any feasible starting time t_{feas}, it is useful to compute the rightmost *boundary* of the allocation, i.e., the largest interval $[t_{\text{feas}}, t_{\text{lim}}]$ that would still make $b_0^N \in [t_{\text{feas}}, t_{\text{lim}}]$ and, in turn, A_N feasible, even for larger values $t_{0,\text{start}}^N$ and T_{blk}^N. This boundary can be computed by finding the minimum distance between each $b_h \in A_N$ and each $b_k \in A_n$, $n \neq N$. The final results will be the minimum measured distance. A graphical illustration of how the algorithm behaves when the new request is infeasible is shown in Fig. 4.8, while a new feasible request is shown in Fig. 4.9.

Following this definition and the above numerical example, the first allocation request to be generated, i.e., A_1, will find itself in the optimal condition where $t_{\text{feas}} = t_{\min}$ and $t_{\text{lim}} = t_{\max}$.

In general, multiple feasible intervals may exist. To find an exhaustive list, we can iterate Alg. 4.1 with $t_{0,\text{start}}^N$ initialized to the start time of the BI, and progressively updated at each iteration with the value of t_{lim} found in the previous execution. This procedure continues until the shift of $t_{0,\text{start}}^N$ leads to an infeasible allocation. We define the list of feasible intervals (which depend on T_{blk}^{N}) as $\mathcal{I}_N = \left\{I_1^N, \ldots, I_M^N\right\}$, where $I_m^N = \left[t_{m,\text{feas}}^N, t_{m,\text{lim}}^N\right]$, $m = 1, \ldots, M$ (see Fig. 4.9c). Hence, the new allocation A_N can be fitted in any of these intervals, considering all previous allocations. A good scheduling algorithm should then assess which interval yields the best overall performance, possibly trying to optimize a target KPI.

4.5.2 Simple Scheduler

The first scheduler that we propose assumes that the block duration and periodicity of already accepted traffic streams cannot be varied. Then, a new request A_N with a block duration of $T_{\text{blk}}^N \in [T_{\text{min}}^N, T_{\text{max}}^N]$ can be accepted only if there exists a feasible interval in $\mathcal{I}_N$ with a duration of at least T_{min}^N. Therefore, the maximum amount of available resources that can be allocated to A_N is determined by the longest feasible interval, or by T_{max}^N, whichever is smaller; $t_{0,\text{start}}^N$ and T_{blk}^N need to be set accordingly. We can already notice that, using this simple first-come-first-served approach, the latest requests are highly disadvantaged if the first ones require large fractions of the time resources. In the long term, as we will see in Sec. 4.5.4, this could lead not only to poor fairness performance, but also to a very low admission rate.

4.5.3 Max-Min Fair Scheduler

A more flexible approach consists in dynamically adapting the duration of the allocated intervals within the admissible range, $T_{\text{blk}}^n \in [T_{\text{min}}^n, T_{\text{max}}^n]$, $\forall n = 1, \ldots, N$, in order to distribute time resources among all traffic streams in a fairer manner.

Consider the following parameterized block duration:

$$T_{\text{blk}}^n(r) = T_{\text{min}}^n + r_n(T_{\text{max}}^n - T_{\text{min}}^n), \quad r_n \in [0, 1] \ . \tag{4.4}$$

We consider a scheduler to be fair if $r_{\text{min}} = \min_n\{r_n\}$ cannot be increased without breaking the limits imposed by some allocation under the *strict periodicity* constraint (see Sec. 4.5). The scheduling algorithm, then, should assign the largest possible SP to each allocation, while respecting all the constraints.*

To fit a new traffic stream, the pre-existing allocations will thus have to either maintain or reduce their block duration, depending on whether and how the new allocation collides with them. This will lead to a lower rejection rate with respect to the *Simple Scheduler* (Sec. 4.5.2), and more fairness among requests distributed in time.

The proposed algorithm is here presented in two parts: the first part describes how the allocation scheme works (Sec. 4.5.3), while the second part describes the fairness paradigm (Sec. 4.5.3).

Allocation Algorithm

Differently from the simple scheduler, this scheduler can change the block duration within the range imposed by the requesting STA, i.e., $T_{\text{blk}}^n \in [T_{\text{min}}^n, T_{\text{max}}^n]$. To reduce the rejection rate, we

*Note that if $T_{\text{min}}^n = T_{\text{max}}^n$, r_n has no meaning. For simplicity, this case has not been included in this study.

Algorithm 4.2 Max-min fair scheduling.

```
function MaxMinFairScheduling($A_1, \ldots, A_N$, $T_p^{1,\ldots,N}$)
    Compute $\mathcal{I}_N$ considering $T^n_{\text{blk}} = T^n_{\min} \forall n = 1, \ldots, N$
    for all $I^N_m = [t_{\text{feas}}, t_{\text{lim}}]^N_m \in \mathcal{I}_N$
        $t^N_{0,\text{start}} \leftarrow t_{\text{feas}}$
        Set $r_N$ such that $T^N_{\text{blk}} = \min \left\{T^N_{\max}, t_{\text{lim}} - t_{\text{feas}}\right\}$        ▷ (4.4)
        for all $A_n$, $n = 1, \ldots, N-1$
            for all (block $k \in A_n$) $\in T_p^{1,\ldots,N}$
                if block $k$ collides with $A_N$
                    Update $r^*_n$, $r^*_N$, $t^{N*}_{0,\text{start}}$        ▷ Sec. 4.5.3
                    if $r^*_n < r_n$
                        Add/update $A_n$ to a list $\mathcal{C}$ of colliding allocations
                        Memorize $r_{n,\text{prev}} \leftarrow r_n$
                    Update $r_n$, $r_N$, $t^N_{0,\text{start}}$
        for all $A_n \in \mathcal{C}$
            Compute $\Delta t = t_{\text{lim}} - t^n_{0,\text{start}}$ for $A_n$ given $A_N$        ▷ see Sec. 4.5
            $T^n_{\text{blk}} \leftarrow \min\{T^n_{\text{blk}}(r_{n,\text{prev}}), \Delta t\}$ ▷ Try to improve the allocation duration if $A_N$ has been further reduced
        Compute allocation score $s_m \leftarrow \min_{n=1,\ldots,N} r_n$
    return The configuration which maximizes the allocation score $\{s_m\}$
```

check the feasibility of a new allocation A_N (Sec. 4.5.1) by assuming that all existing allocations are shrunk to their minimum, i.e., $T^n_{\text{blk}} = T^n_{\min}$ for $n = 1, \ldots, N$. If A_N is infeasible even under these conditions, then the allocation cannot be granted without disattending the requests of some previously accepted flow. Therefore, A_N is rejected. Conversely, if A_N is feasible, it gets accepted, and in a later step the algorithm will try to increase the resource utilization of all allocations fairly.

From now on, we use the superscript * to indicate the parameter values at the end of the execution of the algorithm. We recall that, based on the strict periodicity assumption, the starting times of the already allocated blocks cannot change.

Note that, given a set of feasible allocations, reducing any r_n (and, in turn, T^n_{blk}) still yields a valid configuration. Similarly, a valid configuration for A_N with $t^N_{0,\text{start}}$ and $r_N \geq 0$ will remain valid if $t^{N*}_{0,\text{start}} \geq t^N_{0,\text{start}}$ and $t^{N*}_{\text{end}} = t^{N*}_{0,\text{start}} + T^N_{\text{blk}}(r^*_N) \leq t^N_{\text{end}}$. We thus consider the following constraints:

$$r^*_n \leq r_n, \forall n \leq N; \tag{4.5a}$$

$$t^{N*}_{0,\text{start}} \geq t^N_{0,\text{start}}; \tag{4.5b}$$

$$t^{N*}_{\text{end}} \leq t^N_{\text{lim}}. \tag{4.5c}$$

The algorithm starts by considering the first feasible interval I^N_1, which ensures a valid configuration when $r_n = 0$, $\forall n \leq N$. The new request is temporarily accepted with $t^N_{0,\text{start}} = t^N_{1,\text{feas}}$ and maximum possible r_N, such that $T^N_{\text{blk}}(r_N) = \min\left\{T^N_{\max}, t_{\text{lim}} - t_{\text{feas}}\right\}$.

Then, the algorithm tries to re-balance the resource allocation by increasing all $\{r_n, \forall n \leq N\}$ to their previous values. Given that feasible intervals $\mathcal{I}_N$ were computed considering all allocations with minimum duration, though, setting $t^N_{0,\text{start}} = t^N_{1,\text{feas}}$ may (or may not) create a collision with a generic A_i when setting $r_i \geq 0$ back to its previous value.

On the other hand, thanks to the information given by t_{lim}, we can always choose r_N such

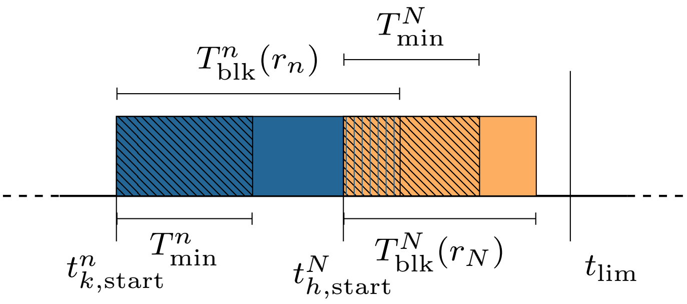


Fig. 4.10: Representation of a collision between A_n and A_N.

that the new allocation does not collide with a previous one, on the right.

Collisions can be found iteratively over each block of each previous allocation in a joint period.

If for a certain block $b^n_k \in A_n$ and a block $b^N_h \in A_N$ we have

$$t^n_{k,\text{start}} + T^n_{\text{blk}}(r_n) \geq t^N_{h,\text{start}}, \tag{4.6}$$

then the two allocations are in conflict, as shown in Fig. 4.10. In this case, $t^{N*}_{0,\text{start}}$, r^*_N, and r^*_n have to be updated following the constraints in (4.5), as described in Sec. 4.5.3.

The constraints (4.5), the existence of a non-empty set of feasible intervals, and the iterative nature of the problem ensure that the algorithm will stop in a finite time with a valid configuration. Since each feasible interval $I^N_m \in \mathcal{I}_N$ has one locally fairest configuration, the exhaustive search described in Alg. 4.2 is able to find the globally fairest configuration.

Optimally fair allocation

In this section, we will discuss how fairness can be achieved given a pair of colliding allocations A_n, $n \in \{1, \dots, N-1\}$, and A_N. In Sec. 4.5.3 we explained how such a collision can be found, e.g., between blocks b^n_k and b^N_h. For the sake of clarity, in this section we will drop the notation for the specific colliding blocks.

In order to fully exploit the available resources, looking at Fig. 4.10, we force $t^N_{0,\text{start}} = t^n_{\text{end}}$ and $t^N_{\text{end}} \leq t_{\text{lim}}$. In this way we make A_N start right after A_n, still respecting the limits imposed by t_{lim}.

While possibly not being optimal, this is still a sensible choice for a greedy approach that tries to maximize the fairness of the current configuration. By doing so and imposing $r_n = r_N = r^*$, what we call the *fairness equation*, and by noting that $r^* \leq 1$ should hold, we have that

$$r^* = \min\left\{1, \frac{t_{\text{lim}} - t^n_{0,\text{start}} - T^n_{\text{min}} - T^N_{\text{min}}}{(T^n_{\text{max}} - T^n_{\text{min}}) + (T^N_{\text{max}} - T^N_{\text{min}})}\right\}. \tag{4.7}$$

We mention again that the edge case where $T_{\text{min}} = T_{\text{max}}$ has not been included in this preliminary study. We call r^* the *fair allocation ratio*, and note that if $r^* < 1$, it must be that $t^N_{\text{end}} = t_{\text{lim}}$, whereas if $r^* = 1$ in general $t^N_{\text{end}} \leq t_{\text{lim}}$ by construction.

Depending on the initial conditions of the problem, there is a number of different cases which have to be properly managed in order to obtain a fair distribution of resources.

First of all, if $r_n \leq r^*$, following (4.5a), it means that previous adjustments do not make it possible for A_n to obtain more resources while still ensuring a valid configuration, and thus $r^*_n = r_n$. Furthermore, since we assume that a collision happens between A_n and A_N with this configuration, A_N has to be delayed by setting $t^{N*}_{0,\text{start}} = t^n_{0,\text{start}} + T^n_{\text{blk}}(r^*_n) > t^N_{0,\text{start}}$.

In case also $r_N \leq r^*$, allocation A_N cannot be extended either. Since both allocations have $r_n, r_N \leq r^*$, they will both surely fit in the feasible interval. If, instead, $r_N > r^*$, then A_N can obtain $r^*_N \geq r^*_n$, i.e., $T^{N*}_{\text{blk}} = \min\left\{T^N_{\text{blk}}(r_N), t_{\text{lim}} - t^{N*}_{0,\text{start}}\right\}$.

On the other hand, if $1 \geq r_n > r^*$, the block duration must be reduced so that $r^*_n = r^*$. Then, if also $r_N > r^*$, both allocations must be trimmed and are fairly allocated, i.e., $r^*_n = r^*_N = r^* < 1$. It follows from (4.7) that $t^{N*}_{\text{end}} = t_{\text{lim}}$ and $t^{N*}_{0,\text{start}} > t^N_{0,\text{start}}$.

Finally, if $r_N \leq r^* < r_n \leq 1$, and therefore $r_N < 1$, (4.7) implies that $t^N_{\text{end}} = t_{\text{lim}}$. Since A_N cannot be extended without possibly reducing the allocation ratio of other allocations, $t^{N*}_{0,\text{start}} = t^N_{0,\text{start}}$ and $r^*_N = r_N$. Given that, by assumption, $t^n_{\text{end}} > t^N_{0,\text{start}}$, the duration of A_n needs to be reduced so that $T^{n*}_{\text{blk}} = t^{N*}_{0,\text{start}} - t^n_{0,\text{start}} < T^n_{\text{blk}}$.

4.5.4 Scheduling Performance

In this section, we evaluate the algorithms described in Sec. 4.5. The proposed schedulers have been implemented in Python, only focusing on their capabilities of allocating communication resources to the different traffic streams. The simulations do not consider full stack details, which will be investigated in future works, and their aim is thus to highlight the fundamental characteristics of each algorithm.

We shape the offered traffic based on three parameters, namely: the *average allocation request*

$$T_{\text{avg}} = \frac{T_{\min} + T_{\max}}{2}; \tag{4.8}$$

the *interval ratio*

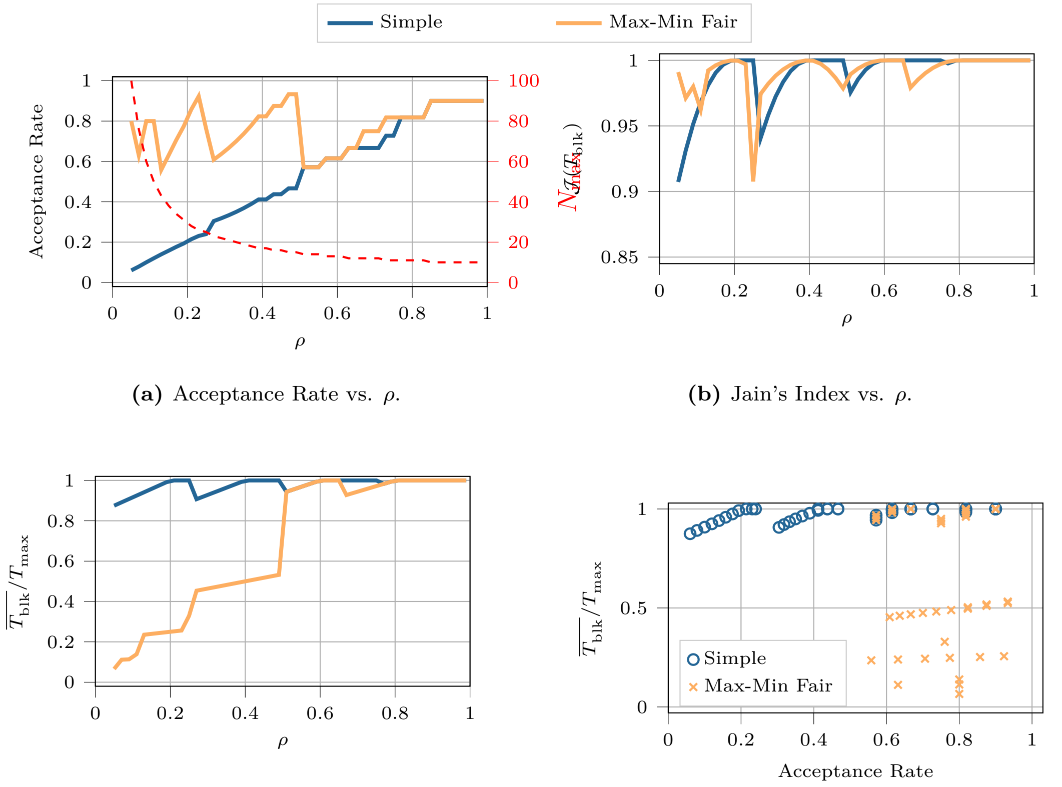
$$\rho = \frac{T_{\min}}{T_{\max}} \in (0, 1); \tag{4.9}$$

and the *load factor*

$$\lambda = \frac{T_{\text{avg}}}{T_p}. \tag{4.10}$$

Note that, for a given average allocation request T_{avg}, a low interval ratio ρ corresponds to very flexible allocations, while $\rho = 1$ corresponds to rigid allocations where $T_{\min} = T_{\max}$, although this last case is not considered in this thesis. Furthermore, for a fixed T_{avg}, both $T_{\max}$ and $T_{\min}$ become a function of ρ, e.g., $T_{\max}(\rho)$.

The proposed algorithms are compared in two different simulation scenarios:



(a) Acceptance Rate vs. ρ.

(b) Jain's Index vs. ρ.

(c) Normalized average T_{blk} vs. ρ.

(d) Normalized average T_{blk} vs. Acceptance Rate.

Fig. 4.11: Results for Scenario 1.

- *Scenario 1*: all traffic streams are homogeneous, i.e., all requests have the same parameters. Specifically, we consider the case with periodicity $T_p = \frac{T_{\text{BI}}}{3}$, load factor $\lambda = 0.1$, and $\rho \in (0.05, 1)$. The impact of different periodicities and load factors is also discussed.
- *Scenario 2*: multiple non-homogeneous applications coexist in the same network, thus generating traffic streams with different characteristics. We analyze a scenario where traffic streams can be of class C_1 or C_2, with periodicity $T_p^{C_1} = \frac{T_{\text{BI}}}{3}$ and $T_p^{C_2} = \frac{T_{\text{BI}}}{5}$, respectively. Both classes have load factor $\lambda = 0.1$ and interval ratio $\rho = 0.1$.

Based on their design criteria, we expect the two algorithms to differ mainly with respect to three Key Performance Indicators (KPIs), namely the acceptance rate of new requests, the fairness among accepted allocations, and the average scheduled block duration. The performance of the algorithms in *Scenario 1* is shown in Fig. 4.11 (note that, given the discrete nature of the problem and the deterministic behavior of the proposed algorithms, the plots cannot be smoothed by running multiple repetitions).

The first metric is the acceptance rate (Fig. 4.11a), defined as the ratio between the number of accepted allocation requests and the maximum number of acceptable requests. To compute this achievable upper bound, since all allocations share the same parameters we ignore the strict periodicity assumption and calculate how many allocations with minimum duration $T_{\text{blk}} = T_{\text{min}}$ can fit in a period T_p, which is equal to $N_{\text{max}}(\rho) = \left\lfloor \frac{T_p}{T_{\text{min}}(\rho)} \right\rfloor$ and shown as a red, dashed line in the figure. The acceptance rates can thus be normalized in the interval $[0, 1]$, where 1 means that the scheduler reaches the peak acceptance rate.

As expected, the *simple scheduler* suffers from a lower acceptance rate than the *max-min fair* one, even though, starting from $\rho = 0.5$, the two algorithms tend to behave similarly. In fact, more rigid allocations do not give enough flexibility to the *max-min fair scheduler* to perform its optimization, thus yielding similar performance to the much simpler *simple scheduler*.

The second metric is *Jain's Fairness Index* [149] (Fig. 4.11b), defined as:

$$\mathcal{J} = \frac{\left(\sum_n x_n\right)^2}{n \cdot \sum_n (x_n)^2}, \tag{4.11}$$

where only accepted allocations are counted and x_n can be either the block duration T_{blk}^n or the block duration ratio r_n. If the $\{x_n\}$ are all equal, then $\mathcal{J}(x) = 1$. On the other hand, the more dissimilar the values of $\{x_n\}$, the closer the metric to its minimum $\mathcal{J}(x) = \frac{1}{N}$.

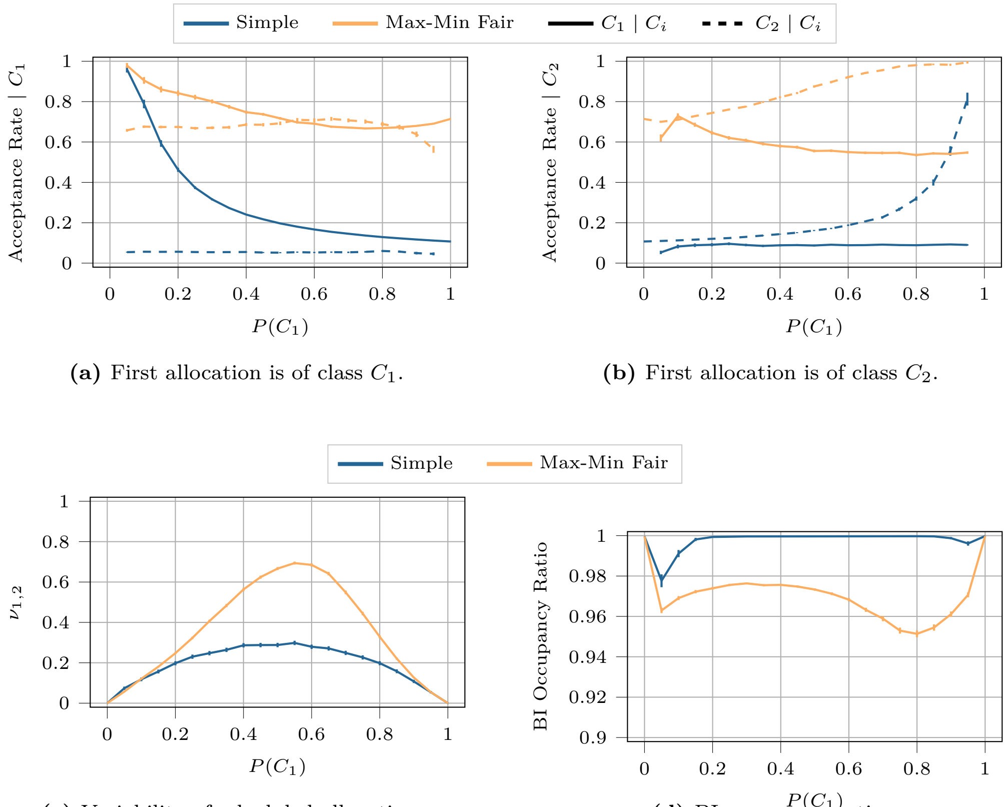
Based on the results plotted in Fig. 4.11b, both algorithms behave very fairly with respect to the accepted allocations. Note that, the larger ρ, the higher the rigidity of the resource requests and, in turn, the larger the fairness among accepted flows, considering that, in this scenario, they are homogeneous.

The third metric, shown in Fig. 4.11c, offers a different perspective considering the average normalized block duration $\overline{T_{\text{blk}}}/T_{\text{max}}(\rho)$. As expected, the *simple scheduler* shows an oscillating trend, due to the rigid and discrete allocation policy. In fact, if the portion of DTI left by the previous allocations is less than $T_{\text{min}}(\rho)$, no additional requests can be accepted (we recall that, in this scenario, all requests have the same parameters). On the other hand, the *max-min fair scheduler* will try to reduce all allocations down to their minimum duration in order to avoid rejecting new ones, granting more accepted allocations at the cost of an overall lower block duration.

Finally, the two most discriminating metrics, namely the average normalized block duration and the acceptance rate, are plotted against each other in Fig. 4.11d. In general, the *simple scheduler* tends to favor a higher average block duration for a lower acceptance rate, while the *max-min fair scheduler* tends to favor the acceptance rate at the cost of a lower average block duration, as expected. Both algorithms are able to ensure high fairness to the accepted allocations, generally well above 0.85.

Similar behaviors were also observed for load factors $\lambda \in \{0.025, 0.4\}$, not shown here. As expected, higher loads tend to have more pronounced variability in both the average block duration and the fairness granted to the accepted allocations. Interestingly, regardless of the load factor, for values of the interval ratio larger than $\rho \approx 0.5$, the two algorithms tend to have very similar performance due to the more rigid allocation requests that do not allow the *max-min fair scheduler* to exploit its agility.

Scenario 2 allows us to analyze the impact of allocations with different periodicities on the overall network performance, as a function of the probability $P(C_1)$ that a request C_1 is offered



(a) First allocation is of class C_1.

(b) First allocation is of class C_2.

(c) Variability of scheduled allocations.

(d) BI occupancy ratio.

Fig. 4.12: Results for Scenario 2.

to the system.

Since allocations with different periods coexist, it is mandatory to decide how many allocations should be offered to the schedulers, as this will affect how the acceptance rate is normalized. To this end, we define the *minimum occupancy* of category C_i as $O^i_{\text{min}} = T^i_{\text{min}}/T^i_p$. Allocations are offered to the schedulers as long as the cumulative minimum occupancy does not exceed the value 1.

In Figs. 4.12a and 4.12b we show the biasing effect of the first accepted allocation on the proposed schedulers. Clearly, the *simple scheduler* suffers from a strong and symmetric effect, meaning that once the first allocation is scheduled with maximum duration, it will be harder for subsequent allocations with a different period to fit the constrained BI, making the scheduler favor allocations with the same period. On the other hand, it is significantly harder to interpret the behavior of the more complex *max-min fair scheduler*. From further results, not shown here due to lack of space, it is possible to notice that allocations with a lower average BI occupancy are favored, with a slight preference towards those with lower values of T_p and T_{min}. As also shown here, in fact, allocations with lower values of T_p, such as C_2 with respect to C_1, tend to fragment the BI more, making it harder to then fit allocations with different periodicity and higher T_{min}.

To further confirm this biasing behavior, we show the variability ν among the scheduled allocations, defined as

$$\nu_{1,2} = \frac{\min\{|C_1|, |C_2|\}}{\max\{|C_1|, |C_2|\}}, \tag{4.12}$$

where $|C_i|$ represents the number of accepted allocations of category C_i. This metric takes values in $[0, 1]$, where a value of 0 means that only allocations of a single type have been accepted, while a value of 1 means that the same number of allocations of both categories have been accepted. Results are shown in Fig. 4.12c, which confirms that the *simple scheduler* favors a more homogeneous BI allocation, while the *max-min fair scheduler* shows once again more flexibility, being able to fairly accommodate requests from different classes, as shown by the higher variability of the accepted allocations.

Finally, we studied how efficiently the two algorithms are able to use the radio resources by measuring the BI occupancy ratio, i.e., the ratio between scheduled and unscheduled air time. It can be noticed that, while the *simple scheduler* accepts fewer requests, it is able to use almost all available resources. This is due to the fact that the scheduler tends to accept homogeneous allocations, allowing them to be packed more efficiently in the BI. On the other hand, the *max-min fair scheduler* successfully fits multiple allocations of both types, but the constraints on the periodicity and the minimum block duration T_{min} prevent it from fully utilizing the whole BI when a mixture of the two types of sources is presented. Nonetheless, it ensures very high occupancy ratios, always above 95% for this example.

4.6 Conclusions

In this chapter, we briefly described the main characteristics of the WiGig standards, mainly focusing on the MAC layer and especially on the newly introduced scheduling mechanisms by IEEE 802.11ad, allowing different types of traffic to coexist and potentially improving the performance of QoS-sensitive applications [147]. As shown in Sec. 4.2, some research has already been done in this direction but lacks a common and realistic testing ground, making it unclear whether the assumptions may hold.

We then presented an open-source scheduling framework for WiGig based on the ns-3 implementation of the IEEE 802.11ad standard [145] [150]. We implemented two schedulers, one based on contention-based channel access, the other based on periodic SP allocations, and compared their performance on three different scenarios. Results show that SP scheduling is able to surpass contention-based channel access and yield the best performance only when cross-layer information between the MAC and APP layers is exchanged. Moreover, adding even small amounts of randomness to the periodic application results in great performance degradation for periodic SP scheduling, making contention-based access the preferred option in most cases.

Lastly, strong of the results obtained from the first simple simulation campaigns, we presented a mathematical framework for periodic scheduling in WiGig-compatible devices, attempting to design more solid and better performing schedulers. We proposed two heuristic algorithms, *simple* and *max-min fair schedulers*, and accurately described their inner workings [151]. Finally, we assessed their performance in two different scenarios, showing that the *max-min fair scheduler* tends to trade the portion of resources allocated to each single station for a much higher acceptance rate, contrary to the *simple scheduler*'s behavior, while both schedulers obtained a high value of Jain's fairness index for the accepted allocations.

Even working in a simplified setting without considering further sources of complexity from other parts of the communication stack, it was possible to notice that both the design and the evaluation of WiGig-specific scheduling algorithms for periodic sources is non-trivial and can show unexpected results. In fact, while the formalization of the problem is rather straightforward, scheduling algorithms often have to deal with many hard-to-predict edge cases, which greatly increase the difficulty of designing a general algorithm.

Work is currently being done to implement the schedulers derived from our mathematical framework of Sec. 4.5 into our ns-3 simulation framework of Sec. 4.4, allowing to better evaluate the E2E performance of the designed schedulers, especially in non-ideal conditions and non-ideal CBR periodic traffic.

Having created some tools that can be easily worked upon, we propose some research directions:

1. From the STA point of view, a DMG TSPEC has to be formulated in order to send the relevant information about the characteristics of the traffic stream to the PCP/AP. The definition of this information element is important to obtain the required resources, and it should thus be studied and optimized depending on the QoS requested by the application. Specifically, the application can give information about the details of the traffic flow (periodicity and block size, among others), although the final block duration will also depend on the channel state and the uncertainty about the application traffic flow itself. Two main contributions should be better studied:

    (a) Channel quality prediction is essential to dynamically adapt the scheduled resources among users. Given the significant differences in channel dynamics of IEEE 802.11ad with respect to sub-6 GHz Wi-Fi and to the peculiarity of packet exchanges during SPs, new MCS adaptation mechanisms could be proposed to maximize the communication efficiency, while improving channel prediction.

    (b) Most relevant applications that could be supported by WiGig are almost CBR, although not quite. Both periodicity and block sizes are often not constant, requiring to find a good balance between allocating extra resources (which may be unused) and potentially reducing the final user experience due to insufficient resources.

2. Understanding how the current state of the art performs in a realistic simulator will allow understanding the strong and weak points of the proposals in realistic settings. With a flexible simulation platform like the one we propose, algorithms from the state of the art can be implemented and finally be fairly compared against each other, giving further insights and possibly improving them with respect to their weak points.

3. Our studies can be easily transferred to the future IEEE 802.11ay standard, which will add further complexity on top of the already existing one, by introducing channel bonding and MU-MIMO. Even more complex schedulers will then have to be designed, but starting from the solid ground of the proposed work further improvements will be possible.

4. Whenever heuristic algorithms are not sufficient to obtain satisfying results, or when the complexity of the problem is excessive for a simple algorithm to be conceived, ML techniques, and specifically RL algorithms could come in handy. Scheduling algorithms can indeed benefit from this modern approach, possibly leading to better performance.

5

Extended Reality Applications

5.1 Introduction

After several years of innovations, the technology is finally ready for applications such as VR, AR and Maximum Rate (MR) to go mainstream (in the following we will use the term XR as a general expression to consider all these distinct interaction modes). According to some estimates, by 2025 there will be over 200 million people using XR for immersive gaming experience and 95 million enjoying live events in this novel way [152]. This immediately translates in increasing sales of devices and headsets dedicated to experience this new type of contents, with an estimated shipment of these devices in the order of tens of millions in the coming decade, generating billions in revenue for all the fields in which this technology will be deployed [153, 154, 155].

Although it all started from the entertainment and video gaming arenas, where players could immerse in a virtual 3D world, now we can see XR applied in various fields, such as building or landscape design, real estate, marketing, and healthcare, opening up the possibility of learning new concepts and training employees for difficult situations in a completely different way [156, 153, 157, 158, 159, 160]. Automotive companies, for example, are using VR to cut the time that leads to the physical model of a new product from weeks to days [155]. Regarding the general retail market, instead, VR can give customers realistic experiences with products, allowing them to easily consider different options and configurations [153] and thus increasing sales and decreasing product returns.

The peculiarity of this new class of contents, besides the wide range of use cases, is that the end user does not passively receive the information, but acts on it, possibly affecting the future behavior of the application itself. Hence, the traffic flow to and from the content provider is highly dependent on the interaction with the virtual environment in which the user is immersed.

In this paper, we will focus on examples related to the video gaming world (even though equivalent conclusions can be drawn for different XR applications), where the user interacts with the application using a keyboard or joypad, and the results of such actions are immediately seen on the PC or TV screen. Through Head Mounted Devices (HMDs), when playing with videogames supporting VR, users can also react by moving their heads, causing the application to stream distinct portions of the environment depending on where the head is pointing [161].

Even though traditionally gaming software ran on devices which needed to respect several hardware constraints to generate high-quality images, now the paradigm is shifting towards a cloud approach [162, 163]. This can be extended to all other use cases besides gaming, and for this reason, we can refer to this new paradigm as *Cloud XR*. By moving the computing and graphical processing units into the cloud, less powerful devices can be used to fully exploit this new technology. This would benefit not only in terms of the actual cost of an HMD (which still plays a huge role in promoting the adoption by new users) but also in the final QoE. Having all the computing resources self-contained in the device would mean not only a higher weight and volume, but also concerns in terms of heat and battery life [154, 159].

This shift towards cloud infrastructures requires the optimization of current communication systems, to fully support distributed XR services. To this end, we need accurate models of the applications that generate these data flows and, to the best of our knowledge, no previous work has addressed this problem so far.

In this work we try to fill this gap by proposing a traffic model that emulates an XR application, while also sketching a roadmap to guide researchers in the development of more precise models, using ours as a baseline.

To further understand what are the steps that most influence the XR performance, it is useful to describe a common end-to-end XR architecture [152, 159]. First, we can start from the collection and processing of sensory and tracking information, delegated to an ad hoc device. Then, this information is sent to an XR server to compose the viewport, i.e., what is actually shown to the user. This process includes 2D/3D media encoding and the generation of additional metadata (including the scene description). The device's presentation engine at the client side, after receiving and decoding the information stream, generates the images to display. These images are derived from the decoded signals, the rendering metadata, and other information, if applicable. Finally, video and audio tracks associated with the current pose are generated by synchronizing and spatially aligning the rendered media. These steps need to be accomplished with minimal delay to guarantee adequate QoE.

In fact, the *motion-to-photon* latency, i.e., the time from an action (e.g., a head movement) to the update of what is shown on the display, must be below 20 ms to avoid the so called *cybersickness*, associated to disorientation and dizziness [164, 152, 165, 166, 167, 168]. Following this physiological constraint, several industry players pose the network requirements, in terms of latency, in the range of 5–10 ms [164, 169, 152, 154, 159]. Also in terms of the gaming QoE, it has been demonstrated that for first-person shooters, racing games, and team soccer matches, application latency directly impacts the results of competitive e-sports and, if not properly addressed, would lead to abandoning the game [162]. This translates into stringent constraints both in Downlink (DL) and in Uplink (UL), considering that not only the content must be streamed as soon as it is required, but also the user movements need to be promptly notified to the server. For this reason, the software that collects each movement input must consider all 6 Degrees of Freedom (6DoF), tracking both translations and rotations in the three perpendicular axes (based on the VR device, some may consider only rotational motion, i.e., 3 Degrees of Freedom (3DoF)). To take immersive mobile experience to the next level, many improvements will be required in head, body, and even gaze tracking [157].

It is also important to distinguish between processing latency, associated with computation and rendering, and network latency. Rendering complex gaming images can be quite demanding, and the delay introduced by these operations can be larger than that caused by network services, which further motivates the need to offload these functions to proper cloud infrastructures [162].

Besides delay-related issues, an additional problem consists in the bursty nature of the XR traffic, meaning that the throughput measured over short time windows could be much higher than its long-term average value [159], which can be the case for an application that periodically generates collections of packets to refresh the viewport. Another aspect impacting the throughput is that, in order for the technology to be as close as possible to human vision, we will need a higher spatial and temporal resolution of the content presented to the user than currently possible (i.e., 3D 360° 8196×4096 resolution at 90 Hz and beyond display refresh rate) [164, 157, 159].

The core technology that is expected to guarantee the satisfaction of all these requirements, by paving the way for an optimized distribution of processing capabilities, is 5G. Many players have already invested in 5G for the rising of XR, for operations at both sub-6 GHz and mmWave [158, 157, 170, 171].

Nonetheless, even though some efforts have also been devoted by standard bodies to the redaction of technical reports [164, 169], at the present time researchers are limited by the lack of precise traffic models representing the stream to/from an XR server. Having these models would allow the research community to design telecommunication solutions that could reduce the delay contribution related to the network, while also considering all the processing steps. For this reason, we propose a generative model for XR traffic sources, obtained from real application traces, and we also delineate a roadmap of the necessary steps to further improve it with additional features able to cope with the aforementioned problems, i.e., motion-to-photon latency, burstiness, capacity.

While in Sec. 5.2 we summarize the current state of the art in the XR arena, Sec. 5.3 is devoted to the description of the acquisition setup that we used to collect about 70 GB of data for a total of more than 4 hours of traced traffic time using different VR applications, both from the hardware and from the software point of view. We will also describe each of these applications and illustrate how we analyzed the dataset. The model obtained from this analysis will be presented in Sec. 5.4, and its end-to-end validation, along with some example use cases, are discussed in Sec. 5.6. Finally, in Sec. 5.7 we propose a roadmap to extend our baseline model with additional increasingly complex features, and Sec. 5.8 concludes the paper.

5.2 State of the Art

A seminal conceptual model that describes the human and technical elements creating the participatory environments of virtual reality systems was proposed in [172], dating back to 1994. This demonstrates that the interest in the definition of common models for the study of this framework started even before the technology was ready, or even invented.

Despite the research interest in this field, to the best of our knowledge little work has been done on the creation of generative traffic models in XR contexts, while the focus was put on different aspects of the technology. In particular, a huge effort has been devoted to the creation and validation of practical systems that use immersive technology to interact with the world in different ways. An example can be found in [173], which describes a system for the interactive analysis of large datasets with time-dependent data, realized on a multi-processor parallel machine in order to guarantee a smooth user experience. Instead, in [174] the authors developed a proof-of-concept system, combining Oculus Rift HMD and the Phantom Premium 1.5 High Force haptic device with the goal of demonstrating the feasibility of combining HMD and haptics in one system. Also, XR solutions have been tested for purposes of architectural design [175] and for providing virtual performance instructions and feedback on users that want to play a real piano [176].

From a more technical perspective, a complete overview of the latest developments on immersive and 360° video streaming can be found in [161], where the author aims at providing a complete overview on four of the most important challenges in this field, namely: omnidirectional video coding and compression, subjective and objective QoE and the factors that can affect it, saliency measurement and FoV prediction, and adaptive streaming of immersive 360° videos. As stressed in [161], finding a proper way to measure the user's QoE may be difficult. This is especially important with respect to the design of telecommunication infrastructures able to optimize the experience of the user, and to guarantee constant and stable service quality.

For this reason, a lot of effort has been devoted to creating network solutions for the maximization of the quality of the delivered content. In [177], for example, the authors proposed a scheme for uplink delivery of tile-based VR video over cellular networks. In particular, they formulate resource allocation as a frequency- and time-dependent non-deterministic polynomial NP-hard problem, and propose three distinct algorithms to solve it. Instead, in [178] the authors consider a QoE-driven transmission of VR 360° contents in a multi-user massive MIMO wireless network. Specifically, in this scenario multiple users in the cell are requesting the same content, and the goal is to optimize the reception of such information through a stable scheme for the transmission of the viewport tiles. In this work, they also try to allocate the power in order to guarantee a consistent delivery rate for each stream.

The impact of latency on the overall experience of the user has been mentioned in Sec. 5.1, along with the importance of tracking the movements of the user in applications with strict delay requirements. The authors of [179] used a real VR head-tracking dataset to maximize the quality of the delivered video chunk under low-latency constraints. In that case, a deep recurrent neural network was designed for the prediction of the users' FoV (allowing to cluster those with overlapping FoV) while information on the future content and the users' locations was used as input of a proactive physical-layer multicast transmission scheme.

A key solution to the latency problem would be to rely on the capabilities of 5G and edge cloud, exploiting what has been referred to as *Cloud XR* in Sec. 5.1. Indeed, in [180] the authors demonstrated that 5G and edge cloud are necessary to sustain the requirements of applications such as VR gaming.

All these solutions, however, lack a model capable of generating data flows that can easily be associated with a real XR application. The approach of [177] consisted in using 240 frames of each 8K 360° uncompressed video sequence available from [181]. In that case the author applied the HVEC Kvazaar encoding procedure, setting the frame rate to 25 Frames Per Second (FPS) and the Group of Pictures (GoP) size to 8, and using a constant tiling scheme, ideal for the purpose of their work. Despite the high level of details implemented in such a model, the use of a trace-based flow is limiting per se, considering also the limited portion of the video that they selected. Having an offline encoding strategy is another drawback, that in our framework has been overcome by integrating the rendering server in the processing pipeline.

Also in [178], the simulation setup from the point of view of the VR architecture was defined in order to highlight the features of the algorithms proposed by the authors, and the nature of the traffic flow (e.g., average frame size, inter and intra-frames correlation, inter-frame interval, etc.) was not taken into account.

Cloud gaming [182] was identified as a closely related problem, where a remote server renders and streams a video to a client with limited computational resources, which only feeds basic information to the rendering servers, such as keys pressed and mouse movements. The main difference with the problem under analysis is given by the more restrictive QoS constraints of XR applications, mainly due to the limits imposed by motion sickness. Furthermore, in cloud gaming, client and server are often in different WLANs, making it harder to obtain reliable measurements of packet generation times. In fact, due to the specific constraints and requirements of XR applications, we expect the rendering server to be in a local network rather than being remotely accessed via the Internet.

Most works in the literature focus on network performance and limitations of cloud gaming [183], and we could find only two main contributions addressing traffic analysis and modeling. The authors of [184] provide a simple traffic analysis for three different games played on *OnLive*, a cloud gaming application that was shut down in 2015. The analysis focuses on packet-level statistics, such as packet size, inter-packet time, and bit rate. They measured the performance of the streaming service under speed-limited networks, showing an evident frame rate variability. In [185], the authors tried to model the traffic generated by two games, also played on the *OnLive* platform. In particular, they recognized that video frames were split into multiple fragments, and re-aggregated them before studying their statistics. A number of DL and UL data flows were recognized, and characterized in terms of *application packet data unit size* and *packet inter arrival time*. Unfortunately, correlation among successive video frames was not modeled and the analysis referred to a single game played with an average data rate of about 5 Mbps.

Regarding the problem of tracking the movements of the users, in [179] the authors fed the recurrent neural network with the 3DoF traces from [186], tracking the pose of 50 different users watching a catalog of 10 HD 360° videos from YouTube (60 seconds long, 4K resolution, 30 FPS, FoV of $100° \times 100°$). Having a generative model that creates such a dataset based on statistical studies on a collection of different traces would have greatly aided the training of the neural network used in [179]. Also, finding a dataset that represents well the problem that we want to solve is usually not feasible, and this may further limit the research outcomes.

As a consequence, our goal is to provide the community with a tool for the automatic generation of such traces. A preliminary version of this work was proposed in [187], and here we extend it with the acquisition of longer and more heterogeneous traces, that now include realistic interaction with several VR applications. This extension also allowed a more detailed and thorough validation of the model. Besides making both the model and the traces public, we also propose a possible roadmap for making the framework as complete and detailed as possible, highlighting the most important contributions that would benefit researchers aiming at the design of ad hoc network protocol optimizations for this new type of traffic sources.

5.3 VR Traffic: Acquisition and Analysis

In this section, we describe our basic traffic modeling work. Specifically, in Sec. 5.3.1 we describe our acquisition setup and the VR applications that we acquired, then in Sec. 5.3.2 we analyze the raw traffic traces, and the different streams composing them, both in terms of content and in terms of statistics.

5.3.1 Acquisition Setup

For the rendering server, we used a desktop PC equipped with an Intel Core i7 processor, 32 GB of RAM, and an NVIDIA GeForce RTX 2080 Ti graphics card. For the headset, instead, we used an iPhone XS enclosed in a VR cardboard, which allows a realistic interaction with the applications. The two nodes were connected via Wi-Fi to improve the user's freedom of movement, at the cost of a slightly less stable channel and of possible interference from other surrounding devices.

VR applications were thus run on the rendering server and streamed to the headset using the application *RiftCat 2.0* (on the server), and *VRidge* 2.7.7 (on the phone).* This setup allows the user to play VR games on the SteamVR platform for up to a maximum of 10 minutes continuously, enough to obtain traffic traces to be analyzed (note that this limit is given by the free version of VRidge, and is absent in the premium version). Many settings can be tuned in this application, such as the display resolution, the frame rate (either 30 or 60 FPS), the target data rate (i.e., the data rate the application will try to consistently stream to the client, which can be set from 1 to 50 Mbps), the video encoder (NVIDIA NVENC was used), and the video compression standard (H.264 was chosen), among other advanced settings.

*riftcat.com/vridge

As opposed to [187], we acquired traces while realistically interacting with available VR applications using mouse, keyboard, and head movements. Our setup only allowed us to interact with 3DoF, i.e., the user was seated and only head rotations were sensed by the headset. In any case, in order to increase the realism of the collected traces, the user was not required to limit the type of movements, but could freely interact with the application of interest. To simplify the analysis of the traffic stream, audio was not activated.

For this purpose, we selected three popular VR applications targeting different types of interactions. Specifically:

- *Minecraft*: an extremely popular game, with the *Vivecraft* plugin enabling room-scale or seated VR experiences. The user can explore the virtual environment by walking or swimming, and interact with the virtual world by cutting trees, digging holes, crafting tools, etc.

- *Virus Popper*: during this fast-paced educational game, many cartoony-looking viruses swarm a virtual room, and the user has to attack them with cleaning tools for survival.

- *Google Earth VR – Tour*: the VR version of Google Earth, allowing a user to explore the world with satellite imagery, 3D terrain of the entire globe, and 3D buildings in hundreds of cities around the world. The SteamVR application also enables tours, teleporting the user all around the world every few seconds.

- *Google Earth VR – Cities*: in this case, a more interactive experience is provided, allowing the user to fully explore cities or landmarks for as long as they want.

Please note that *Google Earth VR* was used in two different ways, thus allowing us to analyze two different versions of the same application.

To capture streamed packets, we ran Wireshark, a popular open-source packet analyzer, on the rendering server. The traffic analysis was performed at 30 and 60 FPS for target data rates of $\{10, 20, 30, 40, 50\}$ Mbps and for all 4 applications with a resolution of 1920×1080, for a total of over 70 GB of PCAP traces and 4 hours of analyzed VR traffic. Our dataset containing the processed VR traffic traces can be found within our software and can be easily reused, as later described in Sec. 5.6.

5.3.2 Traffic Analysis

As described in [187], we were able to partially reverse engineer both the DL and UL streams, and thanks to the help of *RiftCat*'s developers, we are now able to reliably process the raw traffic traces. We found that UDP sockets over IPv4 are used and both UL and DL streams contain several types of packets. Specifically, the UL stream contains packets such as synchronization, video frame reception information, and frequent small head-tracking information packets, whereas the DL stream contains synchronization, acknowledgment, and video frame packet bursts.

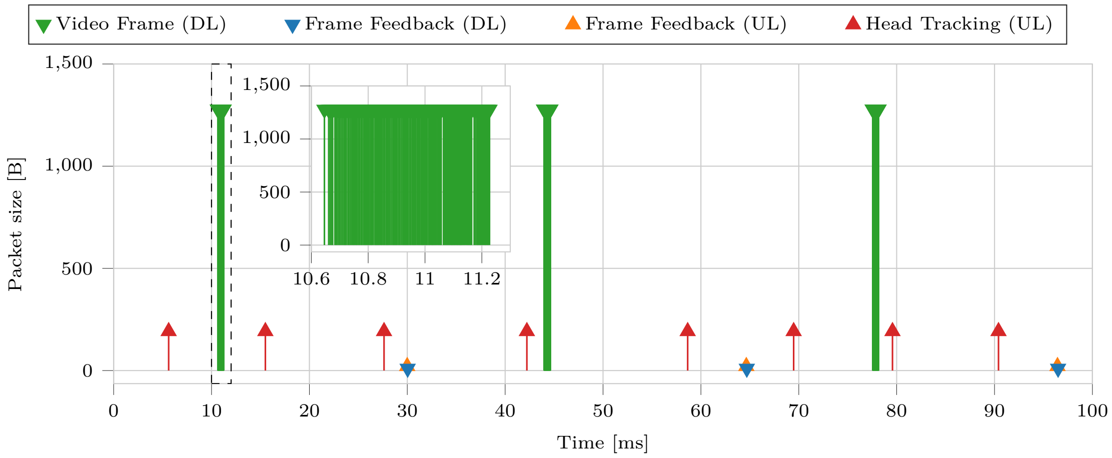


Fig. 5.1: Portion of traffic trace from *Virus Popper* (50 Mbps, 30 FPS). For this trace, 130–140 individual fragments make up a video frame burst.

To improve the stream quality, the RiftCat team developed a custom version of the ENet protocol*, a relatively thin, simple and robust network communication layer on top of UDP, which offers reliable, in-order packet delivery.

In Fig. 5.1 we show a visual representation of a slice of bidirectional VR streaming. The plot shows the main data streams in both DL and UL, giving an idea of how this transmission works.

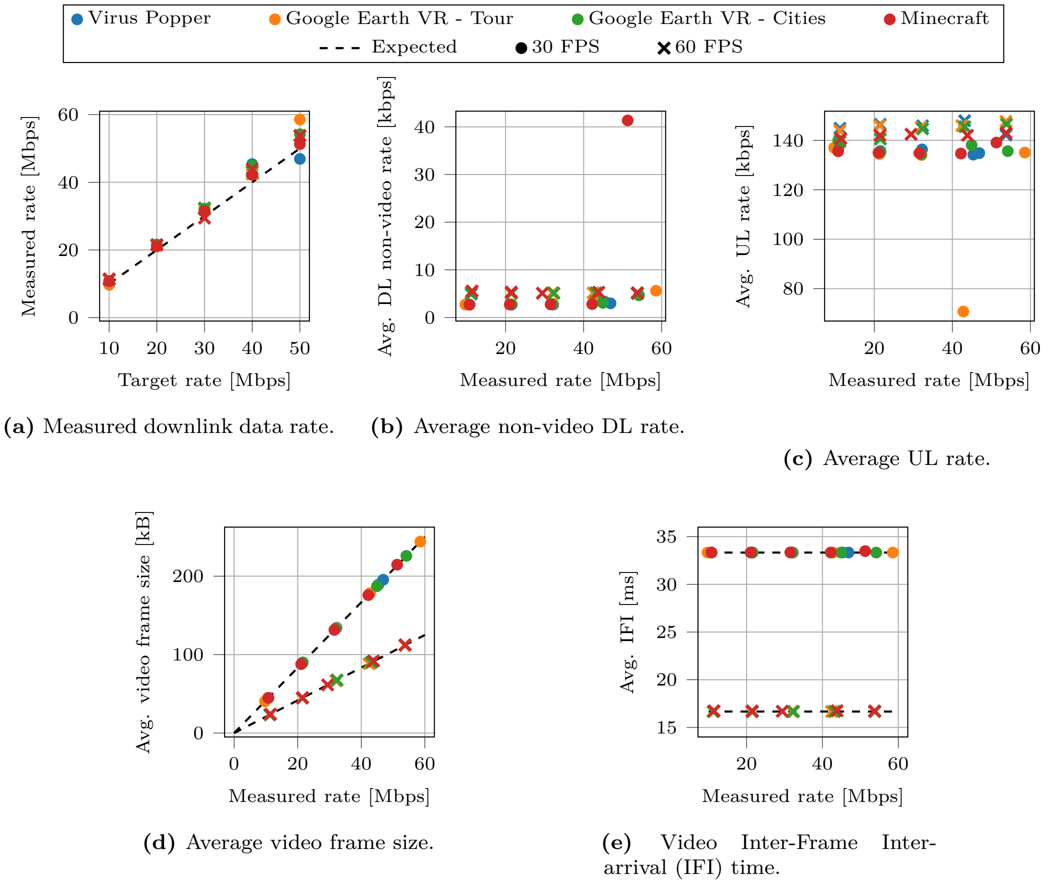
Most of the traffic is concentrated in DL and is made up of packet bursts encoding video frames. Video frame fragments were consistently found to be 1320 B long in all acquired traces, with a data size (the UDP payload) of 1278 B. The last packet of the burst also has the same size as the others, suggesting that padding has been used in order to simplify the protocol, although this biases the frame size distribution to be discrete.

The second most noticeable traffic stream is the UL head tracking information, which the headset acquires and sends to the rendering server to update the viewport to be rendered. The head tracking payload was identified to be either 192 B or 97 B long, sometimes changing over the course of a single traffic trace, although the reason why different packet sizes were found is unclear.

Finally, smaller packets in both UL and DL, with payloads of respectively 21 B and 10 B, were identified to contain feedback on the reception of video frames, which is probably used in the streaming protocol to decide whether or not to retransmit some frames.

By reverse-engineering the bits composing the UDP payload of video frames, it was possible to identify a recurring set of bits suggesting a 31 B APP layer header and allowing us to identify some key fields, such as (i) the frame sequence number, (ii) the number of fragments composing the frame, (iii) the fragment sequence number, (iv) the total frame size, and (v) a checksum. This information allowed us to reliably process and aggregate video frames.

*Available: https://github.com/nxrighthere/ENet-CSharp



(a) Measured downlink data rate.

(b) Average non-video DL rate.

(c) Average UL rate.

(d) Average video frame size.

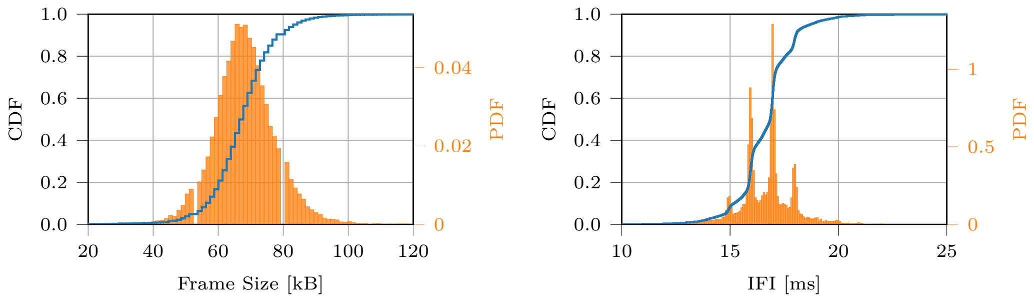
(e) Video Inter-Frame Inter-arrival (IFI) time.

Fig. 5.2: Results from acquired VR traffic traces.

Given the settings of the streaming application (i.e., frame rate and target data rate), it is clear that a CBR video encoding is performed in the background. In Fig. 5.2a we show the performance of the video encoder, almost always exceeding the target rate (though by only 5–10%). A simple explanation of this behavior might be the underestimation of header sizes in the computations of the CBR encoder, such as the header of the custom ENet protocol. Notably, both frame rates behave similarly across all four applications, with stable performance.

Figs. 5.2b and 5.2c show the low overhead due to non-video DL and UL transmissions (including head tracking), respectively. Specifically, non-video DL traffic only accounts for 3–5 kbps while UL traffic for about 135–150 kbps, with 60 FPS traces consistently showing higher rates with respect to 30 FPS ones, probably due to the doubled amount of feedback. Only two out of our forty traces show different rates, possibly due to some imperfection in the streaming. In any case, these traffic flows are much lower than the target rates and appear constant, irrespective of the data rate or the application. This consideration lead us to the decision of ignoring them, focusing only on modeling the DL video traffic.

Considering R the target data rate and F the application frame rate, the *average video frame*



(a) Video frame size distribution.

(b) IFI distribution.

Fig. 5.3: Video frame distributions for *Virus Popper* (30 Mbps, 60 FPS).

size is expected to be close to the ideal $S = R/F$, as shown in Fig. 5.2d. Note that the x-axis reports the measured data rate rather than the target data rate, i.e., the average data rate estimated from the acquired traces, which differs slightly from the target rate, as shown in Fig. 5.2a.

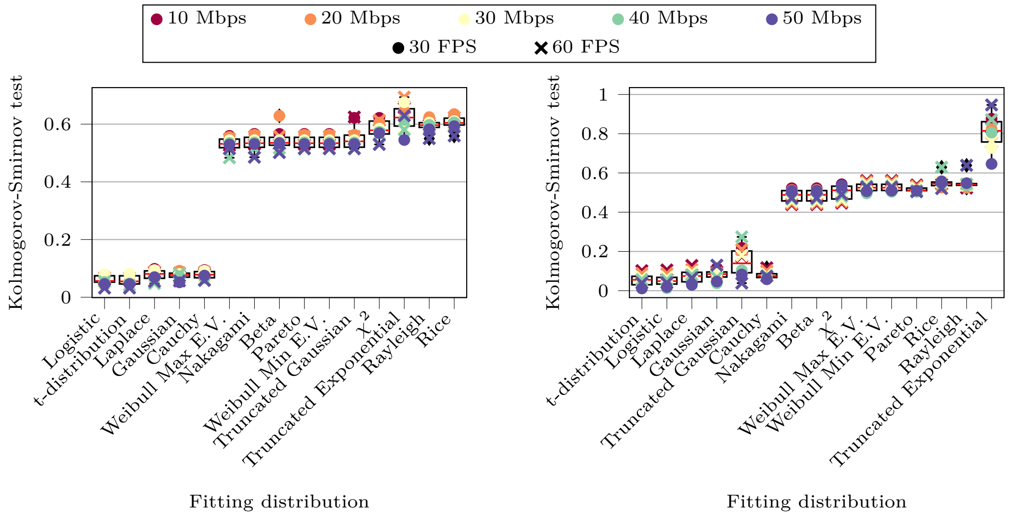
Furthermore, Fig. 5.2e shows that the average Inter-Frame Inter-arrival (IFI) time perfectly matches the expected $1/F$, equal to $33.\bar{3}$ ms for 30 FPS traces and $16.\bar{6}$ ms for 60 FPS traces.

Moving to the analysis of the Probability Density Functions (PDFs), it is important to know that in a collection of packets associated to a video source, we can usually distinguish Intra-coded frames (I-frames) (sometimes called *keyframes*), Predictive-coded frames (P-frames), and Bipredictive-coded frames (B-frames). While I-frames are compressed similarly to simple static pictures, P-frames exploit the temporal correlation of successive frames to reduce the compressed frame size. B-frames, instead, can exploit the information from both previous and subsequent frames, further improving the compression efficiency at the cost of non-real-time transmission. All the details associated with these compression techniques are regulated by standards like H.264 [188].

Interestingly, Fig. 5.3a shows that the frame size distribution is unimodal rather than multimodal, as would be expected considering the different compression levels of the I, P, and B frames generated by a typical H.264 encoder. As confirmed by the *RiftCat* team, the reason for such a smooth frame size distribution is that the encoder makes use of the H.264 Periodic Intra Refresh compression scheme where the reference image used to predict (and compress) the frames in a GoP, rather than being the first image as in H.264, is instead obtained from consecutive vertical slides taken from all the frames in the GoP. This results in a reduced variance of the frame size, making the encoded video stream almost CBR.

As already mentioned, *VRidge* simplifies the transmission by discretizing some units. Fig. 5.3a shows a clear staircase CDF for the video frame size, suggesting that video frames have been padded as the underlying distribution is indeed discrete, with a distance between consecutive stairs of 1278 B, i.e., the UDP payload of packets containing fragments of video frames.

Similarly, the IFI time also appears to be discretized with a millisecond precision around the



(a) Video frame size fit quality.

(b) IFI fit quality.

Fig. 5.4: Video frame fit qualities for the *Google Earth VR – Cities* application. Fit quality is measured using the KS test (lower is better). Box plots show median (red), 1$^{\text{st}}$ and 3$^{\text{rd}}$ quartiles (box), minimum and maximum (whiskers) of the KS test with a given distribution, while markers show the exact values for the different traces.

mean $\frac{1}{F}$, as seen in Fig. 5.3b, although some noise due to, e.g., variable rendering and encoding time, wireless channel condition, transmission queue state, transmission times, just to mention a few, smooths the CDF.

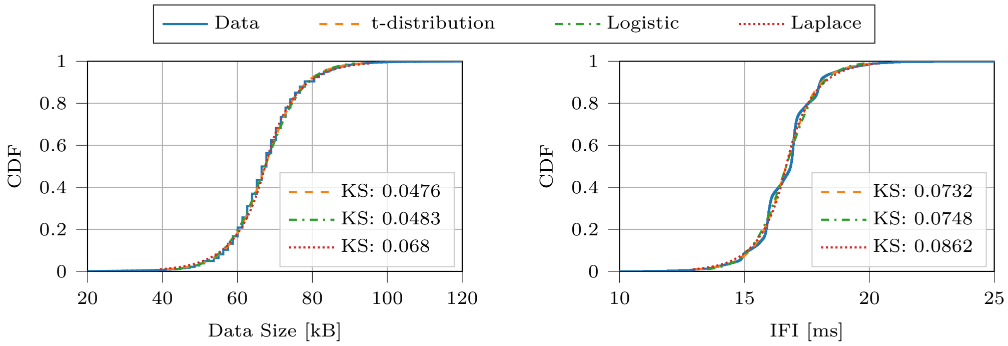
5.4 Traffic Model

Following the analysis of Sec. 5.3.2, in this section we will describe the proposed model for VR traffic based on the collected VR traffic traces.

The analysis presented in the previous section reveals that both packet sizes and IFI times appear to be discrete in the collected data traces. However, such granularity is likely due to specific design choices of the communication protocols used by the considered applications, rather than being a native characteristic of the XR services. Therefore, we believe it is more suitable to use continuous random variables to model the size of the data blocks generated by the XR application and the time between them. By doing so, we free our model from the specific constraints of this streaming application, with no loss of generality (as the discrete case can always be obtained from the continuous one), and in fact making it easier to accommodate other (non-discrete) cases in our framework if needed.

5.4.1 Distribution fitting

Given the extremely large number of samples per trace (200–600 s at 30 or 60 FPS), common quality of fit statistical tests yield poor performance due to the discretized distributions. Intuitively, while the PDF of discrete and continuous distributions takes completely different forms,



(b) IFI distribution comparison.

(a) Video frame size distribution comparison.

Fig. 5.5: Comparison of the three best fitting distributions for *Virus Popper* (30 Mbps, 60 FPS). The KS test is also shown, where lower values indicate a better fit.

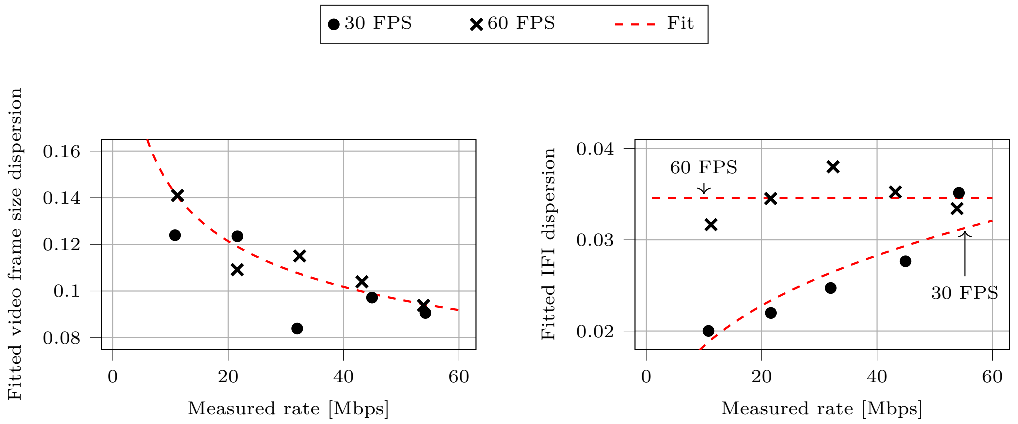
the CDF of a discretized distribution is simply a staircased version of the related continuous distribution. In that case, the goodness of fit can be tested by comparing the CDFs, for example using the KS test [189], defined as:

$$\mathrm{KS} = \sup_x \left| F_e(x) - F_t(x) \right| , \tag{5.1}$$

where $\sup_x$ is the supremum of the set of distances, $F_e(x)$ is the empirical CDF of the acquired data, and $F_t(x)$ is the CDF of the target distribution. The KS test will thus be used to score the quality of fit, where values closer to zero indicate a better parameter estimation.

To fit and evaluate the best probability distributions for our data, we used the popular SciPy library [190]. We tested 15 of the most common continuous univariate distributions available in the `scipy.stats` package, evaluating their performance on both frame size and IFI on our traffic traces. Note that the SciPy library performs a maximum likelihood estimation of the parameters of the distribution, including *location* and *scale*, and applies them to all continuous distributions by transforming the random variable X into $(X - \mathrm{loc})/\mathrm{scale}$. Given the exceptional accordance between expected values and computed averages (see Figs. 5.2d and 5.2e) and considering the proposed generative model (described in Sec. 5.4.2), we fixed the location parameter to the expected value (i.e., R/F for the frame size and $1/F$ for the IFI), fitting only the scale and the remaining parameters. A selection of distributions is shown in Fig. 5.4.

We found that the *Student's t* and *Logistic* distributions, closely followed by the *Laplace*, *Gaussian*, and *Cauchy* distributions, were the best fitting ones in almost all traces for both frame size and IFI. Fig. 5.5 shows how similar the fitted distributions actually are. Although the Student's t distribution performs slightly better than the Logistic one in the slight majority of the collected traces, in our case the Logistic distribution was the best choice. In fact, the third parameter of the Student's t distribution is only able to yield minuscule improvements over the Logistic distribution, which only needs two parameters. Furthermore, if custom simulators need to manually implement the desired random stream, the Student's t distribution is very



(a) Video frame size model.

(b) IFI model.

Fig. 5.6: Generalization models for the *Google Earth VR – Cities* application. Individual points show the *scale* parameter of the Logistic model fitted on the acquired data, while the dashed red lines attempt to generalize the model to intermediate target data rates.

hard to reproduce [191], while the Logistic distribution requires a simple transformation. This is the case when common libraries for random number generation cannot be used, such as in our implementation described in Sec. 5.6.

As a reference, we use SciPy's definition of a logistic distribution, with PDF in its standardized form as follows:

$$f(x) = \frac{e^{-x}}{(1+e^{-x})^2}. \tag{5.2}$$

To shift or scale the distribution, the location and scale parameters are used as previously described.

5.4.2 Generative Model

Now that we characterized and fitted the statistical distributions of the 40 acquired traces, we want to define a generative model which would allow a user to synthesize XR traffic at will, be it for analysis or simulation purposes. As already discussed in the previous section, in this chapter we propose a simple generative model, that only attempts to capture the statistical distributions of video frame size and Inter-Frame Inter-arrivals (IFIs), leaving higher-order statistical descriptions for future work.

We define the *dispersion* as the ratio of the *scale* over the *location* parameter, attempting to find a common value for both frame rates, since absolute values are likely to differ by a constant factor (see Figs. 5.2d and 5.2e). While data aggregation is doable for frame sizes (as shown, for example, in Fig. 5.6a), data for IFI did not allow us to do so. As shown in Fig. 5.6b, in fact, data for 30 and 60 FPS behaves differently, making it impossible for us to create a single model for this parameter. This implies that our model will only be able to generalize over data rates, whereas 30 and 60 FPS are the only supported frame rates, and modeling and testing different

Algorithm 5.1 Generative model for XR traffic

```
function GenerativeModel(AppName, FrameRate, DataRate)
    FsAvg = DataRate / FrameRate
    IfiAvg = 1 / FrameRate
    $\alpha$, $\beta$, $\gamma$, $\delta$, $\epsilon$ = GetParameters (AppName)        ▷ see Table 5.1
    FsDispersion = $\alpha \cdot$ DataRate$^\beta$
    FsScale = FsDispersion · FsAvg
    if FrameRate == 60
        IfiDispersion = $\gamma$
    else if FrameRate == 30
        IfiDispersion = $\delta \cdot$ DataRate$^\epsilon$
    else
        Error: only 30 and 60 FPS supported
    IfiScale = IfiDispersion · IfiAvg
```

values would require new data for the corresponding frame rate.

After carefully studying the acquired traffic traces, we propose to generalize the scale parameters for both video frame size and IFI time with a power law, namely:

$$y = ax^b. \tag{5.3}$$

Furthermore, as Fig. 5.6b suggests, the 60 FPS IFI fits for all applications resulted in $|b| < 10^{-4}$, suggesting a constant behavior, irrespective of the data rate. In that case, we thus assumed a constant fit (a corner case of power law with $b = 0$) by computing the average value across all tested target data rates.

As can clearly be observed from the collected data, the proposed model has been extracted from acquisitions between about 10 and 50 Mbps, thus using it beyond these limits is not advisable since no data in our possession can validate the quality of the synthetic traces.

We let different applications have separate models, obtaining a data set of 10 traces per application (half at 30 FPS, half at 60 FPS). The parameters for all applications can be found in Table 5.1 and the generative algorithm is summarized in Alg. 5.1.

5.5 Ns-3 Implementation

To properly model and test the performance of VR traffic over a simulated network, a flexible application framework has been implemented in ns-3 and made publicly available [192]. The framework is based on the ns-3.33 release and aims at providing a novel additional traffic model, easily customizable by the final user.

Tab. 5.1: Parameters of the proposed generative model. Each VR application is characterized by five parameters: two for the frame size dispersion $D_{FS} = \alpha x^\beta$, one for the 60 FPS IFI dispersion $D_{IFI} = \gamma$, two for the 30 FPS IFI dispersion $D_{IFI} = \delta x^\epsilon$.

	α	β	γ	δ	ϵ
Virus Popper	0.1784	-0.2403	0.03721	0.01433	0.1764
Minecraft	0.1857	-0.1872	0.07133	0.02419	0.2267
GE VR – Tour	0.2554	-0.2031	0.03468	0.01056	0.2756
GE VR – Cities	0.2597	-0.2539	0.03457	0.008953	0.3119

The proposed framework allows the user to send packet bursts fragmented into multiple packets by BurstyApplication, later re-aggregated at the receiver, if possible, by BurstSink. Since the generation of packet bursts is crucial to model a wide range of possibilities, a generic BurstGenerator interface has been defined. Users can implement arbitrary generators by extending this interface, and three examples have been provided and will be described in Sec. 5.5.2. Finally, each fragment comprises a novel SeqTsSizeFragHeader, which includes information on both the fragment and the current burst, allowing BurstSink to correctly re-aggregate or discard a burst, yielding information on received fragments, received bursts, and failed bursts.

More details on the implementation and the rationale behind these applications will be given in the following sections.

5.5.1 Bursty Application

Inspired by the acquired traffic traces described in Sec. 5.3.1, BurstyApplication periodically sends bursts of data divided into multiple smaller fragments of (at most) a given size. Since burst size and period statistics can be quite general, the generation of the burst statistics is delegated to objects extending the BurstGenerator interface, later described in Sec. 5.5.2. BurstyHelper is also implemented to simplify the generation and installation of BurstyApplications with given BurstGenerators to network nodes and examples are provided.

Each fragment carries a SeqTsSizeFragHeader, an extension of SeqTsSizeHeader which adds the information on the fragment sequence number and the total number of fragments composing the burst, on top of the (burst) sequence number and size as well as the transmission time-stamp. After setting a desired FragmentSize in bytes, the application will compute how many fragments will be generated to send the full burst to the target receiver, although the last two fragments may be smaller due to the size of the burst not being a multiple of the fragment size, and the presence of the extra header.

Traces notify the user when fragments and bursts are sent, while also keeping track of the number of bursts, fragments, and bytes sent, making it easier to quickly compute some simple high-level metrics directly from the main script of the simulation.

5.5.2 Burst Generator Interface

A generic bursty application can show extremely different behaviors. For example, an application could send a given amount of data periodically in a deterministic fashion, or the burst size or the period could be random with arbitrary statistics, successive bursts could be correlated (e.g., the concept of GoP for video-coding standards such as H.264 [193]), and even the burst size and the time before the next burst might be correlated.

To accommodate for the widest range of possibilities, a BurstGenerator interface has been defined. Classes extending this interface must define two pure virtual functions:

1. HasNextBurst: to ensure that the burst generator is able to generate a new burst size and the time before the next burst (also called *next period* in the remainder of this section);

2. `GenerateBurst`: yielding the burst size of the current burst as well as the next period, if it exists.

Three classes extending this interface are proposed and briefly discussed in the remainder of this section, allowing users to generate very diverse statistics without the need to implement their own custom generator in most cases.

Simple Burst Generator Inspired from `OnOffApplication`, `SimpleBurstGenerator` defines the current burst size and the next period as generic `RandomVariableStream`s. Users are thus able to model arbitrary burst size and next period distributions, by: using the distributions already implemented in ns-3; implementing more distributions; or simply defining arbitrary CDFs for `EmpiricalRandomVariable`s.

Limitations for this generator lie in the correlation of the generated random variables: burst size and next period are independently drawn as are successive bursts.

VR Burst Generator `VrBurstGenerator` is a direct implementation of the model proposed in Sec. 5.4, where bursts model video frames.

Similar to the *RiftCat* software described in Sec. 5.3.1, this generator makes it possible to choose a target data rate and a frame rate.

While traces were taken at specific frame rates and target data rates, the proposed model attempts to generalize them, although without any knowledge on the quality of the generalization beyond the boundaries imposed by the streaming software.

To generate the frame size and the next period, `LogisticRandomVariable` and `MixtureRandomVariable` have been implemented in ns-3.

A validation of the proposed model based on this burst generator will be discussed in Sec. 5.6.

Trace File Burst Generator Finally, users might want to reproduce in ns-3 a traffic trace obtained by a real application, generated by a separate traffic generator, or even manually written by a user (e.g., for static debugging/testing purposes). For these reasons, `TraceFileBurstGenerator` was introduced, taking advantage of `CsvReader` to parse a csv-like file declaring a (burst size, next period) pair for each row. Once traces are imported, the generator will sequentially yield every burst, returning `false` as output to `TraceFileBurstGenerator::HasNextBurst` after the last row of the trace file is yielded, thus stopping the `BurstyApplication`.

A `StartTime` can be set as an attribute, allowing the user to control which part of the file trace will be used in the simulation. This can be especially useful when the total simulation duration is shorter than the traffic trace, making it possible to decouple users by setting different start times.

Several VR traffic traces using different frame rates and target data rates are available [192] in the described format for a total of over 90 minutes of processed acquisitions, comprising some relevant metadata as part of the commented header. Interested readers can thus simulate real VR video traffic in their ns-3 simulations, or expand the analysis performed in Secs. 5.3.2 and 5.4.

5.5.3 Burst Sink

An adaptation of the existing `PacketSink`, called `BurstSink`, is proposed for the developed bursty framework. This new application expects to receive packets from users equipped with `BurstyApplications` and tries to re-aggregate fragments into packets.

While the current version of `PacketSink` is able to assemble byte streams with `SeqTsSizeHeader`, there are two reasons why `BurstSink` was created, specifically (i) to stress the dependence of this framework on UDP rather than TCP sockets, as the acquisitions suggested, thus expecting individual fragments sent unreliably rather than a reliable byte stream, and (ii) to trace the reception at both the fragment and the burst level.

The application implements a simple best-effort aggregation algorithm, assuming that (i) the burst transmission duration is much shorter than the *next period*, and (ii) all fragments are needed to re-aggregate a burst. Specifically, fragments of a given burst are collected, even if unordered, and, if all fragments are received, the burst is successfully received. If, instead, fragments of subsequent bursts are received before all fragments of the previous one, then the previous burst is discarded. Information on the current fragment and burst can easily be recovered from `SeqTsSizeFragHeader`, allowing the application to verify whether a burst has been fully received or not. If needed and suggested by real-world applications, future works might also introduce the concept of APP layer Forward Error Correction (FEC).

Traces notify the user when fragments are received and when bursts are successfully received or discarded, together with all the related relevant information. Furthermore, similarly to the `BurstyApplication`, also the `BurstSink` application keeps track of the number of bursts, fragments, and bytes received.

5.6 Simulation Results

To further test the validity of the proposed model, we implemented it on top of ns-3, a popular open-source full-stack simulation software, and made it publicly available together with the processed VR traffic traces in `CSV` format.* Further details on the implementation of this traffic model on ns-3 can be found in [187].

To test our model, we set up simulation campaigns where multiple users equipped with HMDs communicate with a central Wi-Fi AP, using a wireless connection based on the IEEE 802.11ac standard. The central AP also acts as rendering server, generating one VR stream for each receiving STA of the scenario. Transmissions randomly start within the first second of simulation, avoiding that different streams start at the same time.

We show results for traffic streams imported directly from the acquired traces as well as for our model. Since a single trace is available for each parameter combination (i.e., application, frame rate, data rate), for a fixed parameter combination the traffic flows will all come from the same trace, although different 60 s windows are sampled to further decouple different users. Instead, simulations running our proposed model have been repeated twice: one with the target

*Available in the ns-3 app store: https://apps.nsnam.org/app/bursty-app/, Release v1.0.0

data rate submitted to *VRidge* when acquiring the corresponding trace, and one with the empirical data rate measured directly from the acquired traces (the two rates differ slightly, as can be seen from Fig. 5.2a). This information can also be found directly into the metadata of the acquired traces, made available in CSV format together with our model.

5.6.1 Model Validation

Exhaustive simulation campaigns have been run for all four applications and five data rates at both 30 and 60 FPS, each repeated 10 times to obtain solid average statistics. Confidence intervals are not shown as they are extremely tight. Additional simulation parameters are shown in Table 5.2.

In the following section, plots will show burst-level rather than fragment-level metrics, which in case of a video stream are much more informative and bring a more realistic perspective on the quality perceived by the user. In fact, in this case we are more interested in the performance regarding full video frames rather than single packets, and thus all packets from a burst will have to be collected before the HMD will be able to process and show the frame to the user.

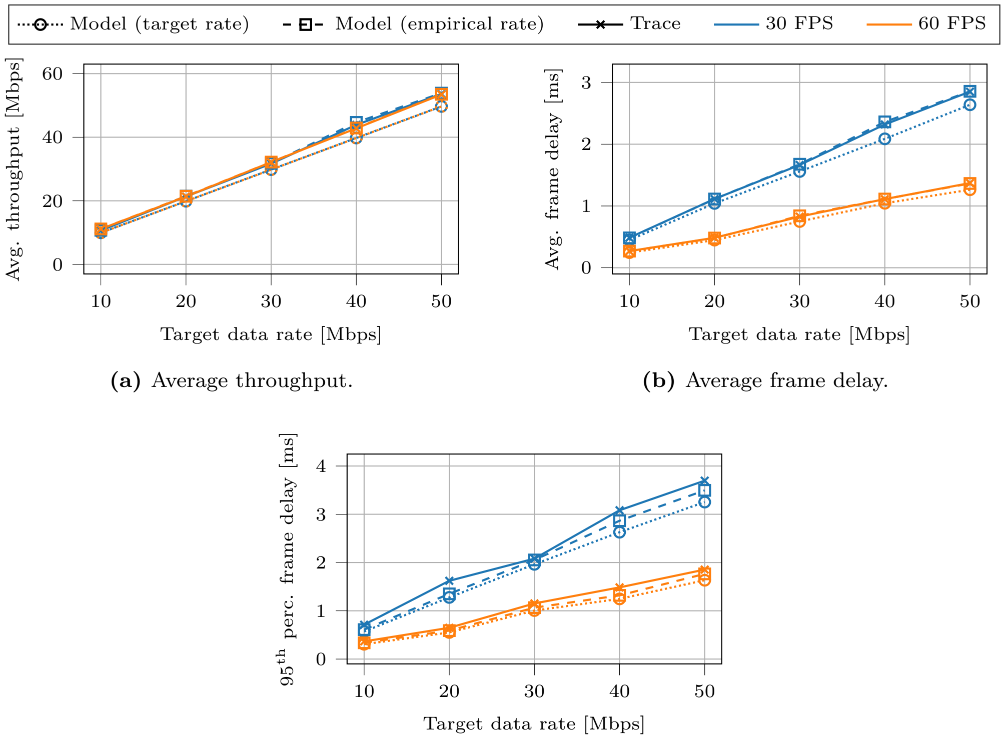
To validate our proposed model, we simulate a scenario as similar as possible to our acquisition setup, where a rendering server transmits the VR stream to a single user. Note that the Wi-Fi connection is able to withstand hundreds of megabits-per-second, thus a single user transmitting up to 50 Mbps is largely underutilizing the channel, allowing us to obtain unbiased results with respect to the limits of the channel capacity. We simulated all 40 combinations of parameters (4 applications, 5 data rates, 2 frame rates), although we only show results for the 10 related to the *Google Earth VR - Cities* application in Fig. 5.7.

In Fig. 5.7a we show the average throughput obtained by the 3 simulation campaigns in the 10 parameter sets. Clearly, both 30 and 60 FPS runs obtain similar results, since this metric disregards the frame rate. In fact, both models targeting the nominal rate (shown with dots and circular markers) are perfectly superimposed on the main diagonal. Simulations using the original traffic traces, instead, tend to have a slightly higher throughput (solid line with cross markers), as was expected by looking at Fig. 5.2a. Since data rate, frame size, and, conversely, latency are correlated, we matched our model's data rate with the empirical one, as shown by the dashed line with square markers. As the flexibility of our model allows us to choose an arbitrary target rate, we can see a perfect match in the computed average throughput.

In Fig. 5.7b, instead, we show the average frame delay measured from the APP layer of the AP to the APP layer of the STA. Processing, encoding/decoding and other technical delays

Tab. 5.2: List of simulation parameters

Parameter	Value	Parameter	Value
Duration	60 s	RTS/CTS	Disabled
Distance	1 m	MCS	VHT MCS 9
Mobility	Fixed	Channel Width	160 MHz
IP	v4	Guard Interval Duration	400 ns
Transport	UDP	Fragment Size	1472 B



(a) Average throughput.

(b) Average frame delay.

(c) 95^{th} percentile of frame delay.

Fig. 5.7: Simulation results for a single user streaming the *Google Earth VR – Cities* application over a Wi-Fi link. The statistics refer to fully received frames rather than to single fragments.

must be added to obtain the full *motion-to-photon* latency, and thus the network delay should remain below 5–10 ms, as mentioned in Sec. 5.1. The most noticeable difference with respect to the previous figure is that the two frame rates are clearly separated. This is because our reference application, described in Sec. 5.3.2, allows us to choose a target data rate, trying to maintain a CBR transmission during the whole duration of the stream. This translates into frame sizes which depend directly on the frame rate, following the formula $S = R/F$ as described in Sec. 5.3.2. Since the channel capacity for these simulations is kept constant, doubling the frame rate halves the frame sizes which, in turn, halves the average video frame delay. As expected, the model using the target rate slightly underestimates the average frame delay, which depends on the real application throughput, always slightly lower than the one empirically computed from the traffic traces. Instead, similarly to the average throughput, setting the model to the trace's empirical rate yields an almost perfect match with the VR traces we acquired. Finally, notice that the average frame delay always remains below 3 ms, well below the bound suggested by the industry experts [164, 169, 152, 154, 159].

To complete this analysis, in Fig. 5.7c we report the 95^{th} percentile delay performance of our simulations. This metric is important as it gives an idea of the worst-case performance of the network. In fact, only ensuring average performance is not enough to obtain a smooth and appreciable user experience, since frequent stutters in the streamed video might easily ruin the interactivity of the application and even disorient the user. Ensuring that the 95th percentile of the delay is within acceptable bounds allows for a more fluid and overall better experience. In the case under analysis, it can be easily seen that both models using target and empirical rate slightly underestimate the frame delay of the acquired traces. It is likely that the fitted Logistic distribution is not able to fully grasp the minute details of the traffic trace, making our model unable to match the real traffic.

Note that, while these results are bound to the specifications of the network under analysis (e.g., MCS, channel width, guard interval duration, fragment size, presence of RTS/CTS, Wi-Fi standard, mobility, environment) the framework that we proposed is general. This suggests that it can be used to study a variety of more or less complex scenarios and network architectures with different sets of parameters, assessing how they affect the end-to-end performance.

To conclude, it appears that our model is indeed able to reliably predict average statistics, while it could still be improved to better mimic slightly more advanced and specific features. These refinements will be pursued in our future work.

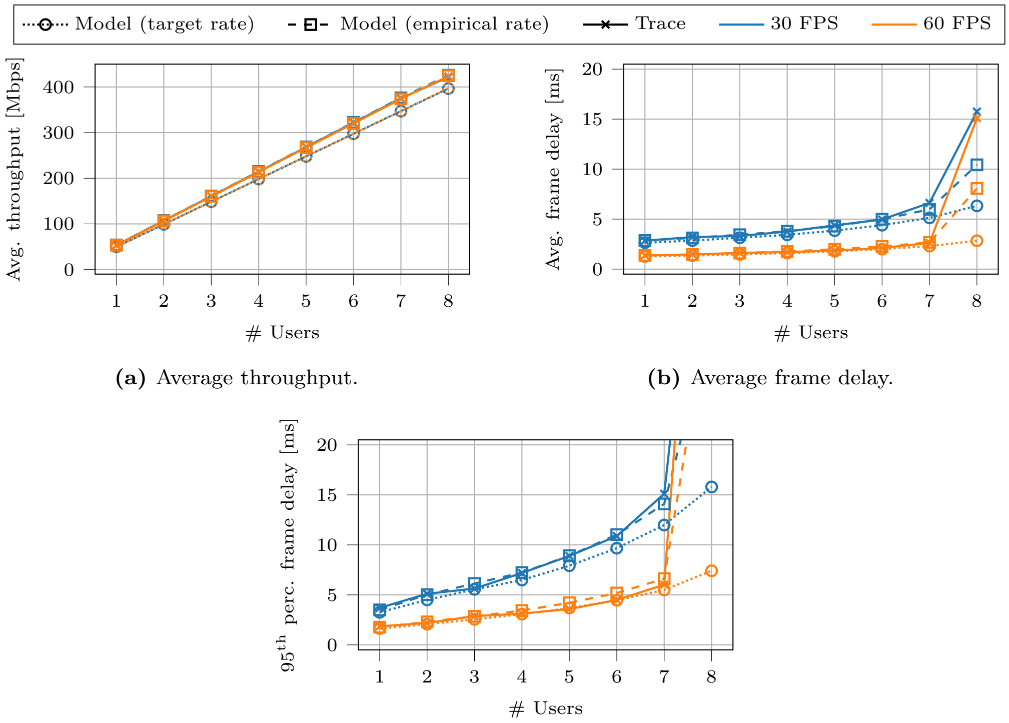
5.6.2 Use Case Example

Finally, we propose a simple example use case for our VR traffic generator. We consider a VR arena setting, where multiple users are attached directly to a single AP streaming wirelessly. We assume that each user requests a 50 Mbps stream and observe how many STAs can be supported by an arena with an analogous setup.

As expected, we notice again from Fig. 5.8 that our model needs to be calibrated against the empirical rate of the acquired trace to yield reliable results. In fact, from Fig. 5.8a we can see that the average throughput of the calibrated model perfectly matches the throughput of the traffic trace up to at least 8 users, where the network is able to support more than 400 Mbps.

In Fig. 5.8b it is possible to see an unstable network condition, when 8 users are trying to stream simultaneously. It appears that the slightly higher throughput required by the trace and the empirical rate model with respect to the target rate model is enough to push the network to its limit, resulting in a sudden increase of the average frame delay, at both 30 and 60 FPS. Focusing on the 30 FPS simulations, the plot shows that up to 6 users can be supported within the 5 ms bound, while 7 users slightly exceed this limit, and finally 8 users make the network unstable and are thus pushed over the 10 ms limit for both the trace and the model using the empirical rate. It is important to notice that the more unstable the network, the worse the prediction accuracy of our model. This is probably due to the simplifications that we introduced, such as the Logistic distribution and the uncorrelated samples for both the IFI time and frame size stochastic processes. Similarly, at 30 FPS, up to 7 users can be supported, but an additional user makes the system highly unstable and with poor prediction performance from our model.

Finally, in Fig. 5.8c we show the results for the 95th percentile of the delay. Similarly to the average delay, this metric also shows the instability of the network for 8 users with much



(a) Average throughput.

(b) Average frame delay.

(c) 95th percentile of frame delay.

Fig. 5.8: Simulation results for multiple users streaming the *Google Earth VR – Cities* application over a Wi-Fi link. The statistics refer to fully received frames rather than to single fragments.

worse performance. Focusing on 30 FPS, the system is able to keep the delay below the 5 ms bound only when no more than 2 users are present, whereas up to 5-6 users can be served if a 10 ms delay is still deemed acceptable. Instead, at 60 FPS up to 6 users can be served while keeping the network delay within 5 ms, while the 10 ms limit is only surpassed when the network becomes unstable with 8 users.

These counterintuitive conclusions come from the fact that the application fixes a data rate, not a quality of experience. This means that doubling the frame rate results in halving the frame size, thus reducing the perceived image quality of the streamed application, which turns into an almost halved delay. Fixing a constant bit rate thus results in higher frame rates yielding lower latencies, at the cost of a lower image quality.

In general, there is good accordance between the results predicted by the calibrated model and the traffic traces, while the uncalibrated model often shows overly optimistic results. When the traffic in the network increases too much and the network becomes unstable, the three simulations diverge significantly, making our synthetic traces less reliable, although this is a corner case that might be of lesser interest.

5.7 XR Traffic Modeling Roadmap

Starting from the model described in the previous sections, in the following we propose an end-to-end framework to evaluate network solutions, tailored for XR applications. The goal is to list and detail the tasks required for the construction of such a framework, in order to encourage researchers in this field to advance with their work the state of the art, using our baseline as a valid starting point.

While Sec. 5.7.1 is devoted to highlighting our contributions, in Secs. 5.7.2, 5.7.3 and 5.7.5 we set down each additional task, describing how they can lead to the optimization of network protocols.

5.7.1 Exploiting First-Order Statistics

The model proposed in this thesis, despite its basic functionalities, represents a solid foundation on top of which future works can iterate to develop more sophisticated strategies. In particular, we designed an open-source, highly customizable setup (described in Sec. 5.3) to acquire traffic traces by sniffing the packets traveling on the local network where the experiments were conducted.

At this stage, packets are generated following first-order statistics, sampling the size and inter-frame interval from the distribution fitted on the collected data (see Sec. 5.3.2). As a consequence, with this model we can emulate the creation of application frames that replicate the strategy implemented by the rendering server used in our experiments. While this model is already useful for some applications, it lends itself to several interesting extensions, which capture other important features of the statistics of XR traffic. As an example, in the rest of this section we discuss the importance of studying the correlation between different packets and of understanding how the movements of the user impact the generated traffic as two key areas of future improvement for our model.

5.7.2 Introducing Temporal Correlation

More advanced studies can be carried out to improve the model with additional features. One important aspect to elaborate on is the correlation among subsequent frames, or even within a specific group of frames.

As mentioned in Sec. 5.3.2, when compressing a video stream both intra-frame and inter-frame compression techniques could be exploited, and this influences not only the structure of the packets since the type of compression greatly influences the frame size, but also the strategy to inject them into the network. It is also possible that some manufacturers use advanced coding techniques such as Periodic Intra Refresh, as was explained in Sec. 5.3.2 for the streaming application used for our analysis, or more advanced standards such as H.265 [194] using different compression techniques. In that case, the importance of temporal correlation might decrease, although further analysis should be carried out to ensure this.

It should be clear, by now, that the availability of a model capable of generalizing how such frame sequences are created, independently of the technical setup, is important, and the fact that each manufacturer may use its own policy represents an additional challenge. In addition, having a model that integrates and generalizes the temporal correlation among frames would allow researchers to elaborate strategies to guarantee a certain level of latency and throughput, for example by giving different priorities and scheduling options to different types of packets.

For applications with constant delay requirements and high values of FPS, a solution could be to buffer (at the device side or at the rendering server) specific packets associated with keyframes, in order to improve the encoding process. This would require stable network performance and an application capable of communicating directly with the network, e.g., exploiting cross-layer solutions, to be aware of any change of the link quality that would trigger specific countermeasures or improvements, if applicable.

5.7.3 Introducing Head Tracking

A further improvement of the model should exploit the information on movement tracking, in particular related to the head, for all 6DoF. In this case, sniffing the packets traveling through the network might not be enough, and we thus need to gather information from different sensors (e.g., gyroscope, accelerometer and compass), that could be integrated into the device used to interact with the virtual world.

With respect to VRidge, the software that we used to make our phone acting as a VR headset and our PC as a rendering server, the developers provide an API for this purpose.[*] By connecting to the head tracking endpoint, the software provides positional, rotational, or combined data, and even the possibility of modifying phone tracking data in real time before it is used for the rendering step.

This is important because, by aligning the motion trace with the traffic generated by the application, it can be determined whether there is correlation between a certain movement of the user and the corresponding drop in the reception of packets, or other network-related events. For example, knowing the direction of the physical movement of the user might help mmW wireless systems (such as 802.11ad/ay) keep beam alignment between the AP and the user device, thus limiting the risk of abrupt connection interruptions if the line of sight is lost.

It is to be highlighted that this approach could benefit every communication infrastructures that can be used to deliver XR content, as user tracking data can be exploited at different layers of the protocol stack.

5.7.4 Full Traffic Emulator

The last step to further increase the fidelity (but also the complexity) of the traffic model is to fully characterize and emulate all the different information sub-flows and how they interact with each other. For example, as shown in Fig. 5.1 and explained in Sec. 5.3.2, the VR stream

[*] `https://github.com/RiftCat/vridge-api`

comprises both DL and UL messages containing information such as video frames, head tracking information, and feedback.

A full-blown emulator would send all this information to and from the user, reacting accordingly whenever a packet is lost or corrupted, or when communication delays are present. This level of detail requires a much more in-depth analysis of the transmission protocol of a real XR application, understanding all the consequences of erratic and unexpected behaviors of the network.

Such a precise model would be extremely useful when running large simulation campaigns as it would give the most accurate and reliable results. However, the amount of work required to analyze and reproduce a realistic behavior would be extremely high.

5.7.5 QoE-centric XR

As highlighted thoroughly in the previous paragraphs, the final goal of all these approaches is to guarantee high-level performance to the final user. In particular, in the XR domain, we tend to measure the performance in terms of overall satisfaction of the customers, referred to as QoE, and, to the best of our knowledge, there is no standardized way to evaluate these metrics.

In our case, besides the quality of the shown image, also the latency of the communication between the HMD and the rendering server can make a difference (especially if the latter is in the cloud), considering that cybersickness has a huge impact on the user experience. For this reason, researchers should be encouraged to design algorithms that guarantee stable and constant performance, taking into account that the traffic in the network varies depending on the application and user activity.

Moreover, since in a common scenario we have different users, there may be a need to support different traffic categories at the same time in the same network. This requires a system able to fairly distribute resources among the flows, where learning algorithms could be implemented to orchestrate every operation, either from a network or from an application perspective.

Given a certain condition of the user, or other available information, the algorithm could predict the QoE trend and act accordingly in case of an anticipated performance drop. At this point on the roadmap, the network design should focus on the user, trying to guarantee a stable experience also when VBR flows are considered. In fact, in a CBR flow (much easier to handle from a network point of view), the perceived image quality can be affected in case of a scene with a large amount of action and details. In this case, it may be difficult to fit everything at a fixed rate and, as a consequence, the user experiences a downgrade in terms of quality. This further highlights the need for novel solutions, able to tackle these problems by trading off system complexity and QoE.

5.8 Conclusions

In this chapter we described the current state of the art regarding the telecommunication aspects needed to support high-quality XR streaming, mainly focusing on the challenges needed to obtain faithful traffic models that the community could use to test protocols and optimize networks.

We then proceeded to acquire over 4 hours of VR traffic, study in detail this type of traffic, and propose a model to generate synthetic traffic traces, while also making freely available to the community both our implementation and the VR dataset [187, 195].

Finally, we show some results on the predictive power of our model, while also acknowledging its weak points. Furthermore, we provided an example use case where multiple users coexist in the same network, naively sharing radio resources up to its collapse. Further work could better study effective scheduling strategies for XR traffic streams, possibly coexisting with other applications in the same network while also ensuring robustness in case of fluctuating channel quality. Also, the model could be tested and validated for higher values of FPS, by collecting and analyzing additional traces at 90 FPS or higher. All the tasks that we think are necessary to build a complete framework for traffic generation have been listed in Sec. 5.7 and represent possible future directions for this work.

The proposed ns-3 framework for bursty applications is publicly available and open source [192], together with the implementation of the proposed traffic model and the actual traffic traces experimentally obtained. We also attempted to generalize the model to arbitrary target data rates and frame rates, allowing users to experiment with arbitrary application-level settings that suit their specific research. The model has been built upon a framework to simulate bursty applications in ns-3, where burst size and period can be customized with little additional code, and traces for burst-level metrics collections allow the user to better analyze a complex application QoS.

With this contribution, we hope to pave the way for the research community to start working towards the optimization and support of this specific type of traffic, given the extreme interest from the main standard bodies and the most prominent telecommunication industries.

6

Conclusion

In this thesis, we discussed about previous limitations for the simulation of communication networks, the advancements that we proposed, and future research directions to further enhance the current state of the art in multiple topics. We invite the reader to consult the previous chapters for a per-topic discussion of relevant future research directions, and report here only the main conclusions drawn by our works.

After a general introduction to the topics discussed in this thesis, in the second chapter we focused mostly on channel modeling, aiming towards simulation scalability on one side and better mmW channel modeling on the other. We first analyzed and proposed simplifications to improve the performance of the stochastic SCM described in 3GPP TR 38.901, obtaining a significant channel generation speedup with little behavioral changes, in terms of both narrowband and wideband channel modeling. Then, after briefly introducing the theory of radio frequency RTs based on the Method of Images, we discussed about two simplification approaches specific for this type of channel modeling. After analyzing both PHY layer and E2E performance, we found that there is indeed a tradeoff between computational complexity and reliability, giving some rules of thumb on how to set there parameters depending on the scenario under analysis. Finally, with respect to more detailed channel models, we introduced a mathematical formulation for a class of mmW channels, i.e., the QD models, that can closely simulate the propagation of rays in a specific environment calibrated on real-world channel measurements. We provided a step-by-step tutorial on how such models can be implemented, and then compared the model against measurements. We also briefly introduced a work in progress software where blockage at mmW can be modeled and then studied with appropriate tools, such as full stack simulators.

In the third chapter, we instead analyze antenna arrays, a topic tightly linked with mmW communications, from both an optimization perspective and a modeling perspective. We propose an ML-based optimization framework allowing us to jointly optimize many antenna parameters at once in just a fraction of the time that would be required otherwise. Our findings suggest that vertical linear arrays provide the best performance in urban scenarios, significantly better than square or rectangular planar arrays. We then presented a modeling framework for the end-to-end evaluation of 5G mmW cellular networks which is compliant with the 3GPP NR specifications. We propose additions to ns-3 such as (i) a ray-tracing based channel model for mobile users, which improves the spatio-temporal coherence over the previous stochastic channel [43], (ii) a flexible antenna module, comprising multiple parametric antenna elements as well as a generic interface for phased antenna arrays, and (iii) a BF module supporting different algorithms for the computation of the optimal BF vectors. This work gives great flexibility to the final user and thus allows for a plethora of new and more complex simulations, which exploiting all the communication modules already implemented in ns-3 will be able to yield E2E performance to more realistic systems.

In the fourth chapter, we started by briefly describing the main characteristics of the WiGig standards, with focus on the MAC layer. We gave some details about the scheduling mechanisms introduced by IEEE 802.11ad, allowing different types of traffic to coexist and potentially improving the performance of QoS-sensitive applications. We then proposed an open-source framework for WiGig periodic scheduling based on ns-3. We tested our framework by implementing two simple schedulers and testing their performance with different types of network traffic. Results show that the new scheduling mechanism proposed by WiGig standards is able to outperform the typical Wi-Fi contention-based channel access, but only when precise cross-layer information between the MAC and APP layers is exchanged. Moreover, even tiny variations over the perfectly CBR behavior of the application traffic result in significant performance degradation, suggesting that some extra resources might need to be allocated to retain the desired performance. Lastly, we presented a mathematical framework for periodic scheduling in WiGig networks, the the objective of designing more solid and better performing schedulers. We studied if and how multiple periodic allocations, possibly with different periodicities and requirements, can be scheduled together with a strong periodicity requirement, needed to eliminate possible jitter in the communication. We proposed, described, and tested two scheduling algorithms based on this framework, showing that they have almost opposite behaviors: one aims at a high traffic flow allocation acceptance rate, the other maximizes the resources allocated to each single station. We showed the complexity behind the design of these algorithms, as well as the benefits that can be obtained by properly tuning them.

Finally, in chapter five, we describe our work trying to model a VR application traffic flow. After setting up an acquisition system, we recorded and analyzed hours of VR traffic streaming, studying its details and proposing a model to generate synthetic traffic traces, while also making freely available to the community both our implementation and the VR dataset. We proposed an ns-3 framework for bursty application in order to support VR-like traffic types, automatically handling the fragmentation and re-aggregation at the two ends of the communication system. This allows users to implement their own model, other than using our own or our VR traffic dataset. We propose a way to generalize our model to arbitrary target data rates at both 30 and 60 FPS, giving lots of freedom to the user for their own simulations. With this work, we hope to pave the way for the research community to start working towards the optimization and support of this specific type of traffic, given the extreme interest from the main standard bodies and the most prominent telecommunication industries.

The telecommunication community is currently observing the deployment of 5G networks around the world, fruit of a decade-long collaboration between worldwide universities, research centers, and big industries, all jointly collaborating and investing in research for this common goal. Researchers are already discussing about future evolutions of this technology, also known as *beyond 5G*, as well as the next generation of communications networks: 6G.

References

[1] ITU-R, “IMT Vision - Framework and overall objectives of the future development of IMT for 2020 and beyond,” International Telecommunication Union (ITU), Recommendation ITU-R M.2083, Sep. 2018.

[2] T. Nitsche, C. Cordeiro, A. B. Flores, E. W. Knightly, E. Perahia, and J. C. Widmer, “IEEE 802.11ad: directional 60 GHz communication for multi-gigabit-per-second Wi-Fi,” *IEEE Communications Magazine*, vol. 52, no. 12, pp. 132–141, Dec. 2014.

[3] P. Zhou, K. Cheng, X. Han, X. Fang, Y. Fang, R. He, Y. Long, and Y. Liu, “IEEE 802.11ay-Based mmWave WLANs: Design Challenges and Solutions,” *IEEE Communication Surveys and Tutorials*, vol. 20, no. 3, pp. 1654–1681, Third quarter 2018.

[4] 3GPP, “New WID on extending current NR operation to 71 GHz,” Qualcomm, RP-193229, Dec. 2019.

[5] T. S. Rappaport, S. Sun, R. Mayzus, H. Zhao, Y. Azar, K. Wang, G. N. Wong, J. K. Schulz, M. Samimi, and F. Gutierrez, “Millimeter wave mobile communications for 5G cellular: It will work!” *IEEE Access*, vol. 1, pp. 335–349, 2013.

[6] S. Sun, T. S. Rappaport, R. W. Heath, A. Nix, and S. Rangan, “MIMO for millimeter-wave wireless communications: Beamforming, spatial multiplexing, or both?” *IEEE Commununication Magazine*, vol. 52, no. 12, pp. 110–121, Dec. 2014.

[7] M. Zhang, M. Polese, M. Mezzavilla, J. Zhu, S. Rangan, S. Panwar, and M. Zorzi, “Will TCP Work in mmWave 5G Cellular Networks?” *IEEE Commun. Mag.*, vol. 57, no. 1, pp. 65–71, Jan. 2019.

[8] M. Giordani, M. Mezzavilla, A. Dhananjay, S. Rangan, and M. Zorzi, “Channel dynamics and SNR tracking in millimeter wave cellular systems,” in *22nd European Wireless Conference*, Oulu, Finland, May 2016.

[9] M. Giordani, M. Polese, A. Roy, D. Castor, and M. Zorzi, “A Tutorial on Beam Management for 3GPP NR at mmWave Frequencies,” *IEEE Communication Surveys and Tutorials*, vol. 21, no. 1, pp. 173–196, First quarter 2019.

[10] M. Giordani, M. Mezzavilla, and M. Zorzi, “Initial access in 5G mmWave cellular networks,” *IEEE Communication Magazine*, vol. 54, no. 11, pp. 40–47, Nov. 2016.

[11] S. Sun, T. S. Rappaport, M. Shafi, P. Tang, J. Zhang, and P. J. Smith, “Propagation models and performance evaluation for 5G millimeter-wave bands,” *IEEE Transactions on Vehicular Technologies*, vol. 67, no. 9, pp. 8422–8439, Jun. 2018.

[12] S. K. Saha, Y. Ghasempour, M. K. Haider, T. Siddiqui, P. D. Melo, N. Somanchi, L. Zakrajsek, A. Singh, R. Shyamsunder, O. Torres *et al.*, "X60: A programmable testbed for wideband 60 GHz WLANs with phased arrays," *Computer Communications*, vol. 133, pp. 77–88, Jan. 2019.

[13] M. Polese, F. Restuccia, A. Gosain, J. Jornet, S. Bhardwaj, V. Ariyarathna, S. Mandal, K. Zheng, A. Dhananjay, M. Mezzavilla, J. Buckwalter, M. Rodwell, X. Wang, M. Zorzi, A. Madanayake, and T. Melodia, "MillimeTera: Toward A Large-Scale Open-Source MmWave and Terahertz Experimental Testbed," in *3rd ACM Workshop on Millimeter-Wave Networks and Sensing Systems (mmNets)*, Los Cabos, Mexico, Oct. 2019, pp. 27–32.

[14] 3GPP, "NR and NG-RAN Overall Description," 3rd Generation Partnership Project (3GPP), Technical Specification (TS) 38.300, Jan. 2018, Version 15.0.0.

[15] *IEEE 802.11ax-2021 - IEEE Standard for Information Technology – Telecommunications and Information Exchange between Systems Local and Metropolitan Area Networks – Specific Requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 1: Enhancements for High-Efficiency WL*, IEEE P802.11 - Task Group ax Std., May 2021.

[16] *802.11ad-2012 - IEEE Standard for Information Technology – Telecommunications and Information Exchange Between Systems – Local and Metropolitan Area Networks – Specific Requirements-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 3: Enhancements for Very High Throughput in the 60 GHz Band*, IEEE P802.11 - Task Group ad Std., Dec. 2012, Superseded.

[17] *IEEE 802.11ay-2021 - IEEE Standard for Information Technology – Telecommunications and Information Exchange between Systems Local and Metropolitan Area Networks – Specific Requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 2: Enhanced Throughput for Operation in License-exempt Bands above 45 GHz*, IEEE P802.11 - Task Group ay Std., Mar. 2021.

[18] E. Hamida, G. Chelius, and J. M. Gorce, "Impact of the Physical Layer Modeling on the Accuracy and Scalability of Wireless Network Simulation," *SIMULATION*, vol. 85, no. 9, pp. 574–588, Jun. 2009.

[19] M. Mezzavilla, M. Zhang, M. Polese, R. Ford, S. Dutta, S. Rangan, and M. Zorzi, "End-to-End Simulation of 5G mmWave Networks," *IEEE Communications Surveys and Tutorials*, vol. 20, no. 3, pp. 2237–2263, Third quarter 2018.

[20] T. Zugno, M. Polese, M. Lecci, and M. Zorzi, "Simulation of Next-Generation Cellular Networks with Ns-3: Open Challenges and New Directions," in *ACM Workshop on Next-Generation Wireless with Ns-3 (WNGW)*, Florence, Italy, Jun. 2019.

[21] S. Rangan, T. S. Rappaport, and E. Erkip, "Millimeter-Wave Cellular Wireless Networks: Potentials and Challenges," *Proceedings of the IEEE*, vol. 102, no. 3, pp. 366–385, Mar. 2014.

[22] Z. Hossain, Q. Xia, and J. Jornet, "TeraSim: An ns-3 Extension to Simulate Terahertz-band Communication Networks," *Nano Communication Networks*, vol. 17, pp. 36–44, Sep. 2018.

[23] P. Ferrand, M. Amara, S. Valentin, and M. Guillaud, "Trends and challenges in wireless channel modeling for evolving radio access," *IEEE Communications Magazine*, vol. 54, no. 7, pp. 93–99, Jul. 2016.

[24] 3GPP, "Study on channel model for frequencies from 0.5 to 100 GHz," 3rd Generation Partnership Project (3GPP), Technical Report (TR) 38.901, Jun. 2018, Version 15.0.0.

[25] M. Rebato, L. Resteghini, C. Mazzucco, and M. Zorzi, "Study of Realistic Antenna Patterns in 5G mmWave Cellular Scenarios," in *IEEE International Communication Conference (ICC)*, Kansas City, USA, May 2018.

[26] M. Drago, T. Azzino, M. Polese, Č. Stefanović, and M. Zorzi, "Reliable Video Streaming over mmWave with Multi Connectivity and Network Coding," in *International Conference on Computing, Networking and Communications (ICNC)*, Maui, Hawaii, USA, Mar. 2018, pp. 508–512.

[27] H. Ott, K. Miller, and A. Wolisz, "Simulation Framework for HTTP-Based Adaptive Streaming Applications," in *Workshop on ns-3 (WNS3)*, Porto, Portugal, Jun. 2017, pp. 95–102.

[28] M. Polese and M. Zorzi, "Impact of Channel Models on the End-to-End Performance of mmWave Cellular Networks," in *IEEE 19th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, Kalamata, Greece, Jun. 2018.

[29] T. Bai and R. W. Heath, "Coverage and rate analysis for millimeter-wave cellular networks," *IEEE Transactions on Wireless Communications*, vol. 14, no. 2, pp. 1100–1114, Feb. 2015.

[30] I. A. Hemadeh, K. Satyanarayana, M. El-Hajjar, and L. Hanzo, "Millimeter-Wave Communications: Physical Channel Models, Design Considerations, Antenna Constructions, and Link-Budget," *IEEE Communication Surveys and Tutorials*, vol. 20, no. 2, pp. 870–913, Second quarter 2018.

[31] M. R. Akdeniz, Y. Liu, M. K. Samimi, S. Sun, S. Rangan, T. S. Rappaport, and E. Erkip, "Millimeter wave channel modeling and cellular capacity evaluation," *IEEE Journal on Selected Areas of Communications*, vol. 32, no. 6, pp. 1164–1179, Jun. 2014.

[32] P. Testolina, M. Lecci, M. Polese, M. Giordani, and M. Zorzi, "Scalable and Accurate Modeling of the Millimeter Wave Channel," in *International Conference on Computing, Networking and Communications (ICNC)*, Big Island, Hawaii, US, Feb. 2020, pp. 969–974.

[33] V. Degli-Esposti, F. Fuschini, E. M. Vitucci, M. Barbiroli, M. Zoli, L. Tian, X. Yin, D. A. Dupleich, R. Müller, C. Schneider, and R. S. Thomä, "Ray-Tracing-Based mm-Wave Beamforming Assessment," *IEEE Access*, vol. 2, pp. 1314–1325, Dec. 2014.

[34] S. G. Larew, T. A. Thomas, M. Cudak, and A. Ghosh, "Air interface design and ray tracing study for 5G millimeter wave communications," in *IEEE Globecom Workshops (GC Wkshps)*, Atlanta, Georgia, US, Dec. 2013, pp. 117–122.

[35] C. Lai, R. Sun, C. Gentile, P. B. Papazian, J. Wang, and J. Senic, "Methodology for Multipath-Component Tracking in Millimeter-Wave Channel Modeling," *IEEE Transactions on Antennas and Propagation*, vol. 67, no. 3, pp. 1826–1836, Mar. 2019.

[36] G. R. MacCartney and T. S. Rappaport, "Rural Macrocell Path Loss Models for Millimeter Wave Wireless Communications," *IEEE Journal on Selelected Areas of Communications*, vol. 35, no. 7, pp. 1663–1677, Jul. 2017.

[37] C. Wang, J. Bian, J. Sun, W. Zhang, and M. Zhang, "A Survey of 5G Channel Measurements and Models," *IEEE Communication Surveys and Tutorials*, vol. 20, no. 4, pp. 3142–3168, Fourth quarter 2018.

[38] T. S. Rappaport, Y. Xing, G. R. MacCartney, A. F. Molisch, E. Mellios, and J. Zhang, "Overview of Millimeter Wave Communications for Fifth-Generation (5G) Wireless Networks – With a Focus on Propagation Models," *IEEE Transactions on Antennas and Propagation*, vol. 65, no. 12, pp. 6213–6230, Dec. 2017.

[39] J. G. Andrews, T. Bai, M. N. Kulkarni, A. Alkhateeb, A. K. Gupta, and R. W. Heath, "Modeling and Analyzing Millimeter Wave Cellular Systems," *IEEE Transactions on Communications*, vol. 65, no. 1, pp. 403–430, Jan. 2017.

[40] A. A. M. Saleh and R. Valenzuela, "A Statistical Model for Indoor Multipath Propagation," *IEEE Journal on Selelected Areas of Communications*, vol. 5, no. 2, pp. 128–137, Feb. 1987.

[41] IST WINNER II, "D1.1.2 V1.2 WINNER II Channel Models," 2007.

[42] M. Gapeyenko, V. Petrov, D. Moltchanov, S. Andreev, Y. Koucheryavy, M. Valkama, M. R. Akdeniz, and N. Himayat, "An Analytical Representation of the 3GPP 3D Channel Model Parameters for MmWave Bands," in *2nd ACM Workshop on Millimeter Wave Networks and Sensing Systems (mmNets)*, New Delhi, India, Oct. 2018, pp. 33–38.

[43] T. Zugno, M. Polese, N. Patriciello, B. Bojović, S. Lagen, and M. Zorzi, "Implementation of a Spatial Channel Model for Ns-3," in *Workshop on ns-3 (WNS3)*, Gaithersburg, MD, USA, Jun. 2020.

[44] J. Palacios, D. De Donno, and J. Widmer, “Tracking mm-Wave channel dynamics: Fast beam training strategies under mobility,” in *IEEE Conference on Computer Communications (INFOCOM)*, Atlanta, GA, US, May 2017.

[45] F. Fuschini, E. M. Vitucci, M. Barbiroli, G. Falciasecca, and V. Degli-Esposti, “Ray tracing propagation modeling for future small-cell and indoor applications: A review of current techniques,” *Radio Science*, vol. 50, no. 6, pp. 469–485, Jun. 2015.

[46] S. Hussain, “Efficient ray-tracing algorithms for radio wave propagation in urban environments,” Ph.D. dissertation, School of Electronic Engineering, Dublin City University, Sep. 2017.

[47] J. G. Cleary and G. Wyvill, “Analysis of an algorithm for fast ray tracing using uniform space subdivision,” *The Visual Computer*, vol. 4, no. 2, pp. 65–83, Mar. 1988.

[48] G. E. Athanasiadou, A. R. Nix, and J. P. McGeehan, “A microcellular ray-tracing propagation model and evaluation of its narrow-band and wide-band predictions,” *IEEE Journal on Selelected Areas of Communications*, vol. 18, no. 3, pp. 322–335, Mar. 2000.

[49] C. Saeidi and F. Hodjatkashani, “Modified angular Z-buffer as an acceleration technique for ray tracing,” *IEEE Transactions on Antennas and Propagation*, vol. 58, no. 5, pp. 1822–1825, May 2010.

[50] V. Degli-Esposti, F. Fuschini, E. M. Vitucci, and G. Falciasecca, “Speed-up techniques for ray tracing field prediction models,” *IEEE Transactions on Antennas and Propagation*, vol. 57, no. 5, pp. 1469–1480, May 2009.

[51] C. Saeidi, F. Hodjatkashani, and A. Fard, “Efficient point-to-area electromagnetic field calculation using tube tracing method,” in *International Conference on Wireless Communications & Signal Processing (WCSP)*, Nanjing, China, Nov. 2009.

[52] R. Hoppe, G. Wolfle, and F. M. Landstorfer, “Accelerated ray optical propagation modeling for the planning of wireless communication networks,” in *IEEE Radio and Wireless Conference (RAWCON)*, Denver, CO, US, Aug. 1999, pp. 159–162.

[53] N. Mataga, R. Zentner, and A. K. Mucalo, “Ray entity based postprocessing of ray-tracing data for continuous modeling of radio channel,” *Radio Science*, vol. 49, no. 3, pp. 217–230, Mar. 2014.

[54] R. A. Valenzuela, S. Fortune, and J. Ling, “Indoor propagation prediction accuracy and speed versus number of reflections in image-based 3-D ray-tracing,” in *48th IEEE Vehicular Technology Conference (VTC)*, Ottawa, ON, Canada, May 1998, pp. 539–543.

[55] Z. Yun and M. F. Iskander, “Ray Tracing for Radio Propagation Modeling: Principles and Applications,” *IEEE Access*, vol. 3, pp. 1089–1100, Dec. 2015.

[56] J. Pascual-García, J. Molina-García-Pardo, M. Martínez-Inglés, J. Rodríguez, and N. Saurín-Serrano, "On the importance of diffuse scattering model parameterization in indoor wireless channels at mm-Wave frequencies," *IEEE Access*, vol. 4, pp. 688–701, Feb. 2016.

[57] University of Padova – SIGNET group, "Open-source ray tracer." [Online]. Available: https://github.com/signetlabdei/qd-realization/tree/feature/power-threshold

[58] J. B. Keller, "Geometrical theory of diffraction," *Journal of the Optical Society of America*, vol. 52, no. 2, pp. 116–130, Feb. 1962.

[59] S. Y. Tan and H. S. Tan, "A theory for propagation path-loss characteristics in a city-street grid," *IEEE Transactions on Electromagnetic Compatibility*, vol. 37, no. 3, pp. 333–342, Mar. 1995.

[60] V. Degli-Esposti, D. Guiducci, A. de'Marsi, P. Azzi, and F. Fuschini, "An Advanced Field Prediction Model Including Diffuse Scattering," *IEEE Transactions on Antennas and Propagation*, vol. 52, no. 7, pp. 1717–1728, Jul. 2004.

[61] A. Maltsev *et al.*, "Channel Models for IEEE 802.11ay," 11-15/1150r9, IEEE Task Group ay (TGay), Tech. Rep., 2017.

[62] M. Lecci, M. Polese, C. Lai, J. Wang, C. Gentile, N. Golmie, and M. Zorzi, "Quasi-Deterministic Channel Model for mmWaves: Mathematical Formalization and Validation," in *IEEE Global Communications Conference (GLOBECOM)*, Taipei, Taiwan, Dec. 2020.

[63] ITU-R, "Propagation by diffraction," International Telecommunication Union (ITU), Recommendation ITU-R P.526-15, Oct. 2019.

[64] K. Bullington, "Radio propagation for vehicular communications," *IEEE Transactions on Vehicular Technology*, vol. 26, no. 4, pp. 295–308, Nov. 1977.

[65] ITU-R, "Attenuation in vegetation," International Telecommunication Union (ITU), Recommendation ITU-R P.833-9, Sep. 2016.

[66] ——, "Propagation data and prediction methods for the planning of indoor radiocommunication systems and radio local area networks in the frequency range 300 MHz to 450 GHz," International Telecommunication Union (ITU), Recommendation ITU-R P.1238-10, Aug. 2019.

[67] ——, "Propagation data and prediction methods required for the design of terrestrial broadband radio access systems operating in a frequency range from 3 to 60 GHz," International Telecommunication Union (ITU), Recommendation ITU-R P.1410-5, Feb. 2012.

[68] ——, "Effects of building materials and structures on radiowave propagation above about 100 MHz," International Telecommunication Union (ITU), Recommendation ITU-R P.2040-1, Jul. 2015.

[69] "METIS Channel Models," Deliverable D1.4, Feb. 2015.

[70] J.-E. Berg, "A recursive method for street microcell path loss calculations," in *IEEE 6th International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, Toronto, ON, Canada, Sep. 1995.

[71] "METIS Simulation Guidelines," Deliverable D6.1, Oct. 2013.

[72] H. Friis, "A note on a simple transmission formula," *Proceedings of the IRE*, vol. 34, no. 5, pp. 254–256, May 1946.

[73] A. Glassner, *An Introduction to Ray Tracing*, A. Glassner, Ed. Morgan Kaufmann, 1989.

[74] R. Kouyoumjian and P. Pathak, "A uniform geometrical theory of diffraction for an edge in a perfectly conducting surface," *Proceedings of the IEEE*, vol. 62, no. 11, pp. 1448–1461, Nov. 1974.

[75] P. Pathak, W. Burnside, and R. Marhefka, "A uniform gtd analysis of the diffraction of electromagnetic waves by a smooth convex surface," *IEEE Transactions on Antennas and Propagation*, vol. 28, no. 5, pp. 631–642, Sep. 1980.

[76] J. McKown and R. Hamilton, "Ray tracing as a design tool for radio networks," *IEEE Network*, vol. 5, no. 6, pp. 27–30, Nov. 1991.

[77] M. D. Casciato, "Radio wave diffraction and scattering models for wireless channel simulation," Ph.D. dissertation, University of Michigan, 2001.

[78] M. Ghaddar, L. Talbi, T. A. Denidni, and A. Sebak, "A conducting cylinder for modeling human body presence in indoor propagation channel," *IEEE Transactions on Antennas and Propagation*, vol. 55, no. 11, pp. 3099–3103, Nov. 2007.

[79] K. Schuler, D. Becker, and W. Wiesbeck, "Extraction of virtual scattering centers of vehicles by ray-tracing simulations," *IEEE Transactions on Antennas and Propagation*, vol. 56, no. 11, pp. 3543–3551, Nov. 2008.

[80] J. Kunisch and J. Pamp, "Ultra-wideband double vertical knife-edge model for obstruction of a ray by a person," in *IEEE International Conference on Ultra-Wideband*, Hannover, Germany, Sep. 2008.

[81] M. Jacob, S. Priebe, A. Maltsev, A. Lomayev, V. Erceg, and T. Kurner, "A ray tracing based stochastic human blockage model for the IEEE 802.11ad 60 GHz channel model," in *5th European Conference on Antennas and Propagation (EuCAP)*, Rome, Italy, Apr. 2011.

[82] A. Maltsev *et al.*, "Channel models for 60 GHz WLAN systems," 09/0334r8, IEEE Task Group ad (TGad), Tech. Rep., May 2010.

[83] J. Medbo and F. Harrysson, “Channel modeling for the stationary UE scenario,” in *7th European Conference on Antennas and Propagation (EuCAP)*, Gothenburg, Sweden, Apr. 2013.

[84] A. Bhardwaj, D. Caudill, C. Gentile, J. Chuang, J. Senic, and D. G. Michelson, “Geometrical-Empirical Channel Propagation Model for Human Presence at 60 GHz,” *IEEE Access*, vol. 9, pp. 38 467–38 478, 2021.

[85] 3GPP, “Study on channel model for frequency spectrum above 6 GHz,” 3rd Generation Partnership Project (3GPP), Technical Report (TR) 38.900, Jul. 2018, Version 15.0.0.

[86] ITU-R, “Multipath propagation and parameterization of its characteristics,” International Telecommunication Union (ITU), Recommendation ITU-R P.1407, Jun. 2017.

[87] M. Lecci, P. Testolina, M. Giordani, M. Polese, T. Ropitault, C. Gentile, N. Varshney, A. Bodi, and M. Zorzi, “Simplified ray tracing for the millimeter wave channel: a performance evaluation,” in *Information Theory and Applications Workshop (ITA)*, San Diego, CA, USA, Feb. 2020.

[88] T. R. Henderson, M. Lacage, G. F. Riley, C. Dowell, and J. Kopena, “Network simulations with the ns-3 simulator,” in *SIGCOMM (demo)*, Seattle, WA, USA, Aug. 2008.

[89] M. Polese, M. Giordani, M. Mezzavilla, S. Rangan, and M. Zorzi, “Improved Handover Through Dual Connectivity in 5G mmWave Mobile Networks,” *IEEE Journal on Selected Areas of Communications*, vol. 35, no. 9, pp. 2069–2084, Sep. 2017.

[90] D. W. Browne, M. W. Browne, and M. P. Fitz, “Singular Value Decomposition of Correlated MIMO Channels,” in *IEEE Global Communications Conference (GLOBECOM)*, San Francisco, CA, US, Dec. 2006.

[91] wigig-tools, “Open-source ray tracer.” [Online]. Available: https://github.com/wigig-tools/qd-realization

[92] M. Lecci, P. Testolina, M. Polese, M. Giordani, and M. Zorzi, “Accuracy vs. Complexity for mmWave Ray-Tracing: A Full Stack Perspective,” *IEEE Transactions on Wireless Communications*, Jun. 2021, Early Access.

[93] V. Va and R. W. Heath, “Basic Relationship between Channel Coherence Time and Beamwidth in Vehicular Channels,” in *IEEE 82nd Vehicular Technology Conference (VTC2015-Fall)*, Boston, MA, USA, Sep. 2015.

[94] A. Alkhateeb, “DeepMIMO: A Generic Deep Learning Dataset for Millimeter Wave and Massive MIMO Applications,” in *Information Theory and Applications Workshop (ITA)*, San Diego, CA, US, Feb. 2019.

[95] M. Arnold, S. Dörner, S. Cammerer, S. Yan, J. Hoydis, and S. ten Brink, “Enabling FDD Massive MIMO through Deep Learning-based Channel Prediction,” Jan. 2019, Pre-print: arXiv:1901.03664.

[96] O. Simeone, “A Very Brief Introduction to Machine Learning With Applications to Communication Systems,” *IEEE Transactions on Cognitive Communications and Networking*, vol. 4, no. 4, pp. 648–664, Dec. 2018.

[97] T. J. O’Shea and J. Hoydis, “An Introduction to Deep Learning for the Physical Layer,” *IEEE Transactions on Cognitive Communications and Networking*, vol. 3, no. 4, pp. 563–575, Dec. 2017.

[98] M. Zorzi, A. Zanella, A. Testolin, M. De Filippo De Grazia, and M. Zorzi, “Cognition-Based Networks: A New Perspective on Network Optimization Using Learning and Distributed Intelligence,” *IEEE Access*, vol. 3, pp. 1512–1530, Dec. 2015.

[99] A. Testolin, M. Zanforlin, M. De Filippo De Grazia, D. Munaretto, A. Zanella, M. Zorzi, and M. Zorzi, “A machine learning approach to QoE-based video admission control and resource allocation in wireless systems,” in *13th Annual Mediterranean Ad Hoc Networking Workshop (MED-HOC-NET)*, Piran, Slovenia, Jun. 2014.

[100] H. Sun, X. Chen, Q. Shi, M. Hong, X. Fu, and N. D. Sidiropoulos, “Learning to Optimize: Training Deep Neural Networks for Interference Management,” *IEEE Transactions on Signal Processing*, vol. 66, no. 20, pp. 5438–5453, Oct. 2018.

[101] S. Ghosh and D. Sen, “An Inclusive Survey on Array Antenna Design for Millimeter-Wave Communications,” *IEEE Access*, vol. 7, pp. 83 137–83 161, Nov. 2019.

[102] Y. Cheng, W. Shao, S. Zhang, and Y. Li, “An Improved Multi-Objective Genetic Algorithm for Large Planar Array Thinning,” *IEEE Transactions on Magnetics*, vol. 52, no. 3, Mar. 2016.

[103] M. Jijenth, K. K. Suman, V. S. Gangwar, A. K. Singh, and S. P. Singh, “A novel technique based on modified genetic algorithm for the synthesis of thinned planar antenna array with low peak side lobe level over desired scan volume,” in *IEEE MTT-S International Microwave and RF Conference (IMaRC)*, Ahmedabad, India, Dec. 2017.

[104] P. Rocca and R. L. Haupt, “Dynamic thinning strategy for adaptive nulling in planar antenna arrays,” in *IEEE International Symposium on Phased Array Systems and Technology*, Waltham, MA, US, Oct. 2010, pp. 995–997.

[105] C. A. Balanis, *Antenna Theory: Analysis and Design.* Wiley-Interscience, 2005.

[106] M. Zhang, M. Polese, M. Mezzavilla, S. Rangan, and M. Zorzi, “ns-3 Implementation of the 3GPP MIMO Channel Model for Frequency Spectrum above 6 GHz,” in *Workshop on ns-3 (WNS3)*, Porto, Portugal, Jun. 2017.

[107] 3GPP, "Study on scenarios and requirements for next generation access technologies," 3rd Generation Partnership Project (3GPP), Technical Report (TR) 38.913, Jul. 2018, Version 15.0.0.

[108] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics).* Berlin, Heidelberg: Springer-Verlag, 2006.

[109] Y. M. Abdelkader, M. M. Hamada, and A. N. Mohieldin, "System Level Co-Simulation Approach for Ultra-Wideband Massive MIMO Beam Forming Phased Array Transmitters," in *31st International Conference on Microelectronics (ICM)*, Cairo, Egypt, Mar. 2019.

[110] Y. J. Cho, G. Suk, B. Kim, D. K. Kim, and C. Chae, "RF Lens-Embedded Antenna Array for mmWave MIMO: Design and Performance," *IEEE Communications Magazine*, vol. 56, no. 7, pp. 42–48, Jul. 2018.

[111] C. Menudier, J. Lintignat, S. Mons, P. Médrel, N. Delhote, E. Ngoya, S. Bila, M. Thévenot, B. Jarry, P. Gamand, J. Sombrin, and D. Baillargeat, "Design and optimization of multi-element antennas and RF circuits for beamforming with a reduced number of RF Front-ends," in *IEEE MTT-S International Microwave Workshop Series on 5G Hardware and System Technologies (IMWS-5G)*, Dublin, Ireland, Aug. 2018.

[112] P. Testolina, M. Lecci, M. Rebato, A. Testolin, J. Gambini, R. Flamini, C. Mazzucco, and M. Zorzi, "Enabling Simulation-Based Optimization Through Machine Learning: A Case Study on Antenna Design," in *IEEE Global Communication Conference (GLOBECOM)*, Waikoloa, Hawaii, USA, Dec. 2019.

[113] M. Lecci, P. Testolina, M. Rebato, A. Testolin, and M. Zorzi, "Machine Learning-Aided Design Of Thinned Antenna Arrays For Optimized Network Level Performance," in *14th European Conference on Antennas and Propagation (EuCAP)*, Copenhagen, Denmark, Mar. 2020.

[114] M. Lecci, T. Zugno, S. Zampato, and M. Zorzi, "A Full-Stack Open-Source Framework for Antenna and Beamforming Evaluation in mmWave 5G NR," in *IEEE International Conference on Communications (ICC)*, Montreal, QC, Canada, Jun. 2021.

[115] Y. Ghasempour, C. R. C. M. da Silva, C. Cordeiro, and E. W. Knightly, "IEEE 802.11ay: Next-Generation 60 GHz Communication for 100 Gb/s Wi-Fi," *IEEE Communications Magazine*, vol. 55, no. 12, pp. 186–192, Dec. 2017.

[116] H. Assasa and J. Widmer, "Implementation and Evaluation of a WLAN IEEE 802.11ad Model in ns-3," in *Workshop on ns-3 (WNS3)*, Seattle, Washington, Jun. 2016, pp. 57–64.

[117] H. Assasa, J. Widmer, T. Ropitault, and N. Golmie, "Enhancing the Ns-3 IEEE 802.11ad Model Fidelity: beam Codebooks, Multi-Antenna Beamforming Training, and Quasi-Deterministic MmWave Channel," in *Workshop on ns-3 (WNS3)*, Florence, Italy, Jun. 2019, pp. 33–40.

[118] G. Bianchi, I. Tinnirello, and L. Scalia, "Understanding 802.11e contention-based prioritization mechanisms and their coexistence with legacy 802.11 stations," *IEEE Network*, vol. 19, no. 4, pp. 28–34, Jul. 2005.

[119] F. Babich and M. Comisso, "Throughput and delay analysis of 802.11-based wireless networks using smart and directional antennas," *IEEE Transactions on Communications*, vol. 57, no. 5, pp. 1413–1423, May 2009.

[120] G. Bianchi, "Performance Analysis of the IEEE 802.11 Distributed Coordination Function," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 3, pp. 535–547, Mar. 2000.

[121] C. Pielli, T. Ropitault, N. Golmie, and M. Zorzi, "An Analytical Model for CBAP Allocations in IEEE 802.11ad," *IEEE Transactions of Communications*, vol. 69, no. 1, pp. 649–663, Jan. 2021.

[122] T. Azzino, T. Ropitault, and M. Zorzi, "Scheduling the Data Transmission Interval in IEEE 802.11ad: A Reinforcement Learning Approach," in *International Conference on Computing, Networking and Communications (ICNC)*, Big Island, HI, US, Feb. 2020.

[123] C. Hemanth and T. G. Venkatesh, "Performance Analysis of Contention-Based Access Periods and Service Periods of 802.11ad Hybrid Medium Access Control," *IET Networks*, vol. 3, no. 3, pp. 193–203, Sep. 2014.

[124] M. N. U. Rajan and A. V. Babu, "Saturation Throughput Analysis of IEEE 802.11ad Wireless LAN in the Contention Based Access Period (CBAP)," in *IEEE Distributed Computing, VLSI, Electrical Circuits and Robotics (DISCOVER)*, Mangalore, India, Aug. 2016.

[125] E. Khorov, A. Ivanov, A. Lyakhov, and V. Zankin, "Mathematical Model for Scheduling in IEEE 802.11ad Networks," in *IFIP Wireless and Mobile Networking Conference (WMNC)*, Colmar, France, Jul. 2016.

[126] C. L. Liu and J. W. Layland, "Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment," *Journal of the ACM*, vol. 20, no. 1, pp. 46–61, Jan. 1973.

[127] K. Ramamritham, "Allocation and Scheduling of Precedence-Related Periodic Tasks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 6, no. 4, pp. 412–420, Apr. 1995.

[128] Sheng-Tzong Cheng and A. K. Agrawala, "Allocation and Scheduling of Real-Time Periodic Tasks with Relative Timing Constraints," in *International Workshop on Real-Time Computing Systems and Applications (RTCSA)*, Tokyo, Japan, Oct. 1995, pp. 210–217.

[129] Dakai Zhu, D. Mosse, and R. Melhem, "Multiple-Resource Periodic Scheduling Problem: How Much Fairness is Necessary?" in *IEEE Real-Time Systems Symposium (RTSS)*, Cancun, Mexico, Dec. 2003, pp. 142–151.

[130] M. N. U. Rajan and A. V. Babu, "Theoretical maximum throughput of IEEE 802.11 ad millimeter wave wireless LAN in the contention based access period: with two level aggregation," in *International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, Chennai, India, Mar. 2017, pp. 2531–2536.

[131] H. Shokri-Ghadikolaei, L. Gkatzikis, and C. Fischione, "Beam-searching and transmission scheduling in millimeter wave communications," in *IEEE International Conference on Communications (ICC)*, London, UK, Jun. 2015, pp. 1292–1297.

[132] W. Feng, Y. Wang, D. Lin, N. Ge, J. Lu, and S. Li, "When mmWave communications meet network densification: a scalable interference coordination perspective," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 7, pp. 1459–1471, Jul. 2017.

[133] D. Steinmetzer, D. Wegemer, and M. Hollick. (2017) Talon Tools: The Framework for Practical IEEE 802.11ad Research. [Online]. Available: https://seemoo.de/talon-tools

[134] D. Steinmetzer, D. Wegemer, M. Schulz, J. Widmer, and M. Hollick, "Compressive Millimeter-Wave Sector Selection in Off-the-Shelf IEEE 802.11ad Devices," in *13th International Conference on emerging Networking EXperiments and Technologies (CoNEXT)*, Incheon, South Korea, Dec. 2017.

[135] D. Steinmetzer, A. Loch, A. García-García, J. Widmer, and M. Hollick, "Mitigating lateral interference: Adaptive beam switching for robust millimeter-wave networks," in *1st ACM Workshop on Millimeter Wave Networks and Sensing Systems (mmNets)*. Snowbird, UT, USA: ACM, Oct. 2017.

[136] D. Steinmetzer, D. Wegemer, and M. Hollick, "A Practical IEEE 802.11ad Research Platform: The Hidden Potential of Off-the-Shelf Devices," in *3rd NSF Millimeter-Wave Research Coordination Network (RCN) Workshop*. Tucson, AZ, USA: NSF, Jan. 2018.

[137] G. Bielsa, J. Palacios, A. Loch, D. Steinmetzer, P. Casari, and J. Widmer, "Indoor Localization Using Commercial Off-The-Shelf 60 GHz Access Points," in *IEEE International Conference on Computer Communications (INFOCOM)*. Honolulu, HI, USA: IEEE, Apr. 2018.

[138] J. Palacios, D. Steinmetzer, A. Loch, M. Hollick, and J. Widmer, "Adaptive Codebook Optimization for Beam-Training on Off-The-Shelf IEEE 802.11ad Devices," in *The 24th Annual International Conference on Mobile Computing and Networking (MobiCom)*, New Delhi, India, Oct. 2018.

[139] D. Steinmetzer, Y. Yuan, and M. Hollick, "Beam-Stealing: Intercepting the Sector Sweep to Launch Man-in-the-Middle Attacks on Wireless IEEE 802.11ad Networks," in *11th Conference on Security and Privacy in Wireless and Mobile Networks (WiSec)*, Stockholm, Sweden, Jun. 2018.

[140] S. K. Saha, H. Assasa, A. Loch, N. M. Prakash, R. Shyamsunder, S. Aggarwal, D. Steinmetzer, D. Koutsonikolas, J. Widmer, and M. Hollick, "Fast and Infuriating: Performance and Pitfalls of 60 GHz WLANs Based on Consumer-Grade Hardware," in *15th International Conference on Sensing, Communication, and Networking (SECON)*, Hong Kong, Jun. 2018.

[141] D. Steinmetzer, M. Stute, and M. Hollick, "TPy: A Lightweight Framework for Agile Distributed Network Experiments," in *12th International Workshop on Wireless Network Testbeds, Experimental evaluation & CHaracterization (WiNTECH)*, New Delhi, India, Oct. 2018.

[142] D. Steinmetzer, S. Ahmad, N. A. Anagnostopoulos, M. Hollick, and S. Katzenbeisser, "Authenticating the Sector Sweep to Protect Against Beam-Stealing Attacks in IEEE 802.11ad Networks," in *2nd Workshop on Millimeter Wave Networks and Sensing Systems (mmNets)*, New Delhi, India, Nov. 2018.

[143] H. Assasa, J. Widmer, J. Wang, T. Ropitault, and N. Golmie, "An Implementation Proposal for IEEE 802.11ay SU/MU-MIMO Communication in Ns-3," in *Workshop on Next-Generation Wireless with ns-3 (WNGW)*, Florence, Italy, Jun. 2019, pp. 26–29.

[144] *IEEE Standard for Information Technology – Telecommunications and Information Exchange Between Systems Local and Metropolitan Area Networks – Specific Requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE P802.11 Std., Rev. IEEE Std 802.11-2012, Dec. 2016.

[145] SIGNET group. [Online]. Available: https://github.com/signetlabdei/ns3-802.11ad-scheduling

[146] C. M. Demichelis and P. Chimento, "IP Packet Delay Variation Metric for IP Performance Metrics (IPPM)," RFC 3393, Nov. 2002.

[147] S. Mohebi, M. Lecci, A. Zanella, and M. Zorzi, "The challenges of Scheduling and Resource Allocation in IEEE 802.11ad/ay," in *18th Mediterranean Communication and Computer Networking Conference (MedComNet)*, Arona, Italy, Jun. 2020.

[148] R. Zazkis and J. Truman, "From Trigonometry to Number Theory... and Back: Extending LCM to Rational Numbers," *Digital Experiences in Mathematics Education*, vol. 1, no. 1, pp. 79–86, Apr. 2015.

[149] R. K. Jain, D. W. Chiu, and W. R. Hawe, "A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Computer Systems," Eastern Research Lab, Tech. Rep. DEC Research Report TR-301, Sep. 1984.

[150] M. Drago, T. Azzino, M. Lecci, A. Zanella, and M. Zorzi, "A Simulation Framework for Contention-Free Scheduling on WiGig," in *submitted to IEEE GLOBECOM*, Dec. 2021.

[151] M. Lecci, M. Drago, A. Zanella, and M. Zorzi, "Exploiting Scheduled Access Features of mmWave WLANs for Periodic Traffic Sources," in *19th Mediterranean Communication and Computer Networking Conference (MedComNet)*, Ibiza, Spain, Jun. 2021.

[152] Huawei Technologies Co., "Empowering consumer-focused immersive VR and AR experiences with mobile broadband," Huawei Technologies Co., White Paper, 2016. [Online]. Available: https://www.huawei.com/en/industry-insights/outlook/mobile-broadband/insights-reports/vr-and-ar

[153] Oculus Business, "Virtual Reality – Set to Enter the Business Mainstream," Oculus Business, White paper, Sep. 2020. [Online]. Available: https://go.facebookinc.com/security-whitepaper.html

[154] Huawei Technologies Co., "AR Insight and Application Practice," Huawei Technologies Co., White Paper, 2021. [Online]. Available: https://carrier.huawei.com/~/media/CNBGV2/download/bws2021/ar-insight-and-application-practice-white-paper-en.pdf

[155] PriceWaterhouseCoopers, "Seeing is believing – How virtual reality and augmented reality are transforming business and the economy," PriceWaterhouseCoopers, Report, 2019. [Online]. Available: https://www.pwc.com/gx/en/technology/publications/assets/how-virtual-reality-and-augmented-reality.pdf

[156] ZTE, "5G Cloud XR Application," ZTE, White paper, 2019. [Online]. Available: https://www.mobile360series.com/wp-content/uploads/2019/09/zte-white-paper.pdf

[157] Qualcomm Technologies, "The Mobile Future of eXtended Reality (XR)," Qualcomm Technologies, Presentation, Nov. 2020. [Online]. Available: https://www.qualcomm.com/media/documents/files/the-mobile-future-of-extended-reality-xr.pdf

[158] Ericsson, "How 5G and Edge Computing can enhance virtual reality," Ericsson, Blog Post, Apr. 2020. [Online]. Available: https://www.ericsson.com/en/blog/2020/4/how-5g-and-edge-computing-can-enhance-virtual-reality

[159] 5G Americas, "5G Services Innovation," 5G Americas, White paper, Nov. 2019. [Online]. Available: https://www.5gamericas.org/wp-content/uploads/2019/11/5G-Services-Innovation-FINAL-1.pdf

[160] Deloitte, "Real learning in a virtual world," Deloitte, Blog Post, Aug. 2018. [Online]. Available: https://www2.deloitte.com/us/en/insights/industry/technology/how-vr-training-learning-can-improve-outcomes.html

[161] F. Chiariotti, "A survey on 360-degree video: Coding, quality of experience and streaming," *Computer Communications*, vol. 177, pp. 133–155, Sep. 2021.

[162] Nokia, "Cloud gaming and 5G – Realizing the opportunity," Nokia, White Paper, 2020. [Online]. Available: https://onestore.nokia.com/asset/207843

[163] Huawei, “Preparing for a Cloud AR/VR Future,” Huawei, White Paper, 2017. [Online]. Available: https://www-file.huawei.com/-/media/corporate/pdf/x-lab/cloud_vr_ar_white_paper_en.pdf?la=en

[164] 3GPP, “Extended Reality (XR) in 5G,” 3rd Generation Partnership Project (3GPP), Technical Report (TR) 26.928, Dec. 2020.

[165] L. J. Hettinger and G. E. Riccio, “Visually induced motion sickness in virtual environments,” *Presence: Teleoperators and Virtual Environments*, vol. 1, no. 3, pp. 306–310, Jan. 1992.

[166] E. L. Groen and J. E. Bos, “Simulator sickness depends on frequency of the simulator motion mismatch: An observation,” *Presence: Teleoperators and Virtual Environments*, vol. 17, no. 6, pp. 584–593, Dec. 2008.

[167] Sebastian von Mammen, Andreas Knote, and Sarah Edenhofer, “Cyber sick but still having fun,” in *ACM Conference on Virtual Reality Software and Technology (VRST)*, Munich, Germany, Nov. 2016.

[168] H. G. Kim, W. J. Baddar, H.-t. Lim, H. Jeong, and Y. M. Ro, “Measurement of exceptional motion in VR video contents for VR sickness assessment using deep convolutional autoencoder,” in *ACM Symposium on Virtual Reality Software and Technology (VRST)*, Gothenburg, Sweden, Nov. 2017.

[169] ITU-T, “Requirements for mobile edge computing enabled content delivery networks,” International Telecommunication Union (ITU), Report, Nov. 2019. [Online]. Available: ITU-TF.743.10

[170] Orange, “XR: 5G extends the boundaries of reality,” Orange, Blog Post, Jul. 2020. [Online]. Available: https://hellofuture.orange.com/en/xr-5g-extends-the-boundaries-of-reality/

[171] Samsung Research, “Samsung 6G Vision,” Samsung Research, White Paper, Jul. 2020. [Online]. Available: https://news.samsung.com/global/samsungs-6g-white-paper-lays-out-the-companys-vision-for-the-next-generation-of-communications-technology

[172] J. Latta and D. Oberg, “A conceptual virtual reality model,” *IEEE Computer Graphics and Applications*, vol. 14, no. 1, pp. 23–29, Jan. 1994.

[173] B. Hentschel, M. Wolter, and T. Kuhlen, “Virtual reality-based multi-view visualization of time-dependent simulation data,” in *IEEE Virtual Reality Conference*, Lafayette, LA, US, Mar. 2009, pp. 253–254.

[174] E. Saad, W. R. J. Funnell, P. G. Kry, and N. M. Ventura, “A virtual-reality system for interacting with three-dimensional models using a haptic device and a head-mounted display,” in *IEEE Life Sciences Conference (LSC)*, Montreal, Canada, Oct. 2018, pp. 191–194.

[175] O. Ergün, Ş. Akin, İ. G. Dino, and E. Surer, “Architectural design in virtual reality and mixed reality environments: A comparative analysis,” in *IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, Osaka, Japan, Mar. 2019, pp. 914–915.

[176] R. Guo, J. Cui, W. Zhao, S. Li, and A. Hao, “Hand-by-Hand Mentor: An AR based Training System for Piano Performance,” in *IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*, Lisbon, Portugal, Mar. 2021, pp. 436–437.

[177] J. Yang, J. Luo, D. Meng, and J.-N. Hwang, “QoE-Driven Resource Allocation Optimized for Uplink Delivery of Delay-Sensitive VR Video Over Cellular Network,” *IEEE Access*, vol. 7, pp. 60 672–60 683, May 2019.

[178] L. Teng, G. Zhai, Y. Wu, X. Min, W. Zhang, Z. Ding, and C. Xiao, “QoE Driven VR 360° Video Massive MIMO Transmission,” *IEEE Transactions on Wireless Communications*, Jul. 2021, Early Access.

[179] C. Perfecto, M. S. Elbamby, J. D. Ser, and M. Bennis, “Taming the Latency in Multi-User VR 360°: A QoE-Aware Deep Learning-Aided Multicast Framework,” *IEEE Transactions on Communications*, vol. 68, no. 4, pp. 2491–2508, Apr. 2020.

[180] B. Krogfoss, J. Duran, P. Perez, and J. Bouwen, “Quantifying the Value of 5G and Edge Cloud on QoE for AR/VR,” in *12th International Conference on Quality of Multimedia Experience (QoMEX)*, Athlone, Ireland, May 2020.

[181] X. Liu, Y. Huang, L. Song, R. Xie, and X. Yang, “The SJTU UHD 360° Immersive Video Sequence Dataset,” in *International Conference on Virtual Reality and Visualization (ICVRV)*, Zhengzhou, China, May 2017, pp. 400–401.

[182] W. Cai, R. Shea, C.-Y. Huang, K.-T. Chen, J. Liu, V. C. M. Leung, and C.-H. Hsu, “A survey on cloud gaming: Future of computer games,” *IEEE Access*, vol. 4, pp. 7605–7620, Aug. 2016.

[183] Z. Xue, D. Wu, J. He, X. Hei, and Y. Liu, “Playing high-end video games in the cloud: A measurement study,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 25, no. 12, pp. 2013–2025, Dec. 2015.

[184] M. Claypool, D. Finkel, A. Grant, and M. Solano, “On the performance of OnLive thin client games,” *Multimedia Syst.*, vol. 20, no. 5, pp. 471–484, Oct. 2014.

[185] M. Manzano, M. Urueña, M. Sužnjević, E. Calle, J. A. Hernández, and M. Matijasevic, “Dissecting the protocol and network traffic of the OnLive cloud gaming platform,” *Multimedia Syst.*, vol. 20, no. 5, pp. 451–470, Oct. 2014.

[186] W.-C. Lo, C.-L. Fan, J. Lee, C.-Y. Huang, K.-T. Chen, and C.-H. Hsu, “360° video viewing dataset in head-mounted virtual reality,” in *8th ACM Multimedia Systems Conference (MMSys)*, Taipei, Taiwan, Jun. 2017, pp. 211–216.

[187] M. Lecci, A. Zanella, and M. Zorzi, "An ns-3 Implementation of a Bursty Traffic Framework for Virtual Reality Sources," in *ACM Workshop on ns-3 (WNS3)*, Online conference, Jun. 2021.

[188] ITU-T, "H.264 : Advanced video coding for generic audiovisual services," ITU-T Telecommunication Standardization Sector of ITU, Tech. Rep., Jun. 2019.

[189] I. M. Chakravarti, R. G. Laha, and J. Roy, *Handbook of Methods of Applied Statistics.* John Wiley and Sons, 1967, vol. 1.

[190] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, Mar. 2020.

[191] W. Shaw, "Sampling Student's T distribution – Use of the inverse cumulative distribution function," *Journal of Computational Finance*, vol. 9, no. 4, pp. 37–73, Jan. 2006.

[192] M. Lecci, "Implementation of the VR Application Model," 2021. [Online]. Available: https://github.com/signetlabdei/ns-3-vr-app

[193] I. E. Richardson, *The H.264 Advanced Video Compression Standard*, 2nd ed. Wiley Publishing, 2010.

[194] ITU-T, "H.265 : High efficiency video coding," ITU-T Telecommunication Standardization Sector of ITU, Tech. Rep., Nov. 2019.

[195] M. Lecci, M. Drago, A. Zanella, and M. Zorzi, "An Open Framework for Analyzing and Modeling XR Network Traffic," *IEEE Access*, vol. 9, pp. 129 782–129 795, Sep. 2021.

List of Publications

Journals

J1. **M. Lecci**, P. Testolina, M. Polese, M. Giordani, M. Zorzi, "Accuracy vs. Complexity for mmWave Ray-Tracing: A Full Stack Perspective," accepted for publication in *IEEE Transactions on Wireless Communications*, Jun. 2021.
Early access available, DOI: 10.1109/TWC.2021.3088349.

J2. **M. Lecci**, M. Drago, A. Zanella, M. Zorzi, "An Open Framework for Analyzing and Modeling XR Network Traffic," in *IEEE Access*, vol. 9, pp. 129782-129795, 2021.
DOI: 10.1109/ACCESS.2021.3113162.

Conference Proceedings

C1. T. Zugno, M. Polese, **M. Lecci**, M. Zorzi, "Simulation of Next-generation Cellular Networks with ns-3: Open Challenges and New Directions," in *ACM Workshop on Next-Generation Wireless with ns-3 (WNGW)*, Florence, Italy, Jun. 2019.
DOI: 10.1145/3337941.3337951.

C2. P. Testolina, **M. Lecci**, M. Rebato, A. Testolin, J. Gambini, R. Flamini, C. Mazzucco, and M. Zorzi, "Enabling Simulation-Based Optimization Through Machine Learning," in *IEEE GLOBECOM – Wireless Communication Symposium*, Waikoloa, HI, USA, Dec. 2019.
DOI: 10.1109/GLOBECOM38437.2019.9013240.

C3. P. Testolina, **M. Lecci**, M. Polese, M. Giordani, M. Zorzi, "Scalable and Accurate Modeling of the Millimeter Wave Channel," in *IEEE International Conference on Computing, Networking and Communications (ICNC)*, Big Island, HI, USA, Feb. 2020.
DOI: 10.1109/ICNC47757.2020.9049746.

C4. **M. Lecci**, P. Testolina, M. Giordani, M. Polese, T. Ropitault, C. Gentile, N. Varshney, A. Bodi, M. Zorzi, "Simplified Ray Tracing for the Millimeter Wave Channel: A Performance Evaluation," in *Information Theory and Applications Workshop (ITA)*, San Diego, CA, USA, Feb. 2020.
DOI: 10.1109/ITA50056.2020.9244950.

C5. **M. Lecci**, P. Testolina, M. Rebato, A. Testolin, and M. Zorzi, "Machine Learning-aided Design of Thinned Antenna Arrays for Optimized Network Level Performance," in *14th European Conference on Antennas and Propagation (EuCAP)*, Copenhagen, Denmark,

Mar. 2020.
DOI: 10.23919/EuCAP48036.2020.9135310.

C6. S. Mohebi, **M. Lecci**, A. Zanella, M. Zorzi, "The Challenges of Scheduling and Resource Allocation in IEEE 802.11ad/ay," in *18th Mediterranean Communication and Computer Networking Conference (MedComNet)*, Arona, Italy, Jun. 2020.
DOI: 10.1109/MedComNet49392.2020.9191491.

C7. **M. Lecci**, M. Polese, C. Lai, J. Wang, C. Gentile, N. Golmie, M. Zorzi, "Quasi-Deterministic Channel Model for mmWaves: Mathematical Formalization and Validation," in *IEEE GLOBECOM – Wireless Communications Symposium* Taipei, Taiwan, Dec. 2020.
DOI: 10.1109/GLOBECOM42002.2020.9322374.

C8. **M. Lecci**, A. Zanella, M. Zorzi, "An ns-3 Implementation of a Bursty Traffic Framework for Virtual Reality Sources," in *ACM Workshop on ns-3 (WNS3)*, Virtual Event, Jun. 2021.
DOI: 10.1145/3460797.3460807.

C9. **M. Lecci**, T. Zugno, S. Zampato, M. Zorzi, "A Full-Stack Open-Source Framework for Antenna and Beamforming Evaluation in mmWave 5G NR," in *IEEE International Conference on Communications (ICC)*, Montreal, QC, Canada, Jun. 2021.
DOI: 10.1109/ICC42927.2021.9500635.

C10. **M. Lecci**, M. Drago, A. Zanella, M. Zorzi, "Exploiting Scheduled Access Features of mmWave WLANs for Periodic Traffic Sources," in *19th Mediterranean Communication and Computer Networking Conference (MedComNet)*, Ibiza, Spain, Jun. 2021.
DOI: 10.1109/MedComNet52149.2021.9501236.

C11. M. Drago, T. Azzino, **M. Lecci**, A. Zanella, M. Zorzi, "A Simulation Framework for Contention-Free Scheduling on WiGig," submitted to *IEEE GLOBECOM – Communication QoS, Reliability & Modeling Symposium*, Dec. 2021.

Book Chapters

B1. F. Pase, F. Mason, P. Testolina, **M. Lecci**, A. Zanella, M. Zorzi, "Smart Data Gathering for Network Optimization," in *Machine Learning and 5G/6G Networks: Interplay and Synergies*, S. Barbarossa, A. Zanella, Texmat, Jul. 2021, pp. 147–170.