PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA
الجمهورية الجزائرية الديمقراطية الشعبية
MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH
وزارة التعليم العالي والبحث العلمي
École supérieure en Sciences et Technologies de l'Informatique et du Numérique
المدرسة العليا في علوم وتكنولوجيات الإعلام الآلي بجاية



Dissertation Submitted to the Department of 2nd Cycle Training in Partial Fulfillment of the Requirements for Master's Degree in Computer Science

Specialty: Cybersecurity

Submitted By:
Rahma Mihoub

Data-Driven Automatic Calibration of IoT Network Simulators: A Literature Review

Supervised by:

Dr. Samir Si-Mohammed
Université de Lorraine

Dr. Maroua Daoudi
ESTIN

Members of jury:

• **Dr.** Boughaba Houda	President	ESTIN
• **Dr.** Zedek Rezika	Examiner	ESTIN
• **Dr.** Bechar Amine	Examiner	ESTIN

Academic year: 2025/2026

Dedication

سُبْحَانَكَ لَا عِلْمَ لَنَا إِلَّا مَا عَلَّمْتَنَا إِنَّكَ أَنتَ الْعَلِيمُ الْحَكِيمُ

(Glory be to You; we have no knowledge except what You have taught us. Indeed, it is You who is the Knowing, the Wise. – Qur'an 2:32)

To Allah the Almighty, for the countless blessings, guidance, and strength granted to me throughout this journey and beyond. I pray that this work may be a source of beneficial knowledge to others.

To my family, who believed in me and saw my potential since I was a child. Without you, I would not be who I am today. Thank you for making me feel special since day one, and for the immense trust you have placed in me.

To my father, for the immense sacrifices you made to raise us, for your tenderness, and for your unwavering trust in me. You have always been my role model in determination and perseverance, a spirit I am proud to have inherited from you.

To my mother, the kindest soul I know, for your unconditional love, warmth, and endless guidance, and for being not only the best mother, but also my best friend.

Acknowledgements

I would like to take a moment to express my deepest gratitude to all those who contributed, in one way or another, to the completion of this work. Their guidance, patience, encouragement, and trust have been a constant source of strength, allowing me to overcome the difficulties encountered throughout this academic journey.

I would like to thank the members of the jury for accepting to evaluate this work and for the time and attention dedicated to reviewing it.

I would also like to thank my school ESTIN for giving me this opportunity, and Université de Lorraine for hosting me throughout this internship.

My deepest thanks go to my supervisor at CRAN, Dr. Samir Si-Mohammed. No words can fully express how thankful I am for your trust in me and for so generously dedicating your time and energy to this work. Thank you for your continuous guidance, both professional and personal, and for making sure I got the most out of this experience. I am especially grateful for the countless opportunities you offered me. Thank you for always pushing me to go beyond my limits, for your constant words of encouragement, and simply for being the greatest supervisor I could have hoped for.

I would also like to thank Dr. Maroua Daoudi, my supervisor at ESTIN, for this opportunity, for her recommendation, and for believing in me. Thank you for all your efforts and guidance, without which this internship would not have been possible.

My thanks also go to the entire CRAN team, who welcomed me with open arms and included me at "the Cool Seniors' Table": Mohamed, Anouar, Jana, Carin, Chahrazed, Abdessamed, Zaid, and others. A special thanks goes to Ghada, for her kindness and support throughout this experience. Thank you all for your humor, and the well-deserved long lunch breaks we shared, it was a genuine pleasure getting to know you, even for such a short time.

I would also like to thank Dr. Fabrice Theoleyre for his invitation, and all the wonderful Network Team of the ICube laboratory for their warm welcome during my invited seminar there.

On a more personal note, I would like to thank my sister, Naila, my rock and my childhood friend. Thank you for listening to hours of me rambling and for keeping up with my endless stories. Thank you for always reminding me of who I am and for setting the bar high, encouraging me to aim further in every aspect of life.

To my little brother, Mohammed Amine, my favorite partner in questionable jokes and endless laughter, thank you for always being there to make life a little lighter. I am truly grateful to have grown up alongside you and to call you my brother.

To my partners throughout this journey, Houda and Aya, thank you for making ESTIN feel like a second home. Thank you for your continuous support, your time, and your genuine company, and to everyone else who made this college experience unforgettable.

To my lifelong friend and sister, Tasnim, thank you for your unconditional support and for believing in my potential since our teenage years. And to your family, who has always treated me as one of their own.

Finally, I extend my sincere thanks to everyone who, directly or indirectly, contributed to the completion of this work.

Abstract

Network simulators are widely used in Internet of Things (IoT) research and deployment planning, yet a significant gap often exists between simulated and real-world network behavior. This gap stems from the use of default radio model parameters that fail to capture real propagation conditions, such as signal fluctuations, interference, and channel variability. Closing this gap requires automatically calibrating a simulator's radio model to reflect real deployment conditions. This Master's report presents a state-of-the-art review of the theoretical foundations required to build such a calibration system. It covers Wireless Sensor Network (WSN)/IoT fundamentals and network simulation tools, radio propagation modeling, and existing simulator calibration approaches, ranging from empirical/manual methods to data-driven and Machine Learning (ML)-based inverse models, one of which frames calibration as an inverse problem: given Key Performance Indicators (KPIs) observed on a real network (e.g., packet reception ratio, Received Signal Strength Indicator (RSSI)), infer the radio model parameters that make the simulator reproduce the same behavior. It further reviews adaptive calibration strategies capable of detecting condition changes (interference, battery drain, environmental variability) and triggering recalibration to keep a simulator, framed as a network digital twin, synchronized with a real deployment over time.

Keywords: Network Digital Twin, Radio Model Calibration, Machine Learning, Adaptive Calibration, Cooja, FIT IoT-LAB.

Résumé

Les simulateurs de réseaux sont largement utilisés dans la recherche sur l'Internet des objets (IoT) et la planification des déploiements, mais un écart important existe souvent entre le comportement simulé et le comportement réel du réseau. Cet écart provient de l'utilisation de paramètres par défaut des modèles radio qui ne parviennent pas à capturer les conditions de propagation réelles, telles que les fluctuations du signal, les interférences et la variabilité du canal. Combler cet écart nécessite de calibrer automatiquement le modèle radio d'un simulateur pour refléter les conditions réelles de déploiement. Ce mémoire de Master présente une revue de l'état de l'art des fondements théoriques nécessaires à la construction d'un tel système de calibration. Il couvre les fondamentaux des réseaux de capteurs sans fil (WSN)/IoT et les outils de simulation réseau, la modélisation de la propagation radio, ainsi que les approches existantes de calibration de simulateurs, allant des méthodes empiriques/manuelles aux modèles inverses basés sur les données et le Machine Learning (ML), dont l'une formule la calibration comme un problème inverse : étant donné des indicateurs de performance clés (KPI) observés sur un réseau réel (par exemple, le taux de réception des paquets, le RSSI), il s'agit d'inférer les paramètres du modèle radio qui permettent au simulateur de reproduire le même comportement. Il passe également en revue des stratégies de calibration adaptative capables de détecter les changements de conditions (interférences, décharge de batterie, variabilité environnementale) et de déclencher une recalibration afin de maintenir un simulateur, envisagé comme un jumeau numérique de réseau, synchronisé avec un déploiement réel dans le temps.

Mots-clés : Jumeau Numérique de Réseau, Calibration de Modèle Radio, Machine Learning, Calibration Adaptative, Cooja, FIT IoT-LAB.

ملخص

تُستخدم محاكيات الشبكات على نطاق واسع في أبحاث إنترنت الأشياء (IoT) وتخطيط عمليات النشر، إلا أنه غالبًا ما توجد فجوة كبيرة بين سلوك الشبكة المحاكى وسلوكها الحقيقي. وتنشأ هذه الفجوة عن استخدام معلمات افتراضية لنماذج الإشارة الراديوية لا تعكس ظروف الانتشار الحقيقية، مثل تقلبات الإشارة والتداخل وتغيّر القناة. ويتطلب سد هذه الفجوة معايرة النموذج الراديوي للمحاكي تلقائيًا ليعكس ظروف النشر الحقيقية. يقدم هذا التقرير الماجستير مراجعة لأحدث ما توصلت إليه الأبحاث فيما يتعلق بالأسس النظرية اللازمة لبناء نظام معايرة من هذا النوع. ويغطي أساسيات شبكات الاستشعار اللاسلكية (WSN) وإنترنت الأشياء وأدوات محاكاة الشبكات، ونمذجة انتشار الإشارة الراديوية، إضافة إلى أساليب معايرة المحاكيات الحالية، بدءًا من الطرق التجريبية/اليدوية وصولًا إلى النماذج العكسية المعتمدة على البيانات والتعلم الآلي (ML)، والتي تصوغ إحداها المعايرة كمسألة عكسية: بالنظر إلى مؤشرات الأداء الرئيسية (KPI) المُلاحظة على شبكة حقيقية (على سبيل المثال، معدل استقبال الحزم، RSSI)، يتم استنتاج معلمات النموذج الراديوي التي تجعل المحاكي يُعيد إنتاج السلوك نفسه. كما يستعرض استراتيجيات المعايرة التكيفية القادرة على كشف تغيّرات الظروف (التداخل، استنزاف البطارية، التغيّرات البيئية) وتفعيل إعادة المعايرة للحفاظ على تزامن المحاكي، المُتصوَّر كتوأم رقمي للشبكة، مع النشر الحقيقي بمرور الوقت.

الكلمات المفتاحية: التوأم الرقمي للشبكة، معايرة النموذج الراديوي، Machine Learning، المعايرة التكيفية، Cooja، FIT IoT-LAB.

Contents

General Introduction **1**

1 Wireless Sensor Networks and Network Simulation **4**
1.1 Introduction . . . 4
1.2 WSN and IoT Fundamentals . . . 4
1.3 Simulators, Emulators, and Testbeds . . . 6
1.4 Cooja and Contiki-NG . . . 7
1.4.1 Motes . . . 7
1.4.2 Radio mediums . . . 8
1.5 Physical Testbeds: FIT IoT-LAB . . . 9
1.6 On the Simulation-to-Reality Gap . . . 10
1.7 Conclusion . . . 12

2 Radio Propagation Models **13**
2.1 Introduction . . . 13
2.2 Physical Layer Basics . . . 13
2.3 Propagation Models in the Literature . . . 16
2.4 Radio Models in Network Simulators . . . 16
2.5 Limitations of Default/Static Radio Model Parameters . . . 18
2.6 Conclusion . . . 18

3 Calibration of Network Simulators **20**
3.1 Introduction . . . 20
3.2 What Is Simulator Calibration? . . . 20
3.3 Manual and Empirical Calibration Approaches . . . 20
3.4 Calibration as an Inverse Problem . . . 21
3.5 Machine Learning for Calibration . . . 21
3.6 Toward Adaptive Calibration . . . 23
3.7 Conclusion . . . 24

4 Literature Review **25**
4.1 Introduction . . . 25
4.2 Low-Power Wireless Channel Modeling . . . 25
4.3 Machine-Learning-Based Radio Propagation Calibration . . . 26
4.4 Cooja Validated Against Physical Testbeds . . . 27

4.5 Data-Driven Digital Twins 28
4.6 Synthesis and research gap 29
4.7 Conclusion 31

General Conclusion **33**

List of Figures

1.1 The lightweight WSN/IoT protocol stack used in this work. Arrows show data flowing down through successive encapsulation before radio transmission. . . . 5
1.2 The Contiki-NG logo. . . . 7
1.3 The FIT IoT-LAB and SLICES-RI logos. . . . 9
1.4 The FIT IoT-LAB M3 open node (FIT IoT-LAB M3 Board Documentation, 2024). 9
1.5 Floor plan and node distribution at the FIT IoT-LAB Grenoble site (FIT IoT-LAB Grenoble Deployment, 2024). . . . 10
1.6 M3 nodes installed under the raised floor tiles at the FIT IoT-LAB Grenoble site (FIT IoT-LAB Grenoble Deployment, 2024). . . . 11

2.1 LogisticLoss's packet reception probability as a logistic function of RSSI, using the default RSSI Inflection Point (-92 dBm). . . . 17

3.1 Calibration reframed as an inverse problem: the forward model (Cooja) generates synthetic (parameter, KPI) training pairs used to train an ML regressor, which is then applied to real, measured KPIs to estimate the corresponding radio model parameters. . . . 22
3.2 Adeogun's calibration pipeline: the network is trained on synthetic data and then applied to real, unlabeled measurements to estimate the corresponding parameters, adapted from Adeogun (2019). . . . 22

4.1 Scopus search showing no results for the combination of Cooja and radio model calibration. . . . 31

List of Tables

1.1 Selected Contiki-NG/Cooja mote platforms 8
1.2 Comparison of the CC2420 (Z1/Cooja) and AT86RF231 (M3/FIT IoT-LAB) radio chips 11

2.1 LogisticLoss configurable parameters and Cooja default values 18

4.1 Synthesis of reviewed related work 30

List of Abbreviations

6LoWPAN IPv6 over Low-Power Wireless Personal Area Networks.

ADWIN Adaptive Windowing.

API Application Programming Interface.

ARM Advanced RISC Machine.

AWGN Additive White Gaussian Noise.

CNRS Centre National de la Recherche Scientifique.

CoAP Constrained Application Protocol.

CRAN Centre de Recherche en Automatique de Nancy.

CUSUM Cumulative Sum.

DDM Drift Detection Method.

DGRM Directed Graph Radio Medium.

DSSS Direct Sequence Spread Spectrum.

EDDM Early Drift Detection Method.

IEEE Institute of Electrical and Electronics Engineers.

IoT Internet of Things.

IPv6 Internet Protocol version 6.

ISM Industrial, Scientific, and Medical (radio band).

KPI Key Performance Indicator.

MCU Microcontroller Unit.

ML Machine Learning.

MLP Multi-Layer Perceptron.

O-QPSK Offset Quadrature Phase-Shift Keying.

OFDM Orthogonal Frequency-Division Multiplexing.

PH Page-Hinkley.

PLE Path Loss Exponent.

PRR Packet Reception Ratio.

RAM Random-Access Memory.

RF Radio Frequency.

RPL Routing Protocol for Low-Power and Lossy Networks.

RSSI Received Signal Strength Indicator.

RX Receive.

SNR Signal-to-Noise Ratio.

SPI Serial Peripheral Interface.

TX Transmit.

UDGM Unit Disk Graph Medium.

WSN Wireless Sensor Network.

General Introduction

Wireless Sensor Networks (WSNs) and the Internet of Things (IoT) connect large numbers of resource-constrained devices over wireless links, with deployments ranging from home automation to industrial monitoring. These devices are controlled by *firmware*, a type of software designed to provide low-level control of hardware and typically stored in non-volatile memory (ISO/IEC/IEEE, 2017). In resource-constrained WSN/IoT nodes, the firmware executes on the device's microcontroller and may implement hardware drivers, such as those for the radio and sensors, as well as the communication protocol stack. Designing, testing, and validating such firmware directly on physical hardware is slow, costly, and difficult to reproduce at scale, making network simulation and emulation an important step in the WSN/IoT development pipeline.

One commonly used tool for this purpose is Cooja, the network simulator associated with the Contiki-NG operating system (Contiki-NG Cooja Simulator, 2024; Contiki-NG Documentation, 2024). Cooja enables the simulation of low-power IEEE 802.15.4-based WSN/IoT networks running the Contiki-NG protocol stack, including 6LoWPAN, RPL, and CoAP. It allows unmodified firmware to be executed on virtual representations of target motes while the surrounding radio channel can be configured using different radio propagation models (Contiki-NG Cooja Simulator, 2024).

Despite executing the same compiled firmware that would run on real hardware, Cooja simulations do not automatically reproduce the behavior observed on a physical deployment. As argued throughout this report, this simulation-to-reality gap is best understood as a *radio propagation* fidelity problem rather than a hardware discrepancy. The IEEE 802.15.4 stack separates the physical (PHY) layer, responsible for the actual transmission, reception, and propagation of radio signals, from the medium access control (MAC) layer, responsible for coordinating channel access between motes (e.g., framing, collision avoidance) (IEEE, 2020). Cooja executes the exact same compiled firmware, including the same MAC-layer logic, on hardware close enough to the physical motes being modeled (Contiki-NG Cooja Simulator, 2024). Any residual divergence between simulated and real KPIs can therefore be attributed to the PHY layer rather than the MAC layer, specifically, to how well the simulator's radio propagation model represents real propagation conditions. Among Cooja's several configurable radio propagation models, this report focuses on LogisticLoss, the empirical model used throughout this work to relate distance and signal strength to packet reception.

Formally, a KPI observed on a network can be expressed as a function of the radio model and its parameters,

$$\text{KPI} = f(\text{model}, \theta),$$

where θ denotes the model's free parameters. LogisticLoss's parameters, the Path Loss Exponent, the Received Signal Strength Indicator (RSSI) Inflection Point, and the AWGN sigma, are shipped with generic default values intended to approximate typical indoor 2.4 GHz conditions, rather than any specific real deployment. Closing the simulation-to-reality gap therefore requires calibrating these parameters against real measurements, and, since real deployments are not static, ideally doing so continuously as conditions change over time.

This Master's report undertakes a state-of-the-art review of the theoretical foundations required to address this calibration problem. Its scope is deliberately limited to the literature and background theory: it does not present new experimental results or implementation details, which are addressed separately. Specifically, this report aims to:

- Introduce the WSN/IoT context, the distinction between simulators, emulators, and physical testbeds, and the specific tools relevant to this work (Cooja/Contiki-NG and the FIT IoT-LAB (Adjih et al., 2015) testbed);
- review the physical phenomena underlying radio wave propagation and the mathematical models used to describe them, with particular attention to the log-distance/log-normal shadowing model and its implementation as Cooja's LogisticLoss radio medium;
- review how a radio propagation model's parameters can be calibrated, from manual measurement-based approaches to machine-learning-based methods framed as an inverse problem, and how such calibration can be made adaptive to changing conditions through concept-drift detection;
- survey the closest related work in the literature and identify, precisely, what gap remains unaddressed.

The remainder of this report is structured into four chapters:

- **Chapter 1** introduces WSN/IoT fundamentals, situates simulators, emulators, and physical testbeds relative to one another, and presents Cooja/Contiki-NG and FIT IoT-LAB, the simulation and physical platforms used throughout this work. The chapter closes with the argument that the simulation-to-reality gap between them is a question of radio model fidelity rather than hardware mismatch.
- **Chapter 2** reviews the physical phenomena affecting radio wave propagation, situates propagation models within a deterministic/statistical/empirical taxonomy, and details the exact formula and free parameters of Cooja's chosen radio medium.
- **Chapter 3** reviews how a radio propagation model's parameters can be calibrated: from manual, measurement-based approaches, through an inverse-problem framing, to machine-learning-based methods, and finally to adaptive calibration via concept-drift detection.

- **Chapter 4** surveys the closest related work in the literature and synthesizes the gap this report's calibration approach addresses.

The report closes with a general conclusion summarizing these findings.

Chapter 1

Wireless Sensor Networks and Network Simulation

1.1 Introduction

Wireless Sensor Networks (WSNs) form the physical backbone of most Internet of Things (IoT) deployments, connecting resource-constrained devices over wireless links. Following on from the General Introduction, this chapter introduces the concepts needed to situate the calibration problem addressed later in this report. It first reviews WSN/IoT fundamentals and where network simulation fits in the development workflow, then presents Cooja and Contiki-NG, the simulation environment used in this work, together with the FIT IoT-LAB physical testbed used for validation. It closes by discussing what the simulation-to-reality gap actually reflects, and why closing it is a modeling problem rather than a hardware problem.

1.2 WSN and IoT Fundamentals

A WSN is composed of battery-powered, memory- and compute-constrained devices that sense their environment and relay data, sometimes over multi-hop wireless links, toward one or more collection points. IoT deployments extend this model by connecting such networks to the wider Internet, adding requirements around addressing, interoperability, and remote management that a WSN alone does not need to satisfy.

To meet these requirements under tight energy and memory budgets, IoT devices commonly rely on a lightweight protocol stack rather than the standard Internet stack. Figure 1.1 shows this stack: data originates at the application layer and is successively encapsulated as it moves down through the routing and adaptation layers, before finally being transmitted over the radio at the physical layer.

At the application layer, protocols such as CoAP (Shelby et al., 2014) provide a lightweight, RESTful alternative to HTTP suited to constrained devices. Below it, routing is typically handled by the Routing Protocol for Low-Power and Lossy Networks (RPL) (Winter et al., 2012), which builds and maintains a destination-oriented routing topology adapted to unreliable, energy-constrained links. Underneath, 6LoWPAN (Kushalnagar et al., 2007) adapts IPv6 to the small

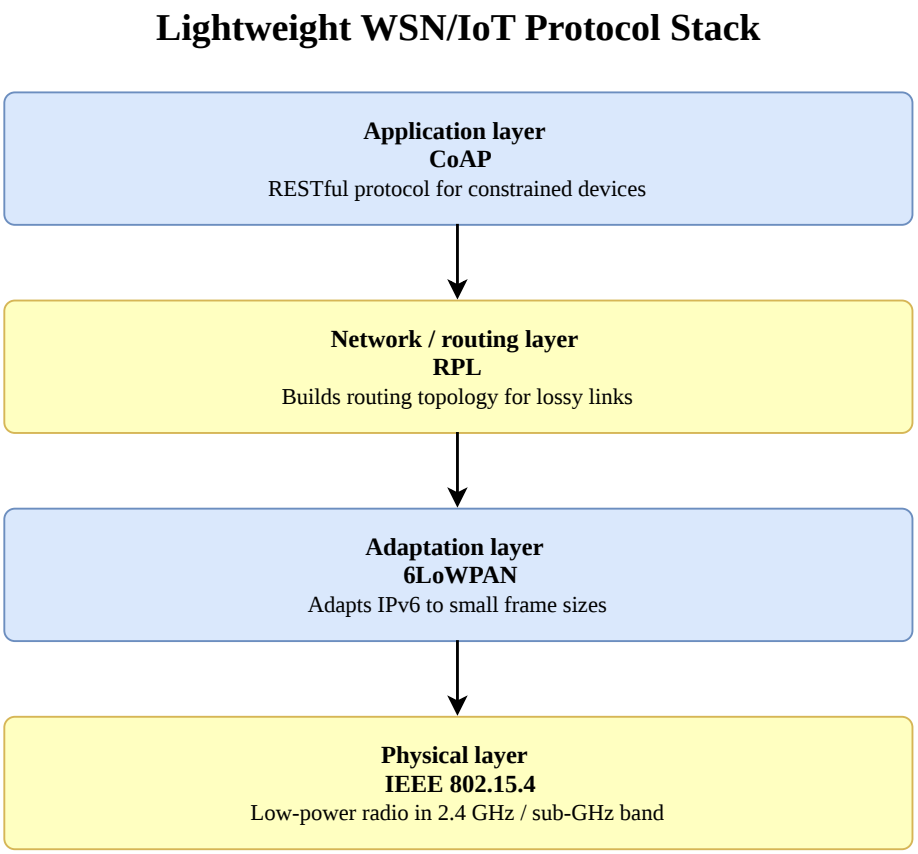


Figure 1.1: The lightweight WSN/IoT protocol stack used in this work. Arrows show data flowing down through successive encapsulation before radio transmission.

frame sizes and limited memory of constrained devices, allowing them to be addressed as first-class Internet hosts. Finally, at the physical layer, IEEE 802.15.4 (IEEE, 2020) defines the low-power radio communication, in the 2.4 GHz or sub-GHz band, over which the resulting frames are actually transmitted. This report does not examine the upper three layers (CoAP, RPL, 6LoWPAN) in depth, since the calibration problem addressed here operates at this physical/radio layer; they are introduced here only to situate where the KPIs used later (e.g., packet reception ratio, RSSI) are observed within the broader stack.

Why “motes” and not “nodes”?

The terms *node* and *mote* are related but are not fully interchangeable in the wireless-sensor-network literature. A “node” is a general networking abstraction: it denotes an entity that participates in a network and may represent a sensor device, router, server, or general-purpose computer. The term is therefore largely hardware agnostic.

A “mote” is a more specific term commonly used for a small, resource-constrained wireless sensor platform. Such a platform typically combines a low-power microcontroller, limited memory, a radio transceiver, and one or more sensors, while operating under strict energy and computational constraints (Levis et al., 2005). In this sense, a mote is a particular type of network node rather than a separate networking concept.

The term is historically associated with the Smart Dust research program at the University of California, Berkeley. Warneke et al. (2001) described Smart Dust as an autonomous sensing, computing, and communication system that could be packaged into a “cubic-millimeter mote”. This work helped establish the use of *mote* to describe miniature, self-contained sensing and communication platforms. The terminology was subsequently adopted in the TinyOS and Contiki ecosystems, both of which targeted small, resource-constrained wireless sensors rather than general-purpose computers (Dunkels et al., 2004; Levis et al., 2005).

The distinction is also useful when discussing Contiki and Cooja. Contiki is an operating system designed for small networked sensors (Dunkels et al., 2004), whereas Cooja is its simulation environment. Within Cooja, a simulated device represents a particular mote type and its corresponding hardware and firmware configuration. Thus, platforms such as the Z1, M3, and Tmote Sky are appropriately described as *motes* when the discussion concerns their specific hardware, software, memory, radio, or sensor characteristics. The term *node* is retained for hardware-independent discussions, such as network topology, routing, addressing, and graph structure.

Accordingly, this report uses “mote” when referring to a specific constrained sensing platform and “node” when referring to a network entity in the general or abstract sense. A mote is therefore treated as a specialized kind of node, but the two terms are not used interchangeably when the physical characteristics of the device are relevant.

1.3 Simulators, Emulators, and Testbeds

Three complementary approaches exist for evaluating WSN/IoT protocols and applications before field deployment.

Simulators model network behavior mathematically, executing an abstract representation of motes, the radio channel, and the protocol stack. They are somehow fast, fully reproducible, and allow large-scale topologies to be tested without any physical hardware, but their fidelity is only as good as the underlying models, particularly the radio propagation model, which is the focus of this report.

Emulators go a step further by executing the actual compiled firmware, instruction-by-instruction, on a virtual representation of the target microcontroller, while still simulating the radio channel around it. This captures real timing and code-execution behavior that a pure simulator would abstract away, at some computational cost.

Physical testbeds deploy real hardware in a real (indoor or outdoor) radio environment,

capturing propagation effects, multipath fading, interference, environmental variability, that no model fully reproduces. Testbeds provide the ground truth against which simulators and emulators must ultimately be validated, but they are costly to build, harder to reproduce exactly across runs, and difficult to scale to very large topologies.

Cooja occupies the emulator category: it emulates real, unmodified firmware at the instruction level on a virtual microcontroller, while the surrounding radio channel is handled by a separate, pluggable radio medium component (Contiki-NG Cooja Simulator, 2024). This distinction matters for the rest of this report: because Cooja executes the same compiled Contiki-NG firmware that runs on the physical mote, the two setups differ only in how the radio channel is realized, modeled in Cooja, physical on the testbed, rather than in how the application logic itself executes.

1.4 Cooja and Contiki-NG

Cooja is the network simulation/emulation tool bundled with Contiki-NG (Contiki-NG Cooja Simulator, 2024), an open-source operating system designed for constrained IoT devices. Contiki-NG is maintained as an open-source project on GitHub [1] , with an active developer community responsible for the radio mediums, mote platforms, and networking stack examined throughout the rest of this chapter.



Figure 1.2: The Contiki-NG logo.

1.4.1 Motes

Contiki-NG organizes hardware support by platform, and Cooja's mote types correspond directly to these platforms (Contiki-NG Documentation, 2024). Table 1.1 summarizes the mote platforms most relevant to this work, including the Z1 mote used here.

Both CC2420-based motes (Z1, Tmote Sky) share the same radio chip and, consequently, the same IEEE 802.15.4 RF characteristics; they differ mainly in MCU and available RAM/flash, which is why the Z1's 8 KB RAM ceiling matters for firmware-level design choices made later in this work.

[1] github.com/contiki-ng/

Table 1.1: Selected Contiki-NG/Cooja mote platforms

Mote	MCU	RAM	Flash	Radio chip
Z1	MSP430F2617 (16 MHz)	8 KB	92 KB	CC2420
Tmote Sky/TelosB	MSP430F1611 (8 MHz)	10 KB	48 KB	CC2420
Zoul (Firefly/RE-Mote)	CC2538 ARM Cortex-M3	32 KB	512 KB	CC2538 (integrated)
nRF52840	ARM Cortex-M4	256 KB	1 MB	nRF52840 (integrated)

1.4.2 Radio mediums

Cooja's radio mediums [2] are all implemented as subclasses of a common `AbstractRadioMedium` base class, which provides the shared logic for registering motes, propagating signal strength updates, and forwarding radio packets between them; each concrete medium then supplies its own model for how signal, interference, and reception probability behave. The current Contiki-NG/Cooja repository provides the following radio mediums:

- **UDGM (Unit Disk Graph Medium), distance-loss variant**: represents transmission and interference range as circles around each mote, with success probability decreasing with distance within those circles; both radii scale with the radio's configured output power.
- **UDGMConstantLoss**: a simplified variant of UDGM in which any mote within transmission range receives packets with a fixed, distance-independent success probability, rather than one that decays with distance.
- **DirectedGraphMedium (DGRM)**: a medium in which the transmission success rate (and optionally RSSI and propagation delay) for each directed link is specified explicitly by the user rather than derived from mote geometry, useful for reproducing a fixed, pre-measured connectivity graph.
- **SilentRadioMedium** : a minimal medium in which motes do not receive each other's transmissions at all; it serves as a baseline/control configuration rather than a model of a real channel.
- **LogisticLoss**: the model used in this work. It improves on UDGM's distance-based abstraction by modeling packet reception probability as a logistic (sigmoid) function of the received signal strength, and by adding Additive White Gaussian Noise (AWGN) to the simulated signal level, while remaining computationally lightweight and configurable

[2] github.com/contiki-ng/cooja/blob/master/java/org/contikios/cooja/radiomediums/

through a small number of parameters. This combination, a more realistic reception model than UDGM's, together with a small set of interpretable free parameters, is what makes LogisticLoss well suited to the calibration approach this report reviews, as discussed further in Chapter 2.

1.5 Physical Testbeds: FIT IoT-LAB

FIT IoT-LAB is a large-scale, open, physical IoT testbed and a precursor infrastructure to SLICES-FR, the French node of the SLICES-RI European research infrastructure for computing and communication experimentation. Both projects are publicly maintained infrastructures and widely used in research with their own visual identity, shown in Figure 1.3.

Figure 1.3: The FIT IoT-LAB and SLICES-RI logos.

It is distributed across several sites in France, including Grenoble, Lille, Paris (Saclay), Strasbourg, and Rennes, together providing access to several thousand fixed and mobile nodes (Adjih et al., 2015). Rather than a single hardware platform, FIT IoT-LAB hosts several node types side by side: legacy WSN430 nodes (available in both 800 MHz and 2.4 GHz radio variants), M3 nodes, more powerful A8 nodes capable of running embedded Linux, and dedicated host/gateway nodes. Users reserve the number and type of nodes they need for a given experiment through the testbed's web portal or API.

We use in this work the M3 open nodes on the Grenoble site. The M3 node is built around a 32-bit STM32 (ARM Cortex-M3) microcontroller paired with an Atmel/Microchip AT86RF231 IEEE 802.15.4-compliant 2.4 GHz radio transceiver (Atmel Corporation, 2014), the successor generation to the WSN430's radio, offering a more capable RF front-end alongside the more powerful 32-bit MCU. Figure 1.4 shows the physical board.

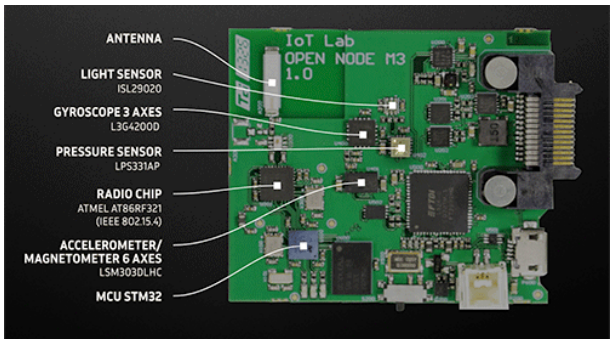


Figure 1.4: The FIT IoT-LAB M3 open node (FIT IoT-LAB M3 Board Documentation, 2024).

Node distribution at the Grenoble site

The Grenoble site is deployed at building scale across the ground floor of the Inria Grenoble Rhône-Alpes building, with nodes fixed under the raised floor tiles and in the dropped ceiling of several corridors (FIT IoT-LAB Grenoble Deployment, 2024). Figure 1.5 shows the resulting floor plan and node distribution: M3 nodes are installed under the floor along corridors J, F1, F2, F3, F4, and G, while A8 nodes are installed in the dropped ceiling above the same corridors; a dedicated room (K) additionally hosts a small number of other boards (Nucleo WL55JC, Microchip SAMR34, Raspberry Pi 3) used for specialized experiments such as LoRa deployments.

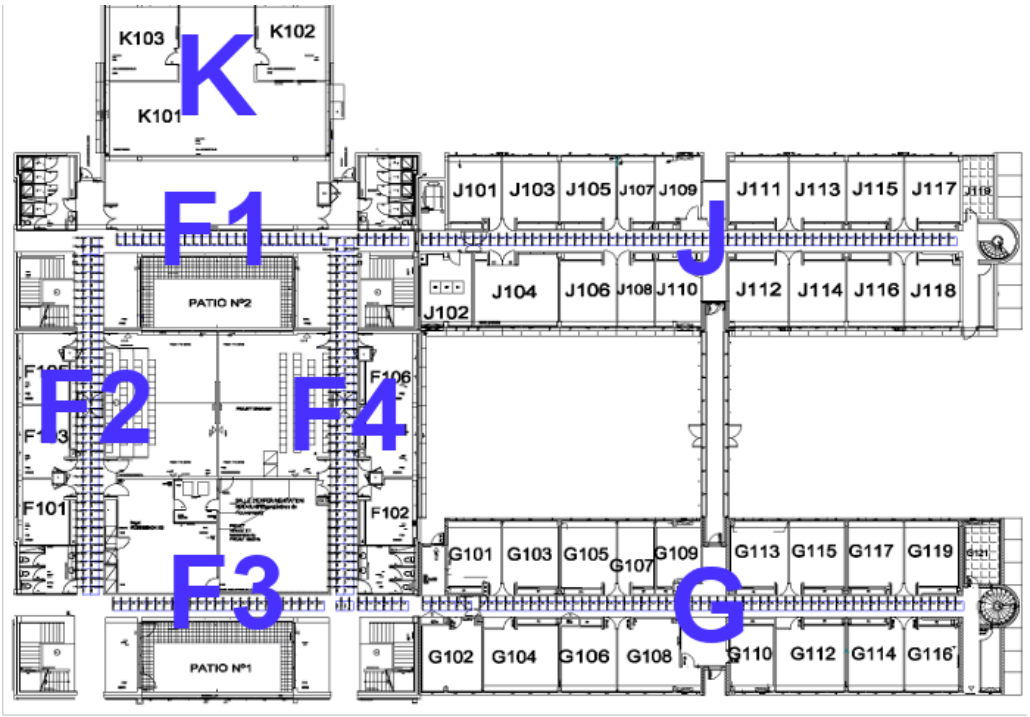


Figure 1.5: Floor plan and node distribution at the FIT IoT-LAB Grenoble site (FIT IoT-LAB Grenoble Deployment, 2024).

Figure 1.6 shows the physical installation underlying this plan: nodes are mounted under the raised floor tiles that run along the corridors, out of sight and out of the way of building occupants, which is representative of how such large-scale WSN testbeds are typically deployed inside an active office building.

1.6 On the Simulation-to-Reality Gap

Table 1.2 compares the two radio chips relevant to this work: the CC2420 used by the Z1 mote in Cooja, and the AT86RF231 used by the M3 node on FIT IoT-LAB.

It is tempting to attribute divergence between Cooja and FIT IoT-LAB results to the fact that the Z1 and M3 motes use different radio chips. Table 1.2, however, shows that the CC2420 and AT86RF231 are both IEEE 802.15.4-compliant 2.4 GHz transceivers, operating at the same

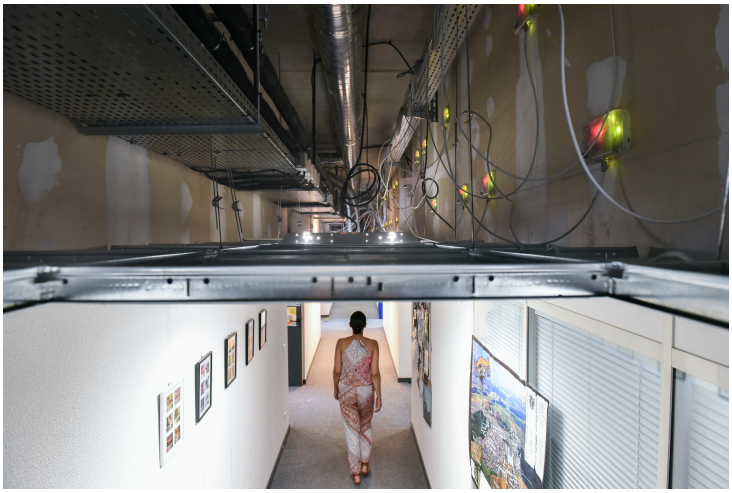


Figure 1.6: M3 nodes installed under the raised floor tiles at the FIT IoT-LAB Grenoble site (FIT IoT-LAB Grenoble Deployment, 2024).

Table 1.2: Comparison of the CC2420 (Z1/Cooja) and AT86RF231 (M3/FIT IoT-LAB) radio chips

Parameter	**CC2420 (Z1)**	**AT86RF231 (M3)**
Standard	IEEE 802.15.4	IEEE 802.15.4
Band	2.4 GHz	2.4 GHz
Data rate	250 kbps	250 kbps (up to 2 Mbps, non-standard mode)
RX sensitivity	-95 dBm	-101 dBm
TX output power	0 dBm (typical)	Programmable, -17 to $+3$ dBm
Modulation	DSSS	O-QPSK
Interface	SPI	SPI

nominal data rate, with sensitivity and output power figures in the same order of magnitude. This is precisely the point of using an emulator such as Cooja rather than requiring identical hardware: Cooja does not aim to reproduce the exact electrical behavior of one specific chip, but to emulate real firmware execution, including the same MAC-layer logic, on a mote whose radio-level characteristics are broadly representative of the class of low-power IEEE 802.15.4 hardware being studied. The hardware is close enough, by design, that it is not the primary variable at play.

Consequently, when KPIs measured in Cooja diverge from those measured on FIT IoT-LAB, the more defensible explanation is not the modest difference between the CC2420 and the AT86RF231, but, consistent with the physical (PHY) versus medium access control (MAC) layer distinction introduced in the General Introduction, a mismatch at the PHY layer: specifically, the fidelity of the *software* radio propagation model standing in for the real channel, LogisticLoss, and the extent to which its free parameters (Path Loss Exponent, RSSI Inflection Point, AWGN

sigma) are set to values that reflect real propagation conditions rather than simulator defaults. Real deployments are also subject to multipath fading, interference from co-located wireless devices, and time-varying channel conditions that a static, uncalibrated model does not capture at all. This framing is what motivates the calibration approach whose theoretical foundations this report reviews: the gap is treated as a PHY-layer model-fidelity problem to be solved by data-driven calibration of the radio medium's parameters, not as a hardware discrepancy to be corrected.

1.7 Conclusion

This chapter establishes the background needed for the rest of this report: the WSN/IoT context in which network simulation is used, the distinction between simulators, emulators, and physical testbeds, and the specific tools used in this work: Cooja/Contiki-NG on the emulation side and the FIT IoT-LAB M3 testbed on the physical side. The radio chip comparison showed that Z1 and M3 hardware are broadly comparable, which reframes the simulation-to-reality gap discussed at the end of the chapter as a question of radio *model* fidelity rather than hardware mismatch. The next chapter examines radio propagation models in more detail, including the LogisticLoss model introduced above and the free parameters that must be calibrated to close this gap.

Chapter 2

Radio Propagation Models

2.1 Introduction

Chapter 1 argued that the simulation-to-reality gap between Cooja and FIT IoT-LAB is best understood as a question of radio *model* fidelity rather than hardware mismatch. This chapter develops the theoretical background needed to make that argument precise. It first introduces the physical phenomena that influence radio propagation, then reviews how these phenomena are represented mathematically in the wireless-networking literature, and finally examines how Cooja's `LogisticLoss` model implements a simplified version of this theory. The exact formula and free parameters of that model are important because they become the targets of calibration in later chapters.

2.2 Physical Layer Basics

Radio propagation is influenced by the operating frequency, the transmitter and receiver characteristics, the distance between the devices, and the physical environment through which the signal travels. For short-range wireless sensor networks, the most important physical effects are path loss, shadowing, reflection, diffraction, scattering, multipath propagation, and small-scale fading (Dezfouli et al., 2015; Rappaport, 2002).

In the 2.4 GHz ISM-band deployments considered in this report, radio signals generally propagate through direct or line-of-sight paths as well as through reflected, diffracted, and scattered paths. Reflections from walls, floors, furniture, equipment, and other objects can create multiple paths between a transmitter and receiver. Diffraction allows part of the signal to reach regions that are not in direct line of sight, while scattering redistributes radio energy when the signal interacts with objects or surfaces whose geometry and material properties disturb the incident wave (Dezfouli et al., 2015; Rappaport, 2002).

Several propagation and link-quality effects are particularly relevant to the constrained, short-range indoor IoT scenarios considered in this report:

- **Path loss**: the systematic reduction in average received power as the transmitter–receiver distance increases. Path loss is the fundamental propagation effect represented in a link budget and in large-scale path-loss models.

- **Shadowing**: additional variation in received power caused by obstacles such as walls, furniture, equipment, and building structures. Shadowing is often represented as a random, location-dependent deviation from the mean path-loss curve.
- **Reflection**: the redirection of radio energy from surfaces or objects. Reflections can produce additional propagation paths and can either strengthen or weaken the received signal depending on the relative phase of the arriving components.
- **Diffraction**: the bending and spreading of radio energy around edges or obstacles. Diffraction can provide partial signal reception even when a direct line-of-sight path is blocked.
- **Scattering**: the redistribution of radio energy caused by irregular objects, rough surfaces, or other structures in the propagation environment.
- **Multipath propagation**: the arrival of multiple copies of the transmitted signal over paths with different lengths, typically because of reflection, diffraction, and scattering.
- **Small-scale fading**: rapid fluctuations in received signal amplitude or power caused by constructive and destructive interference between multipath components.
- **Delay spread**: the difference between the arrival times of significant multipath components. In sufficiently wideband systems, delay spread can cause inter-symbol interference.
- **Noise**: unwanted electrical or electromagnetic energy introduced by the receiver and its surrounding environment. Noise affects the signal-to-noise ratio but is not itself a propagation mechanism.
- **Interference**: unwanted energy from other transmitters or radio systems operating on the same or nearby frequencies. Interference is distinct from path loss and multipath propagation, although it can also reduce packet-reception performance.

These effects are not equally important in every deployment. In the short-range indoor IoT scenarios considered in this report, path loss, shadowing, multipath, diffraction, fading, noise, and interference are more relevant than long-distance ground-wave or ionospheric propagation. This motivates the use of a compact statistical or empirical radio model rather than a full-wave or site-specific electromagnetic simulation. The specific implications for the simulator model considered in this work are discussed in Section 2.4.

Categories of propagation models

Radio-propagation models can be grouped broadly according to the amount of physical detail they represent and the way their parameters are obtained. Three useful categories are deterministic, statistical, and empirical models (Dezfouli et al., 2015; Rappaport, 2002).

Deterministic or site-specific models use a detailed description of the propagation environment to calculate received-signal behavior from specific propagation paths and physical interactions. Ray-tracing models are a representative example. They use the geometry and material properties of the environment to model paths involving line-of-sight transmission, reflection, and diffraction and may also incorporate scattering and penetration, depending on the implementation (Yun & Iskander, 2015). Such models can provide detailed, location-specific predictions, but they generally require accurate environmental information and are computationally more expensive than large-scale empirical models.

Statistical models represent the wireless channel using probability distributions or stochastic processes. Rather than calculating every physical interaction in the environment, they describe quantities such as shadowing, fading, received power, or packet reception using statistical parameters estimated from theory, measurements, or simulated channel realizations (Dezfouli et al., 2015; Rappaport, 2002).

Empirical models fit a relatively small number of parameters to measurements collected in a particular environment. The log-distance path-loss model is a common example. It represents the average dependence of received power on distance through a path-loss exponent and may include a random shadowing term. Its relative simplicity makes it useful for wireless-network simulation and practical calibration, although it does not explicitly represent individual reflection, diffraction, or multipath components (Miranda et al., 2013; Rappaport, 2002).

The general log-distance model used in the wireless-networking literature and Cooja's `LogisticLoss` model are therefore best understood as low-dimensional statistical or empirical abstractions rather than deterministic physical channel models. The precise classification depends on the implementation: a model may contain a deterministic distance-loss component while also using stochastic or empirically fitted terms for shadowing and packet reception. Cooja's implementation should therefore be described according to its actual formula and source-code behavior rather than assigned to a category solely by its name (Contiki-NG LogisticLoss Source, 2024).

This distinction is important for calibration. Adeogun (2019) shows that the parameters of a stochastic radio-propagation model can be estimated from channel data using a machine-learning regressor, specifically a multilayer perceptron trained on synthetically generated channel realizations. This approach avoids requiring an intermediate, manually engineered phase in which individual multipath components are first extracted before model parameters are fitted.

A related calibration perspective is provided by Bharti et al. (2022), who formulate the calibration of stochastic radio channel models as a parameter-estimation problem based on measured channel behavior. Together, these approaches support the calibration strategy adopted in this report: rather than reconstructing the complete physical propagation environment, the free parameters of a compact radio model can be learned from observed measurements or key performance indicators (KPIs). This parameter-level adaptation is the premise on which Chapter 3 builds.

2.3 Propagation Models in the Literature

The most widely used empirical model for path loss as a function of distance is the *log-distance path loss model*. It expresses the path loss $PL(d)$ at distance d as

$$PL(d) = PL(d_0) + 10\,n\,\log_{10}\left(\frac{d}{d_0}\right) + X_\sigma, \tag{2.1}$$

where d_0 is a reference distance, $PL(d_0)$ is the path loss measured at that reference distance, n is the *path loss exponent* governing how quickly the signal decays with distance, and X_σ is a zero-mean Gaussian random variable with standard deviation σ accounting for shadowing, this extension of the bare log-distance model is commonly called *log-normal shadowing*.

Dezfouli et al. (2015) apply this log-normal shadowing model specifically to low-power IEEE 802.15.4 transceivers, deriving path-loss-exponent and shadowing-deviation parameters for chips such as the CC1000 and CC2420 (the latter being the radio used by the Z1 mote in this work) from measured data, and showing how these parameters directly affect the size and shape of a network's connectivity region. This confirms that the log-distance/log-normal family of models is both standard practice and directly applicable to the hardware relevant to this report, which motivates its use as the basis of Cooja's LogisticLoss radio medium, examined next.

2.4 Radio Models in Network Simulators

Chapter 1 introduced Cooja's available radio mediums (UDGM, UDGMConstantLoss, DirectedGraphMedium, SilentRadioMedium, and LogisticLoss). This section details LogisticLoss specifically, since it is the model used throughout this work.

LogisticLoss models packet reception as a two-stage process (Contiki-NG LogisticLoss Source, 2024): first, the path loss between sender and receiver is computed using the log-distance path loss model introduced in Section 2.3; second, the resulting signal level is mapped to a packet reception probability through a logistic (sigmoid) function.

Path loss stage

The path loss at distance d is computed as

$$PL_{\text{dBm}}(d) = PL_0 + 10\,\alpha\,\log_{10}\left(\frac{d}{d_0}\right) + \mathcal{N}(0,\sigma), \tag{2.2}$$

where PL_0 is the receiver sensitivity (the loss at the reference distance d_0), α is the path loss exponent, $\mathcal{N}(0,\sigma)$ is Additive White Gaussian Noise (AWGN) with standard deviation σ, and d_0 is taken to be the mote's configured transmitting range. This is precisely the log-normal shadowing model from Section 2.3, with the reference distance tied to the simulated transmission

range rather than a fixed physical measurement point.

Reception probability stage

Given the resulting signal level (RSSI), the probability of successful packet reception is

$$\mathrm{PRR}(\mathrm{rssi}) = \frac{1}{1 + \exp\big(-(\mathrm{rssi} - \mathrm{rssi}_{50\%})\big)}, \tag{2.3}$$

where $\mathrm{rssi}_{50\%}$ is the RSSI Inflection Point: the signal level at which exactly 50% of packets are received, and the point at which the logistic curve is steepest. Figure 2.1 plots this reception probability curve using the default RSSI Inflection Point.

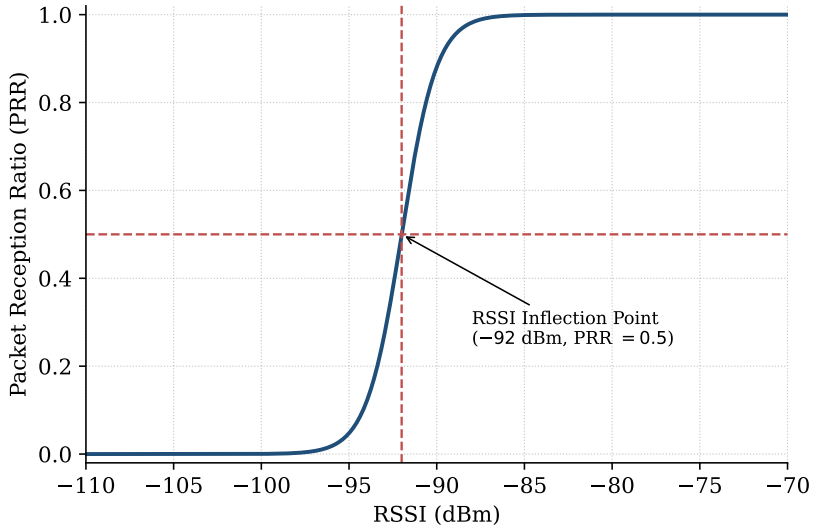


Figure 2.1: LogisticLoss's packet reception probability as a logistic function of RSSI, using the default RSSI Inflection Point (−92 dBm).

Free parameters

Table 2.1 summarizes LogisticLoss's configurable parameters and their default values (Contiki-NG LogisticLoss Source, 2024). Three of these (the Path Loss Exponent, the RSSI Inflection Point, and the AWGN sigma) are the parameters this report's calibration approach targets; the transmitting range is treated as a fixed, empirically determined value per scenario rather than a calibration target (see Chapter 3). This selection is not the only one possible, Rx sensitivity and the transmitting range are also configurable, and which subset of parameters to prioritize as calibration targets is itself a question that was studied rather than assumed; the full justification for this specific selection is beyond the scope of this literature review.

The default values in Table 2.1 are documented as approximating generic "indoor, 2.4 GHz" conditions (Contiki-NG LogisticLoss Source, 2024), rather than any specific measured deploy-

Table 2.1: LogisticLoss configurable parameters and Cooja default values

Parameter	Symbol	Default	Meaning
Rx sensitivity	PL_0	-100 dBm	Signal level at which packet reception probability approaches 0%.
RSSI Inflection Point	$\text{rssi}_{50\%}$	-92 dBm	Signal level at which 50% of packets are received.
Transmitting range	d_0	20 m	Reference distance for the path loss formula; treated as fixed per scenario.
Path Loss Exponent	α	3.0	Governs how quickly signal strength decays with distance.
AWGN sigma	σ	3.0	Standard deviation of the additive Gaussian noise applied to the signal level.

ment. LogisticLoss additionally supports an optional time-varying path loss component, evolving as a bounded random walk, intended to emulate slow drift in channel conditions over time, a feature directly relevant to the adaptive calibration extension discussed in Chapter 3.

2.5 Limitations of Default/Static Radio Model Parameters

The parameter values listed in Table 2.1 are Cooja's defaults, chosen to be broadly representative rather than tuned to any particular real deployment. This is reasonable for general-purpose simulation, but it directly limits fidelity whenever a simulated network is meant to reproduce the behavior of one specific real deployment: a single generic Path Loss Exponent and AWGN sigma cannot simultaneously capture the very different propagation conditions found across, for example, an open office floor, a corridor lined with metal furniture, or an outdoor courtyard. Because these parameters are static once set, they also cannot capture time-varying real-world effects such as interference from co-located devices, human movement, or battery-related changes in transmit power, unless the optional time-varying extension described above is explicitly configured and even then, its bounded random-walk behavior is a generic placeholder rather than a model driven by real observed conditions. This is precisely the gap that calibrating these parameters against real KPIs, and adaptively updating them over time, is intended to close.

2.6 Conclusion

This chapter reviewed the physical phenomena (path loss, shadowing, multipath effects, and noise) that any radio propagation model must approximate, situated these within the broader deterministic/statistical/empirical taxonomy of propagation models, and showed that the log-distance/log-normal shadowing model, well established in the wireless networking literature

and specifically validated for low-power IEEE 802.15.4 transceivers, is exactly the model underlying Cooja's LogisticLoss radio medium. Having established LogisticLoss's exact formula and free parameters, the next chapter turns to how these parameters can be calibrated: from manual/empirical tuning approaches, through calibration framed as an inverse problem, to the machine-learning-based and adaptive calibration strategies this report reviews.

Chapter 3

Calibration of Network Simulators

3.1 Introduction

We saw in Chapter 2 that Cooja's LogisticLoss radio medium is an empirical model with a small set of free parameters (Path Loss Exponent, RSSI Inflection Point, AWGN sigma) whose default values only approximate generic indoor conditions. This chapter reviews how such parameters can be set to match a specific real deployment, a process generally referred to as simulator calibration. It moves from manual, measurement-based calibration approaches, through a formal framing of calibration as an inverse problem, to machine-learning-based calibration methods, and finally to adaptive calibration strategies capable of responding to conditions that change over time, closing with a comparison of concept-drift detectors and a justification for the one used in this work.

3.2 What Is Simulator Calibration?

In the context of network simulation, calibration refers to the process of setting a model's free parameters so that the simulator's output reproduces the behavior of a specific real deployment, as opposed to using generic default values intended to be broadly representative. For an empirical radio propagation model such as LogisticLoss, this means finding values for the Path Loss Exponent, the RSSI Inflection Point, and the AWGN sigma such that KPIs computed from simulated traffic (e.g., packet reception ratio, RSSI distributions) match those measured on the real network being modeled. Calibration is what allows an otherwise generic simulator to serve as an accurate *digital twin*, a virtual representation of a real deployment that functions as its digital counterpart (Khan et al., 2022), rather than only a qualitative approximation of IoT networks in general.

3.3 Manual and Empirical Calibration Approaches

The most direct calibration approach is to derive model parameters from field measurements by hand: deploying real hardware, measuring signal strength and packet loss at known distances, and fitting the model's parameters to this data using standard curve-fitting techniques. Dezfouli

et al. (2015) follow essentially this approach, deriving log-normal shadowing parameters for specific low-power transceivers (including the CC2420 used by the Z1 mote in this work) directly from measured channel data. Similar manual, measurement-driven approaches recur elsewhere in the literature: Miranda et al. (2013) derive path loss exponent and shadowing deviation values for IEEE 802.15.4 wireless sensor networks from field measurements taken across free-space, in-building, and industrial environments, while Zinonos and Vassiliou (2012) hand-fit a custom propagation model to measurements collected inside a running industrial oil refinery, discussed further in Chapter 4. This kind of manual calibration is straightforward to understand and requires no additional machine learning infrastructure, but it does not scale well: each new deployment scenario, hardware platform, or environment requires a new measurement campaign and a new manual fitting exercise, and the resulting parameters remain fixed once derived, unable to track any subsequent change in the environment. These limitations motivate reframing calibration as a problem that can be solved automatically, and ideally continuously, which the rest of this chapter addresses.

3.4 Calibration as an Inverse Problem

Manual calibration, described above, proceeds in the *forward* direction: parameters are fitted so that the model's forward simulation matches measured data, typically through iterative fitting or optimization over a chosen parameter set. Framing calibration as an *inverse problem* makes this relationship explicit and reverses the roles of input and output: given a set of KPIs observed on a real network, the goal is to directly recover the underlying radio model parameters that would cause a simulator to reproduce those same KPIs, without needing to iteratively re-simulate and compare at each step. This framing is what makes calibration amenable to machine learning: rather than treating parameter fitting as a per-deployment manual exercise, a model can be trained once to map observed KPIs directly to radio model parameters, and then reused to calibrate new deployments simply by feeding in newly observed KPIs.

Figure 3.1 illustrates this relationship. The top row is the forward problem the simulator already solves: given a set of radio model parameters, Cooja produces simulated KPIs. The bottom row is the inverse problem this report's calibration approach targets: given real, measured KPIs, an ML regressor estimates the parameters that would have produced them.

3.5 Machine Learning for Calibration

Adeogun (2019) demonstrates this inverse-problem approach directly for stochastic radio propagation models: a multilayer perceptron is trained to map summary statistics of channel data (computed from synthetically generated channel realizations) to the model's parameters, and the trained network is then used to estimate parameters directly from measurements. This bypasses

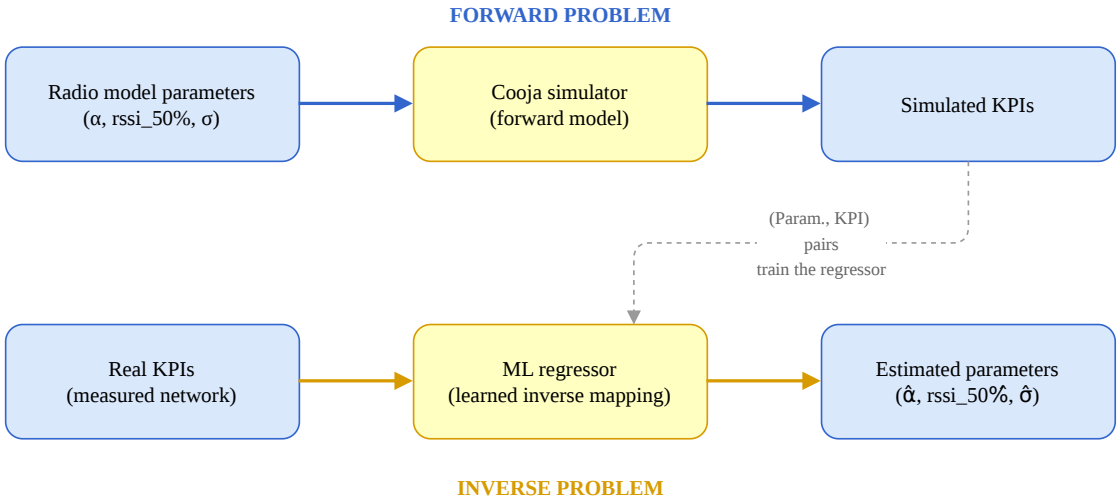


Figure 3.1: Calibration reframed as an inverse problem: the forward model (Cooja) generates synthetic (parameter, KPI) training pairs used to train an ML regressor, which is then applied to real, measured KPIs to estimate the corresponding radio model parameters.

the traditional multi-step process of manually extracting individual multipath components from measured data before fitting a model, replacing it with a single learned mapping from observed channel statistics to parameters. This result generalizes naturally to the calibration problem addressed in this report: an ML regressor (such as a multilayer perceptron) can be trained on synthetically generated KPI data labeled with known LogisticLoss parameters, learning to invert the simulator's forward model, and then applied to KPIs measured on a real network to estimate the parameters that would reproduce that network's behavior in simulation. Figure 3.2 illustrates this calibration pipeline: synthetic channel realizations are generated and processed into summary statistics used to train the regressor, while real, unlabeled calibration measurements are processed the same way and fed into the resulting frozen model to produce the final parameter estimates.

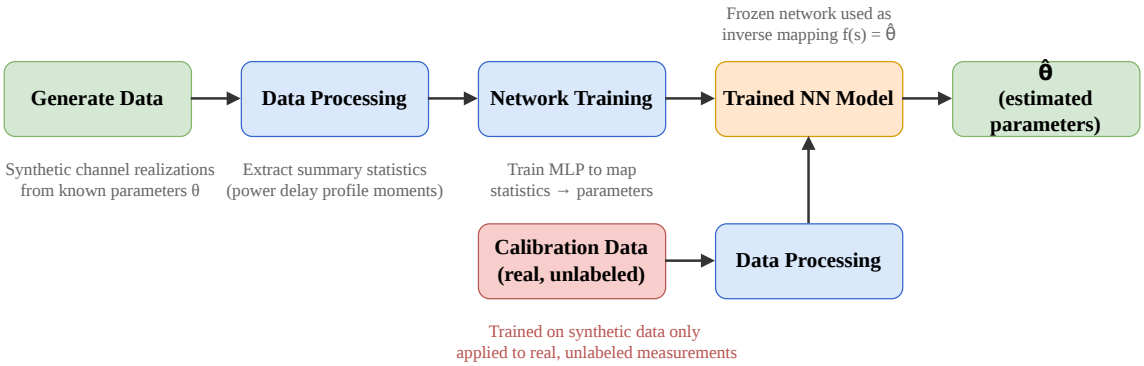


Figure 3.2: Adeogun's calibration pipeline: the network is trained on synthetic data and then applied to real, unlabeled measurements to estimate the corresponding parameters, adapted from Adeogun (2019).

3.6 Toward Adaptive Calibration

A radio model calibrated once, whether manually or via a trained ML regressor, still produces a single fixed set of parameters. Real IoT deployments, however, are not static: interference from co-located devices, battery-related changes in transmit power, and environmental changes (moved furniture, changed occupancy, weather for outdoor deployments) can all shift the true underlying propagation conditions after calibration has already been performed. Keeping a calibrated simulator synchronized with such changes requires detecting when the previously calibrated parameters have become stale, a problem studied extensively in the machine learning literature under the name *concept drift detection* (Gama et al., 2014).

Concept drift detectors

Concept drift detection methods generally fall into three families Gama et al. (2014): *sequential analysis* methods, which accumulate a running statistic over time and signal drift once it crosses a threshold; *control chart / statistical* methods, which track the mean and variance of an error signal directly; and *windowing* methods, which compare statistics computed over a recent window of data against a reference window or against the stream's history. Commonly used detectors from each family include:

- **CUSUM (Cumulative Sum)**: a sequential analysis method introduced by E. S. Page, 1954 that accumulates the deviation of observed values from their mean, signaling drift once this cumulative sum exceeds a user-defined threshold; low in computational and memory cost, making it well suited to resource-constrained settings.
- **Page-Hinkley (PH)** Page (1954): a close variant of CUSUM, also a sequential analysis method, designed specifically to detect changes in the mean of a signal by monitoring the accumulated difference between observed values and their running average against a tunable threshold.
- **DDM (Drift Detection Method)** Gama et al. (2004): a control-chart-based method that tracks a classifier's error rate and its standard deviation, signaling drift when the error rate increases significantly above its historical minimum.
- **EDDM (Early Drift Detection Method)** Baena-García et al. (2006): a variant of DDM that instead monitors the distance between consecutive errors, making it more sensitive to gradual drift than the original DDM.
- **ADWIN (Adaptive Windowing)** Bifet and Gavaldà (2007): a windowing method that maintains a variable-size window of recent data, comparing statistics between sub-windows and signaling drift when their difference exceeds a threshold; more memory-intensive than

the sequential methods above, since it must maintain and compare multiple sub-windows rather than a single running statistic.

Justification for Page-Hinkley

This report's calibration approach adopts Page-Hinkley for adaptive recalibration, for three reasons that follow directly from the comparison above. First, Page-Hinkley is designed specifically to detect a shift in the *mean* of a monitored signal, which matches the calibration use case directly: a drift in real deployment conditions is expected to manifest as a shift in the mean of a monitored KPI (or of a model-prediction residual), rather than as a change in classifier accuracy, which is what control-chart methods such as DDM/EDDM are built around. Second, Page-Hinkley requires only a single running statistic (a cumulative sum and a running mean) rather than multiple retained sub-windows, giving it a lightweight, constant memory footprint compared to windowing methods such as ADWIN, which is desirable when the detector is meant to run continuously alongside a live monitoring system. Third, Page-Hinkley is a well-established, simple method with a small number of interpretable parameters (a threshold and a tolerance/sensitivity term), making its behavior easier to reason about and tune than more complex alternatives, at the cost of being, like CUSUM, sensitive to the choice of these parameter values, which requires care during configuration.

3.7 Conclusion

This chapter reviewed how a radio propagation model's free parameters can be set to match a specific real deployment: from manual, measurement-based calibration, through a formal inverse-problem framing that makes automatic, learned calibration possible, to machine-learning-based calibration methods that can bypass manual parameter-extraction steps entirely. It then addressed the fact that real deployments change over time, reviewing concept-drift detection methods from the sequential analysis, control chart, and windowing families, and justified Page-Hinkley as the detector best matched to monitoring a shift in the mean of a KPI signal under a lightweight, continuously running configuration. Having reviewed the theoretical foundations for both static and adaptive calibration, the next chapter surveys the closest related work in the literature and situates this report's contribution relative to it.

Chapter 4

Literature Review

4.1 Introduction

The previous three chapters established the theoretical foundations this report relies on: Chapter 1 situated Cooja and FIT IoT-LAB within the broader landscape of WSN/IoT simulation and reframed the simulation-to-reality gap as a question of radio model fidelity; Chapter 2 detailed the physical phenomena underlying radio propagation and the exact formula and parameters of Cooja's LogisticLoss model; and Chapter 3 reviewed how such a model's parameters can be calibrated, framed calibration as an inverse problem, and surveyed adaptive calibration strategies via concept-drift detection. This chapter now turns to the closest related work in the literature, discussing groups of related contributions, before synthesizing what they collectively do, and do not, address, in order to situate this report's contribution precisely.

4.2 Low-Power Wireless Channel Modeling

This first group of related work addresses the most direct question a calibration approach must confront: whether default, generic radio propagation parameters are actually a reasonable stand-in for a specific real deployment, or whether measured data reveals otherwise. Dezfouli et al. (2015) address this question for low-power IEEE 802.15.4 transceivers directly relevant to this report: they compile log-normal shadowing path loss parameters (path loss exponent and shadowing standard deviation) for specific chips, including the CC2420 used by the Z1 mote in this work, from dedicated prior measurement campaigns, and show, through simulation, how these parameters shape a network's connectivity region and, in turn, protocol-level performance metrics. A more recent and higher-frequency example of the same underlying question is Dash et al. (2025), who characterize path loss experimentally within a live 3.5 GHz Citizens Broadband Radio Service (CBRS) network, comparing measured path loss against standard analytical models and additionally quantifying how weather and foliage conditions shift received signal strength over time. Although CBRS operates at a much higher frequency and larger scale than the low-power IEEE 802.15.4 networks this report is concerned with, their finding, that commercial analytical models diverge from measured reality and that environmental conditions further shift this gap over time, echoes exactly the motivation behind calibrating LogisticLoss rather

than relying on its generic defaults.

Both works are one of the clearest demonstrations that generic default radio model parameters are a poor substitute for measurement-derived ones (Almeida et al., 2023), even at the level of a single transceiver or a single deployment site considered in isolation. However, Dash et al. derive their parameters once, from a single fixed measurement campaign, with no automatic or repeatable calibration procedure, while Dezfouli et al. synthesize parameters compiled from separate campaigns conducted by others; in neither case is there a mechanism for automatically recalibrating to a new deployment, a new environment, or a new radio chip without repeating, or locating, an entire manual measurement-and-fitting process from scratch. Dezfouli et al.'s work also predates the LogisticLoss radio medium examined in this report and targets the log-normal shadowing model in general rather than any specific Cooja implementation, while Dash et al. operate in a cellular CBRS context entirely outside the low-power WSN/IoT domain. Nonetheless, their results are foundational to this report: they are direct experimental confirmation, across two very different frequency regimes, that radio propagation parameters are transceiver-, environment-, and even time-specific rather than universal, which is precisely the premise that motivates calibrating LogisticLoss's parameters rather than relying on Cooja's built-in defaults.

4.3 Machine-Learning-Based Radio Propagation Calibration

This second group of related work moves beyond one-off manual fitting toward automatic, learned calibration methods where a model is trained to infer a propagation model's parameters from data rather than having them fit by hand, the methodological family this report's own approach belongs to. Adeogun (2019) demonstrates this inverse-problem approach directly for stochastic radio propagation models: a multilayer perceptron is trained to map summary statistics of channel data (computed from synthetically generated channel realizations) to the model's parameters, and the trained network is then used to estimate parameters directly from measurements. Concretely, the network is trained on nine summary statistics derived from the power delay profile: the mean and full covariance of three temporal moments (the zeroth moment, or total received energy; the first moment, or mean delay; and the second moment, or delay spread), each averaged over 500 synthetic channel realizations per parameter setting. This bypasses the traditional multi-step process of manually extracting individual multipath components from measured data before fitting a model, replacing it with a single learned mapping from observed channel statistics to parameters. Bharti et al. (2022) address the same calibration problem with a different tool: rather than training a neural regressor, they propose a likelihood-free calibration method based on approximate Bayesian computation and kernel-based maximum mean discrepancy, applied to two stochastic channel models (Saleh-Valenzuela and propagation graph models) using both simulated and 60 GHz measured indoor data; unlike Adeogun's point-estimate regressor, their method returns a full posterior distribution over parameter values rather

than a single estimate.

Adeogun's approach is the closest methodological precedent to the calibration strategy this report reviews: both frame calibration as a learned, inverse mapping from observed channel statistics to model parameters, trained on synthetic data and applied to real measurements. Bharti et al.'s kernel-based method confirms that this general strategy, learning to invert a stochastic channel model from synthetic data without manual multipath extraction, is not specific to neural networks, but reflects a broader trend toward automatic, likelihood-free calibration across the radio propagation modeling literature. However, both works target general stochastic multipath channel models rather than the specific LogisticLoss model used in network simulators such as Cooja, are not evaluated against any WSN/IoT network simulator or physical IoT testbed, and do not address the possibility that the underlying channel conditions might change after calibration. Together, they demonstrate that the learned-inverse-mapping approach is sound in principle, across at least two different estimation strategies, without addressing its application to network-simulator radio mediums specifically, or to the adaptive, time-varying setting this report is also concerned with.

4.4 Cooja Validated Against Physical Testbeds

This third group of related work addresses a question closer to this report's own validation strategy: how well a network simulator's radio behavior matches a physical testbed once specifically fitted to it, and whether this holds across different simulators and environments rather than only for Cooja. Zinonos and Vassiliou (2012) develop a new radio propagation model directly from measurements collected inside a running industrial oil refinery, as part of the GINSENG project [1], and implement this model in the Cooja simulator; they report that the refinery's measured propagation behavior did not conform to any legacy (pre-existing) radio propagation model available at the time, and show that their custom, measurement-derived model successfully reproduces the refinery testbed's behavior once implemented in Cooja. Kanaris et al. (2019) take a complementary angle on the same simulator, examining how inaccuracies at the physical propagation layer of Cooja propagate into higher-layer network planning decisions; combining Cooja simulation with the TruNET wireless physical testbed (Fractal Networx Limited, 2017), they show that correctly passing physical-layer information (rather than relying on simplified defaults) up to higher layers substantially improves the accuracy of network-level performance predictions, particularly at the edge of radio coverage where IEEE 802.15.4/Zigbee-based systems exhibit markedly bimodal (either mostly-received or mostly-lost) packet behavior. The same underlying question, whether a simulator's radio model can be made to match a physical network once explicitly calibrated to it, also predates Cooja itself: Ivanov et al. (2007) validate the widely used ns-2 simulator by comparing a simulated, an emulated, and a real wireless network side by side,

[1] Project information: GINSENG – Performance Control in Wireless Sensor Networks (EU CORDIS, Project 224282).

explicitly calibrating ns-2's radio propagation model to the real network beforehand so that any remaining differences could be attributed to the simulator itself rather than to an uncalibrated radio model; they find that packet delivery ratios and connectivity graphs are well represented once calibrated this way, while packet latencies remain less accurate. These three works are a direct precedent for using a network simulator together with a physical testbed to validate a radio model's fidelity, and Zinonos and Vassiliou's finding that legacy models failed to match a specific real environment, together with Ivanov et al.'s explicit calibration step prior to comparison, is a strong, concrete illustration of the same fidelity problem this report addresses. However, all three calibration processes are entirely manual: a new propagation model was designed and hand-fitted specifically for the oil refinery environment in Zinonos and Vassiliou, and a one-off calibration to a specific real network was performed in Ivanov et al., with no mechanism proposed in either work for automatically recalibrating the model, applying the same procedure to a different environment, or updating it should conditions change over time. Kanaris et al. reinforce, from a network-planning perspective, the same argument this report makes from a calibration perspective, that physical-layer radio model fidelity is not a peripheral detail but a determining factor in whether a simulator's higher-layer results are trustworthy. And their emphasis on bimodal behavior at the edge of coverage is directly relevant to LogisticLoss, whose logistic PRR-RSSI curve is specifically designed to capture this kind of sharp reception/no-reception transition; however, their contribution addresses *why* physical-layer accuracy matters and how it should be propagated through simulation layers, rather than *how* to automatically obtain accurate physical-layer parameters for a specific deployment in the first place. None of the three use FIT IoT-LAB or any of the standard low-power motes (Z1, M3) considered in this report, and none combine their validation methodology with an automatic or adaptive calibration mechanism.

4.5 Data-Driven Digital Twins

This last group of related work addresses digital twins directly, the framing this report's own calibration approach is ultimately in service of, and situates it relative to two of the most recent contributions in this space. Tarafder et al. (2026) propose a calibration framework, Statistics-Consistent Sim-to-Lab Adaptation, that improves the fidelity of a ray-traced digital twin channel toward laboratory measurements for W-band (millimeter-wave) industrial Orthogonal Frequency-Division Multiplexing (OFDM) links, by explicitly aligning statistical properties of the simulated and measured channels, such as the power delay profile and delay spread distribution, rather than matching individual channel realizations. Closer in spirit to this report's network-level concerns, Si-Mohammed and Théoleyre (2025) propose a lightweight, per-link data-driven approach to replicating a wireless network as a digital twin: rather than modeling detailed radio propagation, they train link-specific quality prediction models directly on a single, informative KPI, the Packet Delivery Ratio, with the explicit goal of supporting a self-adaptive digital twin capa-

ble of tracking real-world conditions without resource-intensive local measurement campaigns (Ozyurt, 2026).

Both works are the most recent evidence that data-driven digital twin calibration remains an active and evolving research area well beyond the specific domain of low-power IoT networks. Tarafder et al.'s statistics-alignment philosophy is conceptually related to this report's KPI-based calibration approach, in that both align a model's behavior against summary statistics of observed data rather than requiring an exact realization-by-realization match; unlike this report, however, their learned correction operates on the ray-traced channel's *output* while leaving the ray-tracing solver, scene geometry, and material properties themselves unchanged, rather than calibrating the parameters of an underlying, reusable propagation model such as LogisticLoss. They also operate in an entirely different regime, millimeter-wave (92–95 GHz) industrial links using ray-tracing-based channel twins, which is far removed from the sub-GHz/2.4 GHz, resource-constrained, empirical-model setting this report is concerned with. Si-Mohammed and Théoleyre's per-link, KPI-driven approach is considerably closer in spirit, sharing this report's emphasis on a single, informative KPI as the basis for a lightweight, self-adaptive digital twin, and their explicit framing around adaptivity to changing real-world conditions is the closest any reviewed work comes to this report's own adaptive calibration extension. Their primary contribution predicts link quality directly from data rather than calibrating the parameters of an underlying physical radio propagation model such as LogisticLoss. A secondary experiment in the same paper does calibrate Cooja's UDGM radio model against the same FIT IoT-LAB testbed used in this report, but only by matching each link's fixed transmission probability to that link's own long-term average PDR. The authors show that even this personalized, per-link calibration fails to track PDR's temporal dynamics, since real-world link quality continues to evolve after the average is fixed, precisely the adaptive gap this report's calibration approach is designed to close. Taken together, both works confirm that, even as data-driven digital twin research advances rapidly, including work explicitly targeting self-adaptivity, no equivalent, systematic, automatic, and adaptive calibration treatment of a specific radio propagation model has emerged for the low-power IoT simulators and testbeds this report addresses.

4.6 Synthesis and research gap

Table 4.1 summarizes the works reviewed in this chapter against the dimensions relevant to this report's contribution.

Table 4.1: Synthesis of reviewed related work

Work	Sim./Twin	Real Testbed	KPI(s)	Method	Adapt.?
Dezfouli et al. (2015)	None	CC1000/CC2420 field data	RSSI, path loss	Manual fitting	No
Dash et al. (2025)	None	Live CBRS network	Path loss, RSRP, SINR	Manual, exp.	No
Adeogun (2019)	None	Synthetic + measured	PDP moments	ML (MLP) inverse map	No
Bharti et al. (2022)	None	Synthetic + 60 GHz	PDP stats	Kernel-based (ABC)	No
Zinonos & Vassiliou (2012)	Cooja	Oil refinery (GINSENG)	RSSI	Manual, env.-specific	No
Kanaris et al. (2019)	Cooja	TruNET wireless	PRR	Cross-layer analysis	No
Ivanov et al. (2007)	ns-2	Real wireless network	PDR, connectivity, latency	Manual calibration	No
Tarafder et al. (2026)	Ray-traced twin	W-band OFDM lab	PDP, delay spread	Post-RT output calib.	No
Si-Mohammed & Théoleyre (2025)	Per-link twin; Cooja (UDGM, secondary)	FIT IoT-LAB	PDR	Data-driven; static avg.-PDR calib.	Partially
This work	**Cooja**	**FIT IoT-LAB**	**RSSI, loss rate, throughput, delay**	**ML inverse mapping**	**Yes**

Two gaps emerge clearly from this comparison. First, only one reviewed work combines Cooja and FIT IoT-LAB at all: Si-Mohammed and Théoleyre's secondary validation experiment calibrates Cooja's UDGM radio model against the same FIT IoT-LAB testbed used in this report, but does so by matching each link's fixed transmission probability to a single long-term average PDR value, a static, per-link calibration of a different, threshold-based radio model (UDGM, not LogisticLoss) rather than a learned, generalizable calibration of a specific radio propagation model's own parameters; the authors themselves show this static approach fails to track PDR's temporal dynamics. None of the remaining works that use Cooja against a real testbed (Zinonos and Vassiliou; Kanaris et al.) validate against FIT IoT-LAB, the work that validates a simulator against a real network through explicit calibration (Ivanov et al.) targets ns-2 rather than Cooja, and the works most methodologically related to this report's calibration approach (Dezfouli et al.; Dash et al.; Adeogun; Bharti et al.; Tarafder et al.) do not use Cooja or FIT IoT-LAB at all. Second, and more importantly, none of the reviewed works combine automatic, learned calibration of a specific radio propagation model with an adaptive mechanism for detecting and responding to changing conditions after calibration: Adeogun's and Bharti et al.'s ML-based approaches are automatic but static and target generic stochastic channel models rather than LogisticLoss, Tarafder et al.'s statistics-alignment approach is likewise automatic but static and

corrects a ray-traced channel's output rather than calibrating an underlying model's own parameters, the manually calibrated Cooja and ns-2 studies (Zinonos and Vassiliou; Kanaris et al.; Ivanov et al.; and the Cooja experiment in Si-Mohammed and Théoleyre) are adaptive to a real environment only in the sense of being manually tuned once for it, and even Si-Mohammed and Théoleyre's explicitly self-adaptive digital twin predicts link quality directly rather than calibrating an underlying, reusable radio propagation model. To the best of our knowledge, no prior work specifically addresses automatic, learned, and adaptive calibration of a network simulator's specific radio propagation model against a physical low-power IoT testbed, and in particular no prior work addresses this for the Cooja/FIT IoT-LAB pairing. This is the gap the calibration approach reviewed in this report is intended to fill.

In order to empirically confirm the absence of directly overlapping work, a systematic search was conducted on Scopus. The query used was:

```
Cooja AND ("radio model" OR "radio propagation model") AND
(calibration OR calibrate)
```

This search did not return any relevant results. While individual searches on radio propagation model calibration, or on Cooja validated against physical testbeds, each return results on their own (as reflected in the works reviewed above), no combination of the two returns any match. This confirms the novelty of the gap identified in this chapter: no existing work addresses calibration of a network simulator's radio model, specifically within Cooja, against a physical testbed. A screenshot of the Scopus search results (showing zero hits) is provided in Figure 4.1 to illustrate this gap.

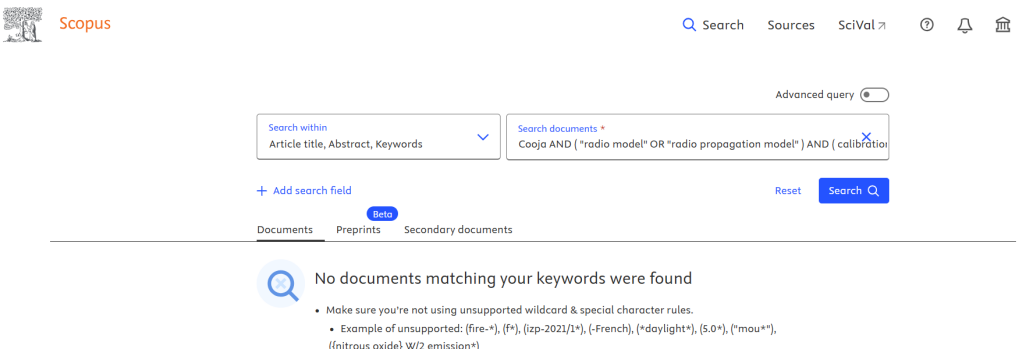


Figure 4.1: Scopus search showing no results for the combination of Cooja and radio model calibration.

4.7 Conclusion

This chapter reviewed closely related works, grouped into four themes: manual, measurement-driven channel modeling; machine-learning- and kernel-based calibration of stochastic radio propagation models; simulator validation against physical testbeds through manual calibration; and recent data-driven digital twin research. Individually, each contributes part of the picture

this report relies on: measurement-derived parameters are necessary and transceiver-specific (Dezfouli et al.; Dash et al.), learned and kernel-based inverse mappings are sound calibration strategies (Adeogun; Bharti et al.), simulators including Cooja and ns-2 can be meaningfully validated, and even manually calibrated, against real hardware (Zinonos and Vassiliou; Kanaris et al.; Ivanov et al.; and, in a secondary experiment, Si-Mohammed and Théoleyre), and data-driven digital twin research, including explicitly self-adaptive approaches, remains an active area of research (Tarafder et al.; Si-Mohammed and Théoleyre). None, however, combine automatic, learned, and adaptive calibration of a specific radio propagation model for the Cooja/FIT IoT-LAB pairing this report is concerned with. Having reviewed the theoretical foundations and situated this report's contribution relative to the closest related work, the general conclusion that follows summarizes this report as a whole.

General Conclusion

This report set out to review the theoretical foundations required to address the simulation-to-reality gap observed between the Cooja network simulator and the FIT IoT-LAB physical testbed. Chapter 1 argued that this gap is best understood as a question of radio model fidelity rather than hardware mismatch: Cooja emulates real firmware on hardware whose radio-level characteristics are broadly representative of the low-power IEEE 802.15.4 devices being studied, so any residual divergence in observed behavior points to the fidelity of the simulator's software radio model rather than to the modest differences between, for instance, the Z1 mote's CC2420 radio and the M3 node's AT86RF231. Chapter 2 then examined the physical phenomena that any radio propagation model must approximate, situated these within the wireless networking literature, and detailed the exact formula and free parameters, the Path Loss Exponent, the RSSI Inflection Point, and the AWGN sigma, of Cooja's LogisticLoss model, the empirical radio medium this report is concerned with.

Chapter 3 reviewed how such parameters can be calibrated: from manual, measurement-based approaches, through a formal inverse-problem framing that recasts calibration as learning a direct mapping from observed KPIs to model parameters, to machine-learning-based methods shown in the literature to bypass traditional manual parameter-extraction steps entirely. It further reviewed adaptive calibration strategies drawn from the concept-drift detection literature, comparing sequential-analysis, control-chart, and windowing approaches, and justified Page-Hinkley as the detector best suited to monitoring a shift in the mean of a KPI signal under a lightweight, continuously running configuration.

Chapter 4 surveyed the closest related work: measurement-derived channel modeling for low-power transceivers, machine-learning-based calibration of stochastic radio propagation models, Cooja-based radio model validation against physical testbeds in industrial and urban settings, cross-layer simulation accuracy analysis, and the most recent digital twin channel calibration research. Considered individually, each of these contributions supports part of the approach this report reviews. Considered together, however, none combine automatic, learned calibration with an adaptive mechanism for responding to changing conditions, and none address the specific pairing of Cooja and FIT IoT-LAB. This is the gap the calibration approach reviewed throughout this report is intended to address.

Taken as a whole, this report establishes that closing the simulation-to-reality gap for low-power IoT network simulation is best approached as a data-driven, adaptive radio model calibration problem: one with clear theoretical grounding in the wireless propagation literature, a sound methodological precedent in machine-learning-based inverse modeling, and a well-defined, cur-

rently unaddressed gap at the intersection of these two lines of work.

Bibliography

Adeogun, R. O. (2019). Calibration of stochastic radio propagation models using machine learning. *IEEE Antennas and Wireless Propagation Letters*, *18*(12), 2538–2542. https://doi.org/10.1109/LAWP.2019.2942221

Adjih, C., Baccelli, E., Fleury, E., Harter, G., Mitton, N., Noël, T., Pissard-Gibollet, R., Saint-Marcel, F., Schreiner, G., Vandaele, J., & Watteyne, T. (2015). FIT IoT-LAB: A large scale open experimental IoT testbed. *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*, 459–464. https://doi.org/10.1109/WF-IoT.2015.7389098

Almeida, E. N., Fontes, H., Campos, R., & Ricardo, M. (2023). Position-based machine learning propagation loss model enabling fast digital twins of wireless networks in ns-3. *Proceedings of the 2023 Workshop on ns-3*, 69–77. https://doi.org/10.1145/3592149.3592150

Baena-García, M., del Campo-Avila, J., Fidalgo, R., Bifet, A., Gavaldà, R., & Morales-Bueno, R. (2006). Early drift detection method. *Proceedings of the 4th International Workshop on Knowledge Discovery from Data Streams*, 77–86.

Bharti, A., Briol, F.-X., & Pedersen, T. (2022). A general method for calibrating stochastic radio channel models with kernels. *IEEE Transactions on Antennas and Propagation*, *70*(6), 3986–4001. https://doi.org/10.1109/TAP.2021.3083761

Bifet, A., & Gavaldà, R. (2007). Learning from time-changing data with adaptive windowing. *Proceedings of the 2007 SIAM International Conference on Data Mining*, 443–448. https://doi.org/10.1137/1.9781611972771.42

Dash, B. K., Caezza, J. A., & Malandra, F. (2025). Experimental propagation analysis in the CBRS spectrum: Path loss characterization, weather and foliage impact. https://doi.org/10.36227/techrxiv.173144788.88858804/v4

Dezfouli, B., Radi, M., Razak, S. A., Hwee-Pink, T., & Bakar, K. A. (2015). Modeling low-power wireless communications. *Journal of Network and Computer Applications*, *51*, 102126. https://doi.org/10.1016/j.jnca.2014.02.009

Dunkels, A., Grönvall, B., & Voigt, T. (2004). Contiki—a lightweight and flexible operating system for tiny networked sensors. *Proceedings of the First IEEE Workshop on Embedded Networked Sensors*, 455–462. https://doi.org/10.1109/LCN.2004.38

Gama, J., Medas, P., Castillo, G., & Rodrigues, P. (2004). Learning with drift detection. In *Advances in artificial intelligence—SBIA 2004* (pp. 286–295, Vol. 3171). Springer. https://doi.org/10.1007/978-3-540-28645-5_29

Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., & Bouchachia, A. (2014). A survey on concept drift adaptation. *ACM Computing Surveys*, *46*(4), 44:1–44:37. https://doi.org/10.1145/2523813

ISO/IEC/IEEE. (2017). *ISO/IEC/IEEE 24765:2017: Systems and Software Engineering—Vocabulary*. International Organization for Standardization.

Ivanov, S., Herms, A., & Lukas, G. (2007). Experimental validation of the ns-2 wireless model using simulation, emulation, and real network. *Communication in Distributed Systems (KiVS), 2007 ITG-GI Conference*, 433–444. https://www.researchgate.net/publication/228619429

Kanaris, L., Sergiou, C., Kokkinis, A., Pafitis, A., Antoniou, N., & Stavrou, S. (2019). On the realistic radio and network planning of iot sensor networks. *Sensors*, *19*(15), 3264. https://doi.org/10.3390/s19153264

Khan, L. U., Han, Z., Saad, W., Hossain, E., Guizani, M., & Hong, C. S. (2022). Digital twin of wireless systems: Overview, taxonomy, challenges, and opportunities. *IEEE Communications Surveys & Tutorials*, *24*(4), 2230–2254. https://doi.org/10.1109/COMST.2022.3198273

Kushalnagar, N., Montenegro, G., & Schumacher, C. (2007, August). RFC 4919: IPv6 over low-power wireless personal area networks (6LoWPANs): Overview, assumptions, problem statement, and goals. https://www.rfc-editor.org/rfc/rfc4919

Levis, P., Madden, S., Polastre, J., Szewczyk, R., Whitehouse, K., Woo, A., Gay, D., Hill, J. L., Welsh, M., Brewer, E. A., & Culler, D. E. (2005). Tinyos: An operating system for sensor networks. In W. Weber, J. M. Rabaey, & E. H. L. Aarts (Eds.), *Ambient intelligence* (pp. 115–148). Springer. https://doi.org/10.1007/3-540-27139-2_7

Miranda, J., Abrishambaf, R., Gomes, T., Gonçalves, P., Cabral, J., Tavares, A., & Monteiro, J. (2013). Path loss exponent analysis in wireless sensor networks: Experimental evaluation. *2013 11th IEEE International Conference on Industrial Informatics (INDIN)*, 54–58. https://doi.org/10.1109/INDIN.2013.6622857

Ozyurt, M. (2026). Descriptor-based tiling for transferable path-loss learning in 6G digital twins [Available online 13 July 2026]. *Computer Communications*. https://doi.org/10.1016/j.comcom.2026.108620

Page, E. S. (1954). Continuous inspection schemes. *Biometrika*, *41*(1–2), 100–115. https://doi.org/10.1093/biomet/41.1-2.100

Rappaport, T. S. (2002). *Wireless communications: Principles and practice* (2nd ed.). Prentice Hall.

Shelby, Z., Hartke, K., & Bormann, C. (2014, June). RFC 7252: The constrained application protocol (CoAP). https://www.rfc-editor.org/rfc/rfc7252

Si-Mohammed, S., & Théoleyre, F. (2025). Per link data-driven network replication towards self-adaptive digital twins. *27th International Conference on Modeling, Analysis and Simula-*

tion of Wireless and Mobile Systems (MSWiM). https://doi.org/10.1109/MSWiM67937.2025.11308908

Tarafder, P., Oyekola, A. O., Mannil, J. A., Ahmed, I., Hassan, Z., Rawat, D. B., & Che, W. (2026). Calibrating the digital twin channel: Statistics-consistent sim-to-lab adaptation for w-band industrial ofdm links.

Warneke, B., Last, M., Liebowitz, B., & Pister, K. S. J. (2001). Smart dust: Communicating with a cubic-millimeter computer. *Computer*, *34*(1), 44–51. https://doi.org/10.1109/2.895117

Winter, T., Thubert, P., Brandt, A., Clausen, T. H., Hui, J. W., Kelsey, R., Levis, P., Pister, K., Struempler, M., Vasseur, J.-P., & Alexander, R. K. (2012, March). RFC 6550: RPL: IPv6 routing protocol for low-power and lossy networks. https://www.rfc-editor.org/rfc/rfc6550

Yun, Z., & Iskander, M. F. (2015). Ray tracing for radio propagation modeling: Principles and applications. *IEEE Access*, *3*, 1089–1100. https://doi.org/10.1109/ACCESS.2015.2453991

Zinonos, Z., & Vassiliou, V. (2012). Radio propagation in industrial wireless sensor network environments: From testbed to simulation evaluation. *Proceedings of the 7th ACM Workshop on Performance Monitoring and Measurement of Heterogeneous Wireless and Wired Networks (PM2HW2N)*. https://doi.org/10.1145/2387191.2387209

Webography

Atmel Corporation. (2014). AT86RF231 datasheet [https://ww1.microchip.com/downloads/en/DeviceDoc/doc8111.pdf].

Contiki-NG Cooja Simulator. (2024). Cooja: The contiki-ng network simulator [https://github.com/contiki-ng/cooja].

Contiki-NG Documentation. (2024). Contiki-ng documentation—platforms [https://docs.contiki-ng.org/].

Contiki-NG LogisticLoss Source. (2024). LogisticLoss.java [https://github.com/contiki-ng/cooja/blob/master/java/org/contikios/cooja/radiomediums/LogisticLoss.java].

FIT IoT-LAB Grenoble Deployment. (2024). Grenoble site deployment documentation [https://iot-lab.github.io/docs/deployment/grenoble/].

FIT IoT-LAB M3 Board Documentation. (2024). IoT-LAB M3 [https://iot-lab.github.io/docs/boards/iot-lab-m3/].

Fractal Networx Limited. (2017). TruNET Wireless [https://www.fractalnetworx.com].

IEEE. (2020). *IEEE Standard for Low-Rate Wireless Networks*. Institute of Electrical and Electronics Engineers.