Energy-efficient, Large-scale Ultra-wideband Communication and Localization

Davide Vecchia

Advisor **Prof. Gian Pietro Picco**
University of Trento, Italy

Committee **Prof. Stefano Chessa**
University of Pisa, Italy
Prof. Marco Zimmerling
University of Freiburg, Germany
Prof. Niki Trigoni
University of Oxford, UK

University of Trento
Trento, Italy
May 2022

To my parents.

Abstract

Among the low-power wireless technologies that have emerged in recent years, ultra-wideband (UWB) has successfully established itself as the reference for accurate ranging and localization, both outdoors and indoors. Due to its unprecedented performance, paired with relatively low energy consumption, UWB is going to play a central role in the next wave of location-based applications. As the trend of integration in smartphones continues, UWB is also expected to reach ordinary users, revolutionizing our lives the same way GPS and similar technologies have done. But the impact of UWB may not be limited to ranging and localization. Because of its considerable data rate, and its robustness to obstacles and interference, UWB communication may hold untapped potential for sensing and control applications.

Nevertheless, several research questions still need to be answered to assess whether UWB can be adopted widely in the communication and localization landscapes. On one hand, the rapid evolution of UWB radios and the release of ever more efficient chips is a clear indication of the growing market for this technology. However, for it to become pervasive, full-fledged communication and localization systems must be developed and evaluated, tackling the shortcomings affecting current prototypes. UWB systems are typically single-hop networks designed for small areas, making them impractical for large-scale coverage. This limitation is found in communication and localization systems alike. Specifically for communication systems, energy-efficient multi-hop protocols are hitherto unexplored. As for localization systems, they rely on mains-powered anchors to circumvent the issue of energy consumption, in addition to only supporting small areas. Very few options are available for light, easy to deploy infrastructures using battery-powered anchors. Nonetheless, large-scale systems are required in common settings like industrial facilities and agricultural fields, but also office spaces and museums. The general goal of enabling UWB in spaces like these entails a number of issues. Large multi-hop infrastructures exacerbate the known limitations of small, single-hop, networks; notably, reliability and latency requirements clash with the need to reduce energy consumption. Finally, when device mobility is a factor, continuity of operations across the covered area is a challenge in itself.

In this thesis, we design energy-efficient UWB systems for large-scale areas, supporting device mobility across multi-hop infrastructures. As our opening contribution, we study the unique interference rejection properties of the radio to inform our design. This analysis yields a number of findings on the impact of interference in communication and distance estimation, that are directly usable by developers to improve UWB solutions. These findings also suggest that concurrent transmissions in the same frequency channel are a practical option in UWB. While the overlapping

of frames is typically avoided to prevent collisions, concurrent transmissions have counter-intuitively been used to provide highly reliable communication primitives for a variety of traffic patterns in narrowband radios. In our first effort to use concurrent transmissions in a full system, we introduce the UWB version of Glossy, a renowned protocol for efficient network-wide synchronization and data dissemination. Inspired by the success of concurrency-based protocols in narrowband, we then apply the same principles to define a novel data collection protocol, Weaver. Instead of relying on independent Glossy floods like state-of-the-art systems, we weave multiple data flows together to make our collection engine faster, more reliable and more energy-efficient.

With Glossy and Weaver supporting the communication aspect in large-scale networks, we then propose techniques for large-scale localization systems. We introduce Talla, a TDoA solution for continuous position estimation based on wireless synchronization. We evaluate Talla in an UWB testbed and in simulations, for which we replicate accurately the behavior of the clocks in our real-world platforms. We then offer a glimpse of what Talla can be employed for, deploying an infrastructure in a science museum to track visitors. The collected movement traces allow us to analyze fine-grained stop-move mobility patterns and infer the sequence of visited exhibits, which is only possible because of the high spatio-temporal granularity offered by Talla. Finally, with Sonar, we tackle the issue of large-scale ranging and localization when the infrastructure cannot be mains-powered. By blending synchronization and scheduling operations into neighbor discovery and ranging, we drastically reduce energy consumption and ensure years-long system lifetime.

Overall, this thesis enhances UWB applicability in scenarios that were previously precluded to the technology, by providing the missing communication and localization support for large areas and battery-powered devices. Throughout the thesis, we follow an experiment-driven approach to validate our protocol models and simulations. Based on the evidence collected during this research endeavor, we develop full systems that operate in a large testbed at our premises, showing that our solutions are immediately applicable in real settings.

Keywords:
Ultra-wideband, Low-power Wireless, Network Protocol Design, Concurrent Transmissions, RF Localization, TDoA, Ranging, RTLS, Internet of Things, Mobility Patterns.

Acknowledgements

The accomplishments in this thesis would not have been possible without the support I received from many people, to whom I express my sincere gratitude.

First and foremost, I want to thank my advisor Gian Pietro Picco. During my M.Sc. in Trento, I was impressed by his flawless teaching and his passion for research, that motivated me to undertake this challenge. Since I started my Ph.D., my admiration has not changed. I am deeply grateful for the opportunity he gave me and everything I learned from him during these years.

I want to thank Stefano Chessa, Marco Zimmerling, and Niki Trigoni for reviewing my work, for their positive comments, and for serving in my examination committee. I'm truly honored to share with them this important step of my journey in research.

I also thank current and former members of the D3S research group. In particular, Timofei Istomin and Pablo Corbalán played a decisive role in the early days of my Ph.D., encouraging me to pursue this path and helping me shape the first papers. They have been a true inspiration with their creativity and dedication. But there are also other key figures that I owe my gratitude to. The programming prowess of Diego Lobba is behind Chapter 3. Matteo Trobinger has been there throughout all this experience, giving precious insight on our research questions, and listening to my rants about paperwork and bureaucratic issues. Our collaboration culminated in the work of Chapter 4. Davide Molteni deserves a special mention for all the advice and technical support I received from him. Not to mention the maintenance of the wireless testbed at the core of many of our systems. Beyond your scientific inputs, I want to thank you all for your friendship. I vividly remember our plenary remote meetings during the lockdown. We helped each other get through it and feel less isolated.

I want to acknowledge the contributions from people outside the group as well. Chapter 6 is the result of a fruitful collaboration with Maria Luisa Damiani and Fatima Hachem, from Università degli Studi di Milano. Moreover, the study would not have been possible without the organizational and technical support from MUSE, and especially from Vittorio Cozzio, who was our main reference. The content of Chapter 7 stems from a project with Thales Alenia Space. I thank Enrico Varriale for the invaluable feedback he has continually provided since the system was first devised.

Finally, I express my deepest gratitude to my wonderful family, for their unconditional love and support. I am extremely fortunate to have you there by my side.

Trento, Italy, 20 June 2022 Davide Vecchia

List of Publications

International Conferences

1. D. Vecchia, P. Corbalán, T. Istomin, and G. P. Picco. "Playing with Fire: Exploring Concurrent Transmissions in Ultra-wideband Radios". In *Proceedings of the* 18^{th} *IEEE International Conference on Sensing, Communication and Networking (SECON)*. 2019. DOI: 10.1109/SAHCN.2019.8824929. (Chapter 2)

2. D. Lobba, M. Trobinger, D. Vecchia, T. Istomin, and G. P. Picco. "Concurrent Transmissions for Multi-hop Communication on Ultra-wideband Radios". In *Proceedings of the* 17^{th} *International Conference on Embedded Wireless Systems and Networks (EWSN)*. 2020. (Chapter 3)

3. M. Trobinger, D. Vecchia, D. Lobba, T. Istomin, and G. P. Picco. "One Flood to Route Them All: Ultra-fast Convergecast of Concurrent Flows over UWB". In *Proceedings of the* 18^{th} *ACM Conference on Embedded Networked Sensor Systems (SenSys)*. 2020. (Chapter 4)

4. D. Vecchia, P. Corbalán, T. Istomin, and G. P. Picco. "TALLA: Large-scale TDoA Localization with Ultra-wideband Radios". In *Proceedings of the* 10^{th} *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. 2019. DOI: 10.1109/IPIN.2019.8911790. **Best Paper Award.** (Chapter 5)

5. F. Hachem, D. Vecchia, M. L. Damiani, and G. P. Picco. "Fine-grained Stop-Move Detection in UWB-based Trajectories". In *Proceedings of the* 20^{th} *IEEE International Conference on Pervasive Computing and Communications (PerCom)*. 2022. (Chapter 6)

Contents

Abstract *i*

Acknowledgements *iii*

List of Publications *v*

List of Figures *xi*

List of Tables *xv*

1 Introduction **1**
1.1 Ultra-wideband Radios: Localization *and* Communication 3
1.2 Goals, Contributions, and Methodology 4

I Fast, Efficient, Large-scale Ultra-wideband Communication **9**

2 Playing with Fire: Exploring Concurrent Transmissions in Ultra-wideband Radios **11**
2.1 Ultra-wideband Communication and Ranging 13
2.2 Research Questions 16
2.3 Experimental Setup 17
2.4 Empirical Observations 19
2.4.1 Baseline: Isolated Transmissions 19
2.4.2 Concurrent Transmissions with Different PRFs 19
2.4.3 Concurrent Transmissions with Different Preamble Codes 21
2.4.4 Concurrent Transmissions with the Same RF Configuration . . . 24
2.4.5 Combined Settings 27
2.5 Discussion 28
2.6 Related work 29
2.7 Conclusions 30

3 Concurrent Transmissions for Multi-hop Communication on Ultra-wideband Radios **31**
3.1 Concurrent Transmissions in Narrowband Radios 33
3.1.1 Network Flooding with Glossy 34
3.1.2 Higher-level Abstractions: Crystal 35
3.2 Related Work 37
3.3 Glossy on UWB 37
3.3.1 Implementation Highlights 37

3.3.2 Evaluation . . . 40
3.3.3 Exploring the Limits . . . 45
3.4 Crystal on UWB . . . 48
3.4.1 Implementation Highlights . . . 48
3.4.2 Evaluation . . . 48
3.5 Discussion . . . 51
3.6 Conclusions . . . 53

4 One Flood to Route Them All: Ultra-fast Convergecast of Concurrent Flows over UWB 55
4.1 Background and Related Work . . . 57
4.1.1 CTX Protocols in IEEE 802.15.4 Narrowband . . . 58
4.1.2 CTX Protocols in IEEE 802.15.4 Ultra-wideband . . . 59
4.2 Design Goals and Principles . . . 60
4.2.1 The Drawbacks of a Glossy Legacy . . . 60
4.2.2 WEAVER: The Power of Fine-grained CTX . . . 61
4.2.3 Is It Worth? An Analytical Model . . . 63
4.3 Protocol Details . . . 65
4.4 A Modular Implementation . . . 69
4.4.1 Time Slot Manager: A Flexible CTX Engine . . . 70
4.4.2 Monitoring Energy Consumption . . . 73
4.5 Evaluation . . . 73
4.5.1 Experimental Setup . . . 73
4.5.2 Dissecting WEAVER . . . 75
4.5.3 WEAVER vs. Crystal . . . 77
4.5.4 WEAVER and Mobility . . . 80
4.6 Discussion and Outlook . . . 81
4.7 Conclusions . . . 82

II Ultra-wideband Localization in Large-scale Areas 83

5 TALLA: Large-scale TDoA Localization with Ultra-wideband Radios 85
5.1 TDoA Localization: Fundamentals . . . 86
5.2 Enabling TDoA over Large Areas . . . 87
5.3 Evaluation Methodology . . . 90
5.4 Modeling and Reproducing Timing Inaccuracies . . . 91
5.5 Evaluation . . . 93
5.5.1 Experimental Setup . . . 93
5.5.2 Small-Scale, Single-Hop Experiments . . . 95
5.5.3 Large-scale, Multi-hop Experiments . . . 97
5.6 Related Work . . . 102
5.7 Conclusions and Future Work . . . 103

6 Fine-grained Stop-Move Detection in UWB-based Trajectories 105
6.1 Problem Formulation and Definitions . . . 108

6.2 Segmentation Techniques . . . 108
6.3 A Novel Metric for Stop-Move Detection . . . 109
6.4 Dataset Acquisition and Characteristics . . . 111
6.4.1 Collecting the Dataset . . . 111
6.4.2 Characterizing the Dataset . . . 113
6.5 Findings . . . 115
6.6 Related Work . . . 121
6.7 Conclusions . . . 122

7 Self-Organizing Neighbor Discovery and Ranging for Battery-powered UWB Anchors 125
7.1 Related Work . . . 127
7.2 Goals and Assumptions . . . 128
7.3 Sonar Design . . . 129
7.3.1 High-level Protocol Overview . . . 129
7.3.2 Single-user Mode . . . 130
7.3.3 Multi-user Mode . . . 132
7.3.4 On the Responsiveness of Neighbor Discovery . . . 135
7.4 Implementation . . . 136
7.5 Energy Consumption . . . 138
7.6 Evaluation . . . 142
7.6.1 Discovery and Ranging in Single-user and Multi-user Mode . . . 142
7.6.2 Interference vs. Neighbor Discovery Range . . . 142
7.6.3 Analysis in Multi-user Mode . . . 144
7.7 Conclusions . . . 149

8 Conclusions 151

Bibliography 155

List of Figures

2.1 UWB frame: the SHR is encoded in single pulses, while the data part exploits BPM-BPSK modulation. Preamble codes determine the preamble symbol sequence and the time-hopping code (arrows) for data transmission. 13
2.2 UWB data symbol. 15
2.3 Single-sided two-way ranging. 15
2.4 Network topology of our experiments. M, S_i, and R_i are the master, sender, and receiver (responder), respectively. All nodes are in communication range. The arcs represent the links under study: *weak* (dashed line) and *strong* (solid). 18
2.5 *PRR* with different *PRFs* (channel 4). Concurrent transmissions exploiting different *PRFs* are very likely to be received correctly, especially with *PRF*64. 19
2.6 Concurrent ranging with different *PRFs*. Despite interference, both *PRFs* perform accurate ranging reliably. 20
2.7 CIR with concurrent transmissions using different *PRFs* and $\Delta t = -20.513\,\mu s$. 20
2.8 *PRR* on the *weak* links (channel 4) and different preamble codes for *PRF*16 (2.8a) and *PRF*64 (2.8b). Concurrent transmissions with different preamble codes introduce significant packet loss, especially for the late transmission. 21
2.9 *PRR* on the *strong* links (channel 4) and different preamble codes for *PRF*16 (2.9a) and *PRF*64 (2.9b). 21
2.10 CIR for various time shifts Δt with concurrent transmissions using different preamble codes. 23
2.11 Concurrent ranging with different preamble codes. Significant outliers appear especially with *PRF*16. Many ranging rounds are lost due to interference and RX errors. 24
2.12 *PRR* with the same RF configuration (channel 4, *PRF*64, code 17). 24
2.13 *PRR* in a single-receiver scenario (channel 4, *PRF*64, code 17). 24
2.14 Zoom-in of Figure 2.13. 25
2.15 *PRR* vs. number of concurrent senders (ch.4, *PRF*64, code 17). 26
2.16 Wrong timestamp associated to the received packet due to an interfering transmitter affecting the CIR. 27

3.1 The two Glossy variants in a 4-hop network. 34
3.2 A Crystal epoch (from [84]). 36
3.3 Experimental testbed. Out of the 23 nodes available, 22 were running the protocols under study; node 3 served as a sniffer. 40
3.4 Latency of node 11, the farthest from the initiator (4 hops). Bars denote minimum/maximum values; boxes denote the 25–75% percentile. 42
3.5 Energy consumption: Glossy vs. GlossyTX for narrowband (NB) and UWB. Note the difference in scale between the y-axes. 44

3.6 Time synchronization error. 44
3.7 *PRR* when a transmission is shifted over the symbol duration (short 15 B frame). 46
3.8 Reliability with 4 receivers and 2 transmitters with relative TX attenuation (long packets): average and at-least-one *PRR*. 47
3.9 Average success rate in the first T shared slot, for different values of N and U. 50
3.10 CCDF for T success rate vs. U (N=2). 51
3.11 CCDF for T success rate vs. N (U=5). 51

4.1 Sample executions of Crystal and Weaver. Termination phase is not shown. 61
4.2 Weaving flows of data and acknowledgments. 62
4.3 Latency models for Crystal and Weaver with $U = 3$ initiators (orange, blue, and green) all at maximum hop distance H. Termination phase is not shown. 64
4.4 Estimated number of slots required to deliver U data packets in a network of H hops. 65
4.5 Determining the suppression period L. 67
4.6 Example scenario in which *G*-ACKs block the propagation of data packets. 68
4.7 System architecture. 69
4.8 Flow of control. 71
4.9 Glossy forwarder logic ($N = 1$) atop TSM. 71
4.10 TSM slot structure. 72
4.11 Testbed spanning $84 \times 33m^2$. In Floor, node 1 is the sink. Linear excludes node 20–22; node 19 is the sink. 74
4.12 *G*-ACK batching period Y vs. number of slots required for termination and last packet collected at sink. 76
4.13 Weaver vs. Crystal in the Floor topology. 78
4.14 Weaver vs. Crystal in the Linear topology. 79

5.1 Example time-slotted schedule including n dedicated anchor slots for time synchronization and k slots for tag localization beacons. 88
5.2 Prototypes and evaluation toolchain. 90
5.3 Impact of packaging on clock drift. 92
5.4 Measured (reference, in blue) and simulated clock drift curves. 92
5.5 Anchor deployments. A dark blue square denotes a stationary anchor attached to the ceiling, an orange square stands for a portable anchor, and an X represents a ground-truth landmark. 94
5.6 Localization error vs. synchronization rate in Hall (6 anchors). 95
5.7 Localization error vs. synchronization rate in Corridor (6 anchors, boxed only). Note the different x-axis scale w.r.t. Figure 5.6. 96
5.8 Localization error vs. synchronization rate in real Hall experiments with boxed (left) and naked (right) nodes. The external black crosses are anchors, the internal red ones are landmarks. Error ellipses denote the 3×standard deviation for a given rate. Black dots are individual samples. . 96

5.9 Effect of the number of anchors on the error distribution with a 3.3 Hz synchronization rate in simulation. 98
5.10 GRID: ground truth (blue) vs. estimated position (yellow). 99
5.11 GRID: Number of anchors and localization accuracy. 100
5.12 Estimated trajectories in the CORRIDOR. The color gradient represents time, axis values are in meters. 101
5.13 Estimated trajectory in the ENTRANCE. 102

6.1 Museum target area (top) and map (bottom). Dots are points of interest (POI) in front of exhibits; crosses are UWB anchors on the ceiling. 107
6.2 An (artificial) example of segmentation. G is the set of time intervals of true stops, D the one for estimated stops. 109
6.3 Finding the matching set. 110
6.4 Raw (white) vs. filtered (gray) trajectory. The colored points in the latter fall in ground-truth stop time intervals. 112
6.5 Spatio-temporal view of Figure 6.4 (filtered). 114
6.6 Spatio-temporal characteristics of the dataset. Dashed lines represent the mean and dotted lines the median. 114
6.7 UWB trajectories, ground-truth vs. estimated stops. 115
6.8 *Different* stop properties, *same* unit quality. 117
6.9 *Similar* stop properties, *different* unit quality. 117
6.10 Duration and spatial error for filtered trajectories. 121

7.1 A lander ejects the UWB anchors that will make up the localization infrastructure. From [140]. 126
7.2 Slotframe ownership for an isolated user and for 3 co-located users. 130
7.3 A slotframe schedule example, with one user, three known anchors and one to be discovered. 131
7.4 Time propagation when two user groups merge. Value in brackets is the ID of the time reference user. Synchronization across groups is performed through an anchor: after anchor 4 transmits a message overheard by user 3, the two groups quickly align to reference 1. 134
7.5 Fast discovery and synchronization recovery. Anchor 1 receives the schedule from user 2, but is not in the list of known anchors. Anchor 3 detects synchronization loss when no schedule is received at the beginning of the slotframe. 134
7.6 Model for neighbor churn. 136
7.7 A stationary user (above) and a mobile user (below) range with the neighboring anchors. When the mobile user discovers the same anchors used by the stationary one, they switch to multi-user mode. 143
7.8 Zoom-in of the mobile user, switching from single-user (ranging in all slotframes) to multi-user mode (ranging in a subset of the slotframes). . . . 143
7.8 Users (green) and anchors (blue) in the different connectivity scenarios. The reference user is highlighted in red. 146

7.9 Synchronization time in seconds for all users across different scenarios and varying ND interval. Vertical lines represent the median values. Figure 7.9d shows the improvements when multiple anchors are available to discover the reference and propagate synchronization. 147
7.10 Comparison of the ranging success rate of SONAR and the baseline ($T_{ND} =$ 1 s). 148
7.11 Comparison of the distance estimation of SONAR and the baseline ($T_{ND} =$ 1 s). Boxes represent the 25-75% percentile. Bars represent 0.1-99.9% of the data. 148

List of Tables

2.1 RX errors on a misconfigured link. 22
2.2 *PRR* for 2 co-located networks with different*PRFs*. 28
2.3 Summary of findings. 29

3.1 Operation durations for UWB packets (µs). 39
3.2 Slot durations for UWB and narrowband (NB) in µs. 39
3.3 Glossy vs. GlossyTX: reliability. 41
3.4 Glossy vs. GlossyTX: latency. 41
3.5 Nominal current draw and voltage supply. 43

4.1 Occurrence of failed bootstrap for any node and average energy consumption vs. number B of bootstrap packet retransmissions. Data acquired over $10,000$ epochs in two topologies with no initiator ($U = 0$). 75
4.2 Parameters used for the two configurations of Crystal considered. T_s and T_l are the duration of the T phase optimized for a short (2B) and long (100B) packet, respectively. 77
4.3 Performance of Weaver with 3 mobile nodes. 81

5.1 Localization error for Figure 5.6–5.7 with a 3.3 Hz synchronization rate. . . 98
5.2 Impact of the number of anchors on localization error (cm). 99

6.1 Exploring parameters for filtered trajectories; δ and ϵ are in cm, θ in cm/s. 119
6.2 Segmentation techniques at a glance: Stop-centric metric and additional insights on the nature of false detection. 120
6.3 Comparison of spatio-temporal errors for the various segmentation techniques. 120
6.4 Stop-POI association for true positives, and overall performance for all true stops. 120

7.1 Sample values for the rate of neighbor change, to inform the periodicity of neighbor discovery. 136
7.2 Current draw and duration for each frame type. Write/read refer to the required radio operations in addition to TX/RX. Listen refers to the preamble hunting phase, and is only relevant for anchors, that receive `ND-RESP` and `RNG-INIT`. Idle duration is the portion of the response delay (before `ND-RESP` is sent, or waiting to transmit `RNG-RESP`) in which the anchor is not listening or receiving. 139
7.3 Energy consumption for an anchor in isolated, passive or active mode. Estimated duration of a 10 Ah, 3.7 V battery in each mode and for mixed anchor activity patterns. 141

1

Introduction

The proliferation of wireless communication has profoundly reshaped many industrial sectors, fostering innovative services in mobility scenarios that are now blended in our daily lives: from Internet access at our fingertips through WiFi and cellular technology, to device localization with Global Navigation Satellite Systems (GNSS). The applications made reality by wireless have encouraged disruptive research from industry and academia, towards ubiquitous connectivity and localization. Yet, this vision has not been fully attained, and wired solutions are still necessary in many domains due to strict requirements in terms of reliability, throughput, and latency. In the attempt to grasp the advantages of wireless in more and more application scenarios, a multitude of radio physical layers have emerged, as evidenced by the continuous evolution of standards [6]. In particular, low-power radios can support mobility and ensure long operational lifetime for battery-constrained devices. This makes them one of the key technologies for pervasive Internet of Things (IoT) [7, 8]. In this thesis we focus on ultra-wideband (UWB), a low-power radio technology that can jointly provide communication and localization, and we demonstrate how its unique properties can be exploited to answer emerging challenges in both domains.

Past and future of low-power wireless communication systems. If the benefits of wireless solutions are uncontested, their limitations are also glaringly clear, especially in large-scale, multi-hop systems. In these scenarios, a network of devices connected over the air can be deployed with higher flexibility and lower cost compared to a wired counterpart. However, due to the high consumption of radio chips, these advantages can be easily harvested only if the involved nodes are given access to an unlimited energy supply. When this is not possible and devices must rely on batteries, dedicated techniques are essential to meet communication requirements on reliability and latency, while simultaneously reducing power consumption.

A two-decade quest for dependable, fast, and energy-efficient protocols has led to a plethora of routing strategies, MAC layers, and duty-cycling methods that turn off radios whenever possible, saving energy and ensuring systems can enjoy a long op-

erational life [9, 10, 11]. Due to the adaptability granted by such a wide range of techniques, low-power radios have been put to use in a variety of scenarios, encompassing sensing applications at large [12], and notably in precision agriculture [13], smart homes [14, 15], elderly care [16], smart cities [17, 18], detection of social interactions [19, 20], structural monitoring [21, 22], wildlife tracking [23, 24], eventually appearing in domains traditionally reserved for wired communication, such as industrial control [25].

Low-power wireless is expected to play a pivotal role in the transition towards Industry 4.0, as a core component of the Industrial Internet of Things (IIoT) [26]. Unfortunately, many IIoT requirements are still prohibitive for wireless networks [27]. While high end-to-end reliability can be provided even in multi-hop networks, stringent real-time guarantees are difficult to achieve, especially in dense deployments, with more and more sensors being installed and the ever-growing size of sensed data. Meeting the expected performance would typically come with an unacceptable increase in energy consumption.

Towards accurate, ubiquitous localization. Wireless networks can be used to capture a wide variety of information. In particular, the ability to pinpoint the exact position of an object is rapidly becoming a key asset in many scenarios, and especially in IIoT. Beyond autonomous navigation for manufacturing and smart agriculture, examples include production monitoring and performance evaluation [28], support for digital twins [29], safety monitoring and alerts in construction sites [30], asset tracking and workflow management in healthcare [31, 32]. Ideally, all these location-aware applications could exploit GNSS services, like the Global Positioning System (GPS). However, relying solely on satellite signals is ineffective for multiple reasons.

GPS receivers are known for their high energy consumption, often the primary cause of battery drain in personal devices [33]. While the energy consumption can be reduced, this comes with a trade-off in terms of positioning accuracy [34]. This limitation is true across all applications, but satellite signals may not even reach the receiver in indoor and other "GPS-denied" environments. Moreover, signal multipath due to reflections can cause accuracy degradation, to the point of yielding spurious positions that are practically unusable. This is often the case in city environments, the so-called "urban canyons", with buildings causing reflections and shadowing [35, 36], but also in agricultural fields, where the attenuation of signal power is due to foliage [37, 38]. The unavailability, or unacceptable performance, of satellite-based technology in these conditions has motivated the search for alternative approaches, giving rise to the field of Real-Time Location Systems (RTLS).

Many ad hoc wireless systems have been proposed to enable service indoor, in office environments, industrial facilities, warehouses, subways, and all other areas that are prohibitive for GPS—where localization is nevertheless required. The field of RTLS for GPS-denied areas is currently dominated by radio frequency (RF) techniques, due to their better non-line-of-sight (NLOS) performance compared to Visible Light, and their lower energy consumption w.r.t. acoustic signals approaches [39].

1.1 Ultra-wideband Radios: Localization *and* Communication

Among RF technologies for localization, ultra-wideband (UWB) has steadily gained momentum after its recent resurgence, led by the release of cheap and energy-efficient transceivers, such as the popular DecaWave DW1000 [40]. Unlike those of conventional radio technologies, UWB transmissions are based on short pulses (< 2 ns) whose energy spreads over a large bandwidth. The unique physical layer features of UWB allow the radio to obtain the accurate Time of Arrival (ToA) of received frames. In turn, these timestamps not only make it possible to perform distance estimation (also referred to as *ranging*) with decimeter-level error, but also localization. Several techniques are available, each with its own trade-offs. A mobile node, usually a *tag* attached to the object to track, can compute its own position by ranging with several reference devices (*anchors*). The ranging-based approach is directly applicable in large scale areas, but requires at least two messages per anchor, limiting scalability in the number of tags. When the system has to accommodate many tags, or provide high positioning rate, Time Difference of Arrival (TDoA) schemes are preferred, as a single transmission is sufficient for each position estimation. However, TDoA performance hinges on tight time synchronization among anchors, which requires infrastructure coordination in multi-hop scenarios, like those that we explore in this thesis.

UWB localization infrastructures could be installed outdoors when accurate positioning is required, beyond what GPS-like services can achieve. But most importantly, the exceptional multipath resolution and the robustness to obstacles of this radio technology make it a forefront candidate for localization in indoor or otherwise complex environments [41]. These properties have sparked the interest of researchers for IIoT applications of UWB. UWB has been tested in industrial settings for the navigation of mobile robots [42] and UAVs [43].

Overshadowed by ranging and positioning capabilities, the unique features of UWB communication are often overlooked. One of the main advantages is communication speed: the IEEE 802.15.4 [44] standard specifies a maximum UWB data rate of 27 Mbps. While the DW1000 only supports a data rate up to 6.8 Mbps, it is still significantly faster than what is available with alternative technologies like ZigBee and Bluetooth [45, 46]. This data rate can support high throughput for sensor data, which would normally require wires [47]. UWB transmissions also appear to be more reliable than the counterparts [48] in industrial settings. Another feature of UWB radios is the possibility to transmit data and perform ranging (and localization) simultaneously, as the distance between transmitter and receiver is computed based on the timestamps of the TX/RX of data frames. This dual nature of UWB frames is a valuable asset, since it allows the design of ranging and localization systems on top of communication primitives for data reporting and network coordination, without requiring external hardware and therefore simplifying deployments. Moreover, UWB can co-exist with other communication technologies as it enjoys relatively free frequencies; a recent, notable exception is the 6 GHz band, available to WiFi 6E as per the IEEE 802.11ax standard [49], but countermeasures exist to minimize the impact of interference [50].

However, UWB capabilities do not come without cost. The lifetime of battery-powered UWB devices is a matter of concern for practical deployments. The DW1000 has significantly lower energy consumption compared to WiFi, yet is more energy hungry w.r.t. low-power narrowband radios. While a vast body of research focuses on the achievable ranging and localization accuracy [51], UWB systems are usually designed neglecting energy consumption and evaluated in small-scale settings, failing to adapt to large areas. Together with the lack of fast, reliable and energy-efficient networking strategies, these shortcomings are still hindering the adoption of UWB technology in low-power IoT.

1.2 Goals, Contributions, and Methodology

We aim to advance the state-of-the-art by overcoming the current constraints of UWB communication and localization. The main goal of this thesis is

> *to enable fast, reliable, energy-efficient UWB localization and communication in large-scale areas, through the design of systems that provide seamless operation for nodes moving freely across multi-hop networks.*

The following are the main points of departure w.r.t. traditional UWB systems:

P1. From single-hop to efficient large-scale multi-hop communication. UWB communication has been largely relegated to applications for bulk data transfer in short-range, single-hop links, and is hitherto unexplored in large wireless networks. In this thesis, we analyze UWB physical layer properties to inform our design of communication protocols that provide high end-to-end reliability at a low energy cost. Exploiting the radio's interference rejection capabilities and the high data rate of transmissions, we achieve ultra-fast multi-hop data dissemination and collection.

P2. From small-scale to large-scale localization. A simplistic solution for large-scale localization is to deploy multiple independent infrastructures. However, coordination between these separate networks is cumbersome, and the result is a disconnect for movement traces crossing the various areas. We develop techniques to support seamless movement of target objects to be localized across a large infrastructure, while preserving positioning accuracy. We propose scalable solutions that reduce channel utilization for synchronization and scheduling, leaving the communication medium free for high localization update rate.

P3. From mains-powered to battery-powered anchors. While a large body of literature focuses on reducing the energy consumption of the tag to be localized, few solutions target battery-powered anchors. Those that are available can stretch the system lifetime to months or years, but they do so at the expense of the maximum localization frequency [52]. To support battery-powered anchors without compromising on the update rate, we design protocols for anchors operating under aggressive duty cycle, by means of efficient neighbor discovery and localization

schemes. Our approach enables high-rate localization in environments that were previously precluded to UWB, such as vast agricultural fields, where wired infrastructures are expensive to setup, and battery-powered wireless nodes require maintenance too frequently.

To meet our objectives in **P1**, we first characterize the behavior of UWB frames overlapping in time at the receiver. Because UWB radios can employ different pulse repetition frequencies and coding schemes within the same frequency channels, it is possible, in principle, to set up "parallel" links in the different sub-channels and enhance throughput in communication protocols. We also study the impact of overlapping frames that share pulse repetition frequency and coding scheme, a key step to assess whether communication can sustain contention in the radio medium, but also to verify if UWB can exploit *concurrent transmissions*. While the robustness of concurrent transmissions has already been demonstrated [53, 54] and put to use for the most diverse traffic patterns [55], the technique is still unexplored in UWB radios.

A proper characterization of concurrent transmissions in UWB is necessary for two main reasons. First, UWB encoding, signals and radios are fundamentally different w.r.t. those of other technologies. Second, features unique to UWB like accurate timestamping, and therefore ranging and localization, are potentially affected by concurrent transmissions. We analyze the problem in Chapter 2, where we also provide the necessary background on the UWB technology. The findings, succinctly summarized here, inform the design of the systems presented in the following chapters. Following an experiment-driven approach we show that *i)* different pulse repetition frequencies virtually double the number of non-interfering channels both for communication and ranging, *ii)* concurrent transmissions with different coding are unreliable, unless they are tightly synchronized, *iii)* concurrent ranging exchanges with different coding cause significant outliers, especially for receivers using low pulse repetition frequency, and *iv)* under the same configuration, UWB radios will often report wrong timestamps for ranging, but are very likely to receive one of the frames transmitted concurrently by multiple senders even when their payloads are different, unlocking opportunities similar to those exploited in low-power narrowband radios.

Building on the evidence gathered in support of UWB concurrent transmissions, we investigate the extent to which they can be applied in multi-hop networks rather than the single links we analyze in Chapter 2. Our first target system is Glossy [56], the network flooding protocol that first popularized the technique in narrowband radios. In Chapter 3 we adapt Glossy to UWB radios following a system-driven approach, where techniques and codebases representative of the state of the art are evaluated in a 23-node indoor testbed yielding multi-hop topologies. We show that, once embodied in a full-fledged system, UWB concurrent transmissions yield benefits similar to narrowband, i.e., near-perfect reliability and very low latency and energy consumption, along with order-of-magnitude improvements in network-wide time synchronization—in the order of tens of nanoseconds. Moreover, we observe that UWB dissemination is generally preferable for large packets, in terms of energy consumption. These results pave the way for the exploitation of concurrent transmissions in UWB, which we foster by releasing our systems as open source, enabling

their immediate use and improvement by researchers and practitioners. Indeed, our implementations suggest that existing higher-level protocols built atop Glossy require only minimal adaptation.

Even though Glossy can easily be reused as the building block of other protocols, its fixed-length network-wide floods are *entirely* dedicated to the dissemination of a *single* packet, making its monolithic approach rather inefficient for other traffic patterns. In contrast, in Chapter 4 we present Weaver, a protocol that sets itself apart from the Glossy legacy by breaking down the data forwarding logic to the level of the single transmission slot. In this way, Weaver enables *concurrent* dissemination towards a receiver of *different packets* from *multiple senders* in a *single, self-terminating*, network-wide flood. Weaver intertwines together multiple data flows using a combination of single-hop and network-wide acknowledgments. Our experiments demonstrate that this approach ensures reliable, efficient and ultra-fast data collections, even for large packets. For example, Weaver can collect 30 packets with a payload of 100 bytes, from 30 sources in a 6-hop network, in around 100 ms. While our prototype targets UWB, for which a reference network stack is largely missing, the protocol is generally applicable to any radio supporting concurrent transmissions. As an additional contribution, we design Time Slot Manager (TSM), a thin layer used by Weaver to hide the complexity of network synchronization and simplify the scheduling of operations. Our modular design separates the low-level mechanics of concurrent transmissions from their higher-level orchestration in Weaver, allowing other researchers to easily experiment with alternate designs via our open-source implementation. TSM also comes with a reusable software component, the Radio State Monitor, for the estimation of UWB energy consumption.

In the second part of the thesis, we concentrate on localization systems, starting from the goal described in **P2**. In Chapter 5 we present Talla, a novel wireless-only TDoA approach able to scale over large operational areas without sacrificing positioning accuracy. Talla relies on a TDMA schedule enabling continuous, multi-hop operation of the anchor infrastructure. Few anchor transmissions are sufficient to maintain the TDMA structure, synchronize nodes, and send advertisements for tags to join the schedule, leaving most TDMA slots free and available for localization. We evaluate Talla in our UWB testbed and in much larger (>100 anchors) simulated areas, empowered by a technique that generates synthetic timing information faithfully reproducing the trends of the real one. Our real and simulated results show that Talla achieves decimeter-level accuracy while tracking a moving target across several hops.

The TDoA approach embraced by Talla brings several advantages. The tag is extremely energy-efficient, as it only needs to transmit once for each position estimation. Using a single transmission also means that the system can support multiple co-located tags, each with high localization rate. Exploiting these advantages, in Chapter 6 we present an application case for UWB-based TDoA localization, demonstrating its potential for human movement analytics. Due to the high spatio-temporal resolution of the positions computed by Talla, it is possible to obtain accurate and rich movement information for users carrying a tag, and exploit it to extract higher-level mobility patterns. We focus on the well-known stop-move pattern, and offer

a concrete use case of capturing visits to various exhibits in a real museum. While stop detection has been extensively studied for GPS data, the exploitation of UWB trajectories for detecting these patterns is largely unexplored. We study *i)* whether existing stop detection techniques, developed for coarser-grained localization, apply also to UWB trajectories, and *ii)* the quantitative extent to which this enables finer-grained analyses. To compare representative stop-move detection techniques, we define a novel evaluation metric suited to the high spatio-temporal resolution of UWB. We base our analysis on 70000+ positions and 200+ ground-truth stops to exhibits. These stops are very close in space, with exhibits less than 1 m apart, and have very short duration, down to 10 s. Yet, results confirm very accurate spatio-temporal estimation in the vast majority of cases. While the UWB infrastructure used in our tests covers a specific area of the museum of 25×15 m^2, the multi-hop approach of Talla makes movement analysis readily applicable to larger areas.

Talla targets indoor scenarios, using mains-powered anchors to ensure high localization rate for many tags. As stated in the main goals stated at the beginning of this section, this approach cannot be easily applied outdoors, or when deploying a full infrastructure is inconvenient. To tackle this issue and address **P3**, in Chapter 7 we introduce Sonar, a system specifically designed for battery-powered anchors. Sonar supports ranging-based localization by providing a way for tags to discover surrounding anchors and estimate pairwise distances with minimal energy expenditure on the anchor side. Sonar relieves anchors from all network coordination duties, exploiting neighbor discovery and ranging messages for all other operations like synchronization, scheduling, and contention resolution. By limiting the number of messages in the system and only activating anchors when a tag is in range, the infrastructure as a whole can remain in the lowest power mode ("deep sleep") most of the time. This approach can jointly achieve high ranging and localization rate while still providing years-long anchor lifetime, further expanding the application landscape of UWB.

Despite its recent growth, we believe UWB is still underutilized considering its features for communication and localization. By exploiting concurrent transmissions for communication, and accounting for scale and energy constraints in the localization domain, the techniques proposed in this thesis can make UWB an appealing technology in an ever-increasing number of scenarios. We offer our concluding remarks in Chapter 8, discussing how our contributions can push the envelope of UWB research and what opportunities and challenges await in the future of UWB.

Part I

Fast, Efficient, Large-scale Ultra-wideband Communication

2

Playing with Fire: Exploring Concurrent Transmissions in Ultra-wideband Radios

Concurrent transmissions, i.e., the sending of packets overlapping in space and time, are a fundamental building block of modern protocols for wireless networks.

Concurrent transmissions *over different frequency channels* have been exploited for decades as a simple means to achieve higher scalability w.r.t. users and traffic rates, and avoid interference among senders. For instance, the Time-Slotted Channel Hopping (TSCH) [44], increasingly popular in Internet of Things (IoT) networking, is one of many protocols exploiting the PHY-level separation of IEEE 802.15.4 channels.

In contrast, concurrent transmissions *on the same channel* are commonly considered harmful, as they generally result in packet loss. Nevertheless, this is not always true. Under some tight synchronization requirements, PHY-level radio properties enable reliable packet reception of one of the concurrently transmitted packets, despite interference from the others. This fact has recently been exploited by several protocols yielding unprecedented reliability and performance [57, 58, 56, 59, 60, 61].

These two nuances of concurrent transmissions are well-studied in the context of mainstream IEEE 802.15.4 narrowband radios (e.g., the popular CC2420 [62]). Nevertheless, very little is reported about them in the context of IEEE 802.15.4 ultra-wideband (UWB) radios.

This chapter revises our publication [1]: D. Vecchia, P. Corbalán, T. Istomin, and G. P. Picco. "Playing with Fire: Exploring Concurrent Transmissions in Ultra-wideband Radios". In *Proceedings of the* 18^{th} *IEEE International Conference on Sensing, Communication and Networking (SECON)*. 2019. DOI: 10.1109/SAHCN.2019.8824929.

The return of UWB. This technology has returned to the forefront of research and market interest after a decade of oblivion, thanks to the recent availability of a new generation of UWB chips significantly smaller, cheaper, and less energy-hungry than their predecessors, spearheaded by the popular DecaWave DW1000 [63]. Moreover, UWB radios can also estimate distance with decimeter-level accuracy. The possibility of using a single radio chip for communication and ranging, and the fact that IEEE 802.15.4 includes an UWB PHY layer, has attracted significant interest in several IoT scenarios.

Concurrent transmissions in UWB? A large body of work [64, 65] showed that, in theory, the physical encoding of UWB packets allows for non-interfering concurrent transmissions on different links on the same channel, regardless of the power of the signals involved or their synchronization. This is potentially disruptive, as it would mean that properly configured communication links could operate on their own dedicated "virtual" channels, well beyond the limited number of frequency channels—a formidable asset for designing scalable, reliable, and efficient network protocols.

In practice, however, the situation is less clear. The IEEE 802.15.4 standard [44] defines a *complex channel* as composed by a frequency channel and a preamble code, hinting at the fact that different combinations of these two configuration parameters should yield non-interfering communication links. Instead, the documentation of the IEEE 802.15.4-compliant DW1000 states that non-interfering communication links must be obtained by configuring a different pulse repetition frequency (*PRF*). When using different preamble codes (and the same *PRF*), they report that *"there is still a small amount of cross correlation [...] This may mean that it is not possible to achieve the separation envisioned by the standard's authors"* (p. 218, [63]). These two guidelines are partially at odds, failing to provide a clear indication to the protocol designer. Further, neither document reports quantitative information about the effect of a given configuration.

More generally, while the real-world characteristics of concurrent transmissions in narrowband radios are well-understood for both the different- and same-channel configurations, this is unfortunately not the case for UWB.

Goals, methodology, and contribution. The goal of this chapter is precisely to fill this gap by *ascertaining the extent to which concurrent transmissions can be exploited in UWB*.

We first give an overview of UWB technology in §2.1, encompassing the physical layer, the devices we use in our experiments, and the principles behind ranging. We reference the provided UWB background throughout the dissertation.

We then list research questions (§2.2) aimed at dissecting the conditions under which concurrent transmissions are possible. We focus on communication links in the same collision domain and on the same frequency channel. We specifically investigate the effect of three configurations: *i)* different *PRF ii)* same *PRF* but different preamble codes *iii)* same *PRF* and preamble codes. The first two configurations are aimed at obtaining separate, non-interfering complex channels, following the recommendations of DecaWave and IEEE 802.15.4, respectively. Instead, the third one yields

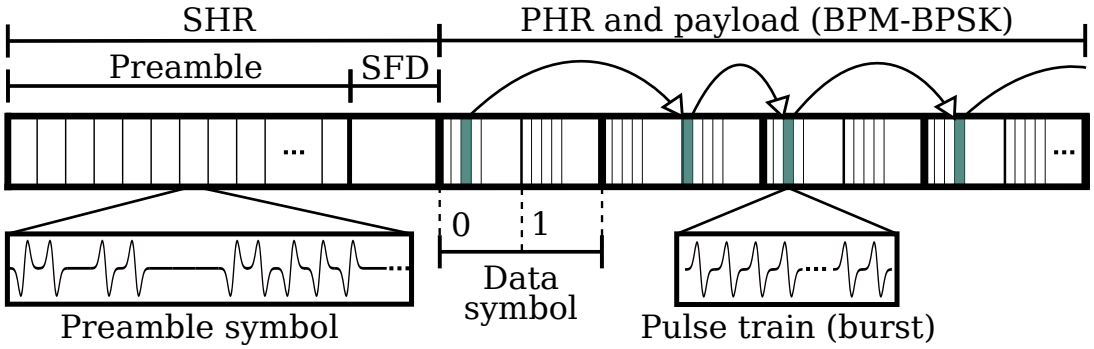


Figure 2.1: UWB frame: the SHR is encoded in single pulses, while the data part exploits BPM-BPSK modulation. Preamble codes determine the preamble symbol sequence and the time-hopping code (arrows) for data transmission.

interfering links, enabling us to investigate the presence of PHY-level effects akin to those in narrowband radios [58, 56].

We achieve our goal *experimentally* with a real-world dedicated setup (§2.3) where we control precisely the synchronization among nodes and therefore the degree of concurrency, i.e., temporal overlapping among transmitted packets.

Using this setup, we derive empirical observations (§2.4) on the reliability and performance of *both communication and ranging* for each of the configurations above. To the best of our knowledge, ours is the first study that *quantitatively* investigates both dimensions in the context of concurrent transmissions, and across different notions thereof.

Based on these empirical observations, we distill higher-level findings to *inform the design of networking and ranging protocols* and exemplify opportunities for their application in practice (§2.5), hopefully inspiring a new generation of UWB networking and ranging protocol stacks.

The chapter ends with a concise survey of related work (§2.6) followed by brief concluding remarks (§2.7).

2.1 Ultra-wideband Communication and Ranging

We provide the necessary background on impulse radio UWB (IR-UWB), encoding, frame timestamping and distance estimation. The IEEE 802.15.4 standard defines the structure of an UWB frame, the embedded error correction mechanisms and the decoding procedure. Here we briefly introduce the standard UWB physical layer and provide the most important details on the DecaWave DW1000 chip used in our evaluation.

Impulse radio. Modern UWB radios are impulse-based. IR-UWB spreads the signal energy across a very large bandwidth by transmitting data through a time-hopping sequence of ns-level pulses [66]. This reduces the power spectral density, the harming interference affecting other wireless technologies, and the impact of multipath components (MPC). The large bandwidth provides high time resolution, enabling

UWB receivers to precisely estimate the time of arrival of a signal, and therefore distance. Time-hopping codes [64] can be used to provide multiple access to the medium. These features make IR-UWB ideal for ranging and localization and also for low-power communication.

UWB PHY layer. The IEEE 802.15.4-2011 standard [67] specifies an UWB PHY layer based on impulse radio. An UWB frame (Figure 2.1) is composed of *i)* a synchronization header (SHR) and *ii)* a data portion. The SHR is encoded in single pulses and includes a preamble for synchronization and the start-of-frame delimiter (SFD). The data portion, instead, exploits a combination of burst position modulation (BPM) and binary phase-shift keying (BPSK), and includes a physical header (PHR) and the data payload. The duration of the preamble is configurable and depends on the number of repetitions of a predefined symbol. A preamble symbol (Figure 2.1) consists of a sequence of elements drawn from a ternary alphabet $\{+1,\ 0, -1\}$, i.e., positive, absent, and negative pulse. This sequence is determined by the preamble code. The standard defines preamble codes of 31 and 127 elements, which are then interleaved with zeros according to a spreading factor. This yields a (mean) *pulse repetition frequency* (*PRF*) of 16 MHz or 64 MHz, respectively; these values, hereafter *PRF*16 and *PRF*64 for readability, are also configurable. The SFD sequence is also made of ternary elements, obtained by the product of preamble symbols and dedicated codes of length 8 or 64. The SFD indicates the beginning of BPM-BPSK modulation for the PHR and data portion. The IEEE 802.15.4-2015 standard [44] describes the sequence of steps for the creation of the UWB waveform to be transmitted for each radio configuration, starting from the input payload. After inserting forward error correction (FEC) bits in the payload and physical header (PHR), data undergoes convolution coding, BPM-BPSK modulation and time-hopping spreading to obtain data symbols.

BPM-BPSK data symbol. BPM-BPSK modulation is employed for the header and payload. Each BPM-BPSK data symbol (Figure 2.2) carries two bits of information, but only represents one input bit due to convolution coding. The data symbol of duration T_{dsym} is partitioned in two halves of duration T_{BPM}, where only one of the two halves is meant to host a burst (or pulse train). The number of pulses in the burst depends on the configured data rate. For the 6.8 Mbps rate used in this work, a burst is made of two pulses. If the radio transmits the pulse burst in the first half, it is interpreted as a 0 bit, 1 otherwise. The second bit is encoded by the phase (polarity) of said burst. Guard intervals, in which pulses are never transmitted, are placed between possible burst positions to serve as protection from high-energy multipath signal components, preventing inter-symbol interference. The combination of BPM and BPSK modulation schemes supports both coherent and noncoherent receivers, as the latter are unable to extract polarity information but can still decode based on the burst positions. To allow multi-user uncoordinated access, the location of pulses within a T_{BPM} duration is defined by a pseudo-random time-hopping sequence.

Complex channels for multiple access. The time-hopping sequence in the transmission of the data part is derived from the same preamble codes used to form the preamble and the SFD. Therefore, different codes also decrease the chance of destructive interference for the data portion. Preamble codes were thus envisaged as a

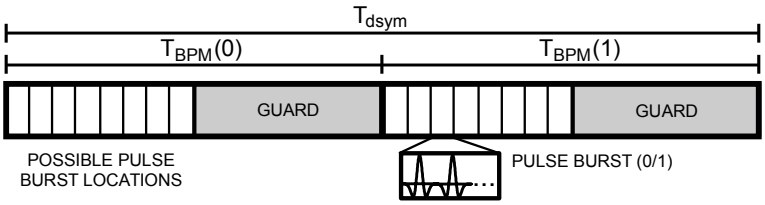


Figure 2.2: UWB data symbol.

mechanism to provide multiple non-interfering access to the wireless medium. However, according to DecaWave[68], frames that overlap in different complex channels ⟨frequency, code⟩ may still interfere with each other unless their codes have different *PRFs*.

Channel impulse response (CIR). The perfect periodic autocorrelation of the preamble code sequence enables coherent receivers to determine the CIR [63], which provides information about the multipath propagation characteristics of the wireless channel between a transmitter and a receiver. The CIR allows UWB radios to distinguish the signal's first path from MPC and accurately estimate the time of arrival of the signal, by means of the internal leading edge detection algorithm (LDE). In §2.4, we exploit CIR information to analyze the interference created by concurrent transmissions under different RF configurations.

Two-way ranging (TWR). The IEEE 802.15.4 standard also specifies two TWR schemes to estimate the distance between two nodes, an initiator and a responder. In the simplest one, single-sided TWR (SS-TWR), the initiator sends a POLL message to the responder, storing the TX timestamp t_1. After an assigned delay T_{RESP}, the responder replies back with a RESPONSE, embedding in the payload the RX timestamp of the POLL, t_2, and the predicted TX timestamp of the RESPONSE, t_3. The initiator then measures t_4, the RESPONSE RX timestamp, computes the time of flight as $\tau = \frac{(t_4-t_1)-(t_3-t_2)}{2}$, and estimates the distance $d = \tau \cdot c$ between the nodes, where c is the speed of light in air. The scheme is shown in Figure 2.3. SS-TWR suffers from clock and frequency drift [69]. Symmetric double-sided TWR (DS-TWR), also part of the standard, mitigates their impact but requires more message exchanges. All the timestamps required for ranging are measured in a packet at the *ranging marker* (RMARKER), which marks the first pulse of the PHR after the SFD. In this chapter, we focus on SS-TWR and analyze how different RF settings can be exploited to perform multiple ranging exchanges concurrently.

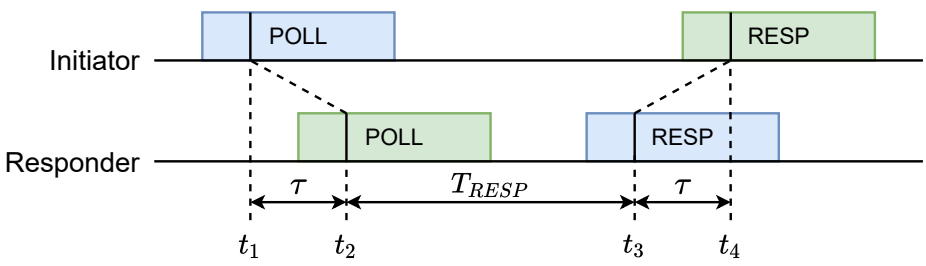


Figure 2.3: Single-sided two-way ranging.

DecaWave DW1000 and EVB1000. The DW1000 is a standard-compliant fully-coherent UWB transceiver, commercialized by DecaWave. The DW1000 supports frequency channels $\{1, 2, 3, 4, 5, 7\}$, each with two different pulse repetition frequencies (*PRF*), nominally 16 MHz and 64 MHz. Three data rates are available: 110 kbps, 850 kbps, and 6.8 Mbps. Channels $\{4, 7\}$ have a larger 900 MHz bandwidth while the others are limited to 499.2 MHz. The DW1000 measures the CIR with a sampling period of $T_s = 1.016$ ns upon preamble reception, storing it in a large 4096B buffer available to the firmware developer. The DW1000 requires an external 38.4 MHz oscillator, with a tolerance of ± 20 ppm [70]. This reference clock is used as phase-locked loop (PLL) input to obtain a frequency of 125 MHz, allowing packets to be scheduled for delayed transmission with a resolution of 8 ns. It is important to emphasize that, while transmissions can only be scheduled based on the PLL frequency, this has no impact on the accuracy of TX timestamps. On the receiver side, the DW1000 provides accurate, sub-ns RX timestamps as well. Relying on the estimated CIR, the LDE algorithm finds the time of arrival with a 15.65 ps resolution. The programmer can trim the crystal frequency with a step that depends on the platform capacitors. The step is ~ 1.45 ppm for the EVB1000 platform we use in our experiments.

Reception errors. UWB transmissions employ forward error correction in the form of convolution coding. The PHR and data payload also employ several mechanisms to detect and correct errors. The PHR includes a 6-bit parity check SECDED (single error correct, double error detect) field. The data payload, instead, employs a Reed-Solomon (RS) encoder that appends 48 parity bits every 330b of data. Hence, uncorrectable bit errors in the PHR or the data payload trigger SECDED and RS errors, respectively. Additionally, a 2-bytes CRC sequence is appended at the end of the frame. All errors are reported in the status register of the DW1000. Moreover, DW1000 radios can also trigger an SFD timeout when a preamble is detected and the SFD is not received within a configured time interval, usually set to the expected SHR duration. In §2.4, we analyze these errors to understand the reasons behind packet loss.

2.2 Research Questions

Concurrent transmissions are a complex, multi-faceted topic. In this section, we concisely outline the intertwined research questions motivating this analysis and answered in §2.4.

Q1: What are the key configuration settings yielding non-interfering complex channels?

As already mentioned, the *PRF* and preamble code values play a key role, but neither the DecaWave documentation nor the IEEE 802.15.4 standard provide an exhaustive answer about how to reliably define complex channels. Therefore we explore both configurations hinted in these documents: *i)* different *PRFs* and preamble codes, and *ii)* same *PRF* and different preamble codes.

Answering this question is key, as UWB frequency channels in commercial chips are fewer than supported. For instance, the DW1000 offers only 6 channels, although IEEE 802.15.4 defines 16 channels for both its narrowband and UWB PHY layers.

Therefore, using complex channels, instead of only frequency channels, significantly increases the degrees of freedom in scheduling non-interfering concurrent transmissions.

Q2: Can concurrent transmissions be reliably exploited even on the same *complex channel?*

To answer this question we experiment also with links configured with the same *PRF* and preamble codes. This allows us to ascertain whether the recent results [57, 58, 56, 59, 60, 61] exploiting concurrent transmissions on the same channel in IEEE 802.15.4 narrowband can be transferred or adapted for UWB, and under what conditions.

Q3: How concurrent the concurrent transmissions can be?

Or, in other words, what is the tolerable amount of overlapping among two transmissions? Answering this question yields precious information to the protocol designer, as it determines the level of synchronization (or de-synchronization) required to prevent performance degradation.

Q4: Is the outcome affected by the relative power of the concurrent signals?

Network nodes are typically configured with the same TX power; nevertheless, the different relative node positions, combined with the well-known path loss attenuation, may induce significant differences in the power of signals concurrently transmitted. This difference in power plays a key role in determining the capture effect in IEEE 802.15.4 narrowband radios [58, 56]; it is therefore worth investigating if similar constraints exist in UWB in the context of Q2. Furthermore, it is also worth investigating the answer of this question in the context of Q1, to ascertain if the relative difference in signal power plays a role in determining non-interference across complex channels.

Q5: Is the outcome affected by the number of concurrent sources or the transmitted packet?

These factors are known to affect concurrent transmissions on the same channel in narrowband IEEE 802.15.4. Therefore, this question relates to Q2 in ascertaining similarities and differences w.r.t. UWB.

We exploit our experimental setup (§2.3) to answer these questions via empirical observations (§2.4) focused on *both communication and ranging* as characterized by the following **metrics**: *i)* packet reception rate (*PRR*), i.e., the ratio of packets successfully received over those sent *ii)* ranging error, and *iii)* ranging reliability, i.e., the ratio of successful ranging exchanges over those performed.

2.3 Experimental Setup

Hardware and testbed. We run experiments in a testbed deployed in the ceiling above the corridors of an office building at our premises. We employ the DecaWave EVB1000 platform [71], featuring an STM32F105 MCU and the DW1000 UWB transceiver with a PCB antenna. Each EVB1000 is connected to a JTAG programmer and a Raspberry Pi. This setup allows us to easily schedule and run numerous experiments without the effort required to manually deploy the nodes.

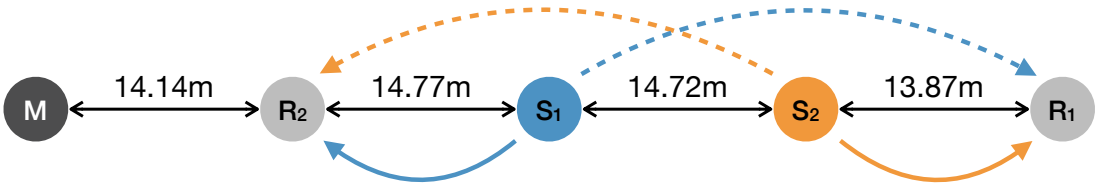


Figure 2.4: Network topology of our experiments. M, S_i, and R_i are the master, sender, and receiver (responder), respectively. All nodes are in communication range. The arcs represent the links under study: *weak* (dashed line) and *strong* (solid).

Network topology. Unless otherwise noted, we use a 5-node subset of the testbed (Figure 2.4) in the same collision domain, with different node roles: *i)* M, synchronization master *ii)* S_i, sender (or SS-TWR initiator) *iii)* R_i, receiver (or responder).

Depending on the senders and receivers chosen, and their relative distance, we can explore different relative signal strengths between transmissions, therefore answering question Q4. Hereafter, we refer to *strong* links as those where the sender transmits towards its nearest receiver ($S_1 \rightarrow R_2$, $S_2 \rightarrow R_1$) and, dually, *weak* links as those where it transmits to the farthest ($S_1 \rightarrow R_1$, $S_2 \rightarrow R_2$).

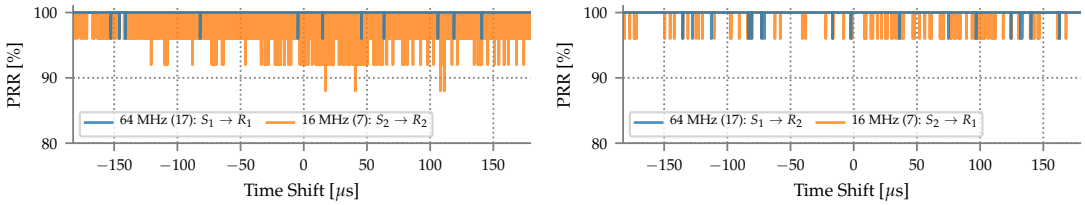
Time (de)synchronization. At the start of each message round, the master M broadcasts a synchronization frame. Senders read the RX timestamp of the synchronization frame and schedule their transmission to start after a given delay. Responders log the received packets and RX errors and, for ranging, transmit their RESPONSE.

To assess the impact of time (de)synchronization, and address question Q3, we also apply a $\Delta t \in [-183\,\mu s, 183\,\mu s]$ in steps of 32 ns to the transmission of sender S_2, therefore controlling the time overlapping among packets. We checked that reducing the step to the supported minimum of 8 ns does not yield new observations though slows down the experiments.

For each time shift, we transmit 25 messages, reporting results from $>$ 296k packets per link and configuration tested. For ranging, we change the time shift range to $\Delta t \in [-511\,\mu s, 511\,\mu s]$ in steps of 250 ns to account for the overlap of the end of RESPONSE from one link with the beginning of POLL from the other. For each time shift, we perform 10 SS-TWR exchanges, resulting in $\geq$ 39k ranging exchanges per link.

UWB settings. We consider UWB channels 2 and 4 with center frequency f_c = 3993.6 GHz and bandwidth of 499.2 MHz and 900 MHz, respectively. We present results for *PRF*16 and *PRF*64 and with preamble codes $\{3, 4, 9, 10\}$ (channel 2) and $\{7, 8, 17, 18\}$ (channel 4). We set the DW1000 to employ the 6.8 Mbps data rate with a preamble length of 128 symbols.

Implementation. We developed our firmware atop Contiki [72] for the EVB1000 platform [73]. At the beginning of the experiments, nodes remain idle for 60 s to allow the clock to stabilize. To ensure the expected overlapping between frames, we re-synchronize all nodes at the beginning of each round, and we compensate the known difference in signal propagation time between the master M and the other nodes. We leave a 1.5 ms delay between the reception of the synchronization frame



(a) Weak links. **(b)** Strong links.

Figure 2.5: *PRR* with different *PRFs* (channel 4). Concurrent transmissions exploiting different *PRFs* are very likely to be received correctly, especially with *PRF*64.

and the first transmission round to account for the time required to switch RF configuration and transmit a preamble. Our SS-TWR implementation sets the RESPONSE delay to $T_{RESP} = 320\,\mu s$ to minimize the impact of clock drift on ranging estimation. In our testbed, we measured a typical clock drift ≤ 3 ppm; this yields a potential desynchronization up to 4.5 ns, negligible as it is < 8 ns, the DW1000 TX scheduling precision [63].

2.4 Empirical Observations

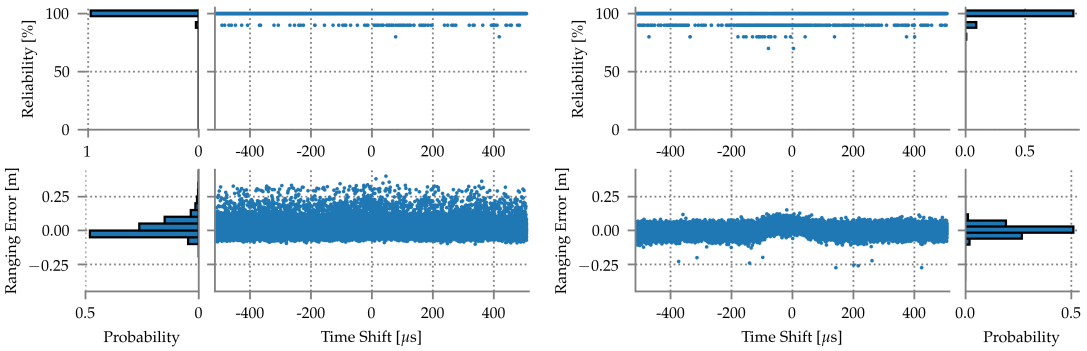
We present our empirical observations, aimed at answering the research questions in §2.2 using the setup in §2.3. We first establish a baseline for communication and ranging by analyzing their performance with isolated transmissions (§2.4.1). We then structure the core of this section around the configurations we explore concurrent transmissions with: *i)* different *PRFs* (§2.4.2) *ii)* different preamble codes within the same *PRF* (§2.4.3) *iii)* exact same RF settings (§2.4.4). Finally, we investigate their combination (§2.4.5).

2.4.1 Baseline: Isolated Transmissions

We first establish the baseline performance of each link *in isolation*, i.e., without concurrent transmissions. The results of communication experiments are averaged over 10000 packets sent by each sender. Across all experiments, we obtain $PRR \geq 99.92\%$ and $PRR \geq 99.99\%$ for *PRF*16 and *PRF*64, respectively. When the sender is closest to the expected receiver (strong links) both *PRFs* achieve 100%. For ranging, we calibrate the antenna delay for all configurations, obtaining zero-mean error with standard deviation $\sigma \leq 4.5$ cm.

2.4.2 Concurrent Transmissions with Different PRFs

Communication reliability. Figure 2.5 shows the *PRR* on channel 4 for each link and *PRF* combination vs. the applied time shift. Overall, with *PRF*64 we obtain $PRR \geq 96\%$ irrespective of the time shift, while *PRF*16 achieves a slightly lower $PRR \geq 88\%$. When the sender is farther from the intended receiver (Figure 2.5a), only 11 packets were lost out of 296800 with *PRF*64, yielding an average $PRR = 99.996\%$ despite the concurrent transmissions with *PRF*16. The latter lost 1722 packets, yield-



(a) Channel 4, *PRF*64, code 17: $S_1 \rightarrow R_1$.

(b) Channel 4, *PRF*16, code 7: $S_2 \rightarrow R_2$.

Figure 2.6: Concurrent ranging with different *PRFs*. Despite interference, both *PRFs* perform accurate ranging reliably.

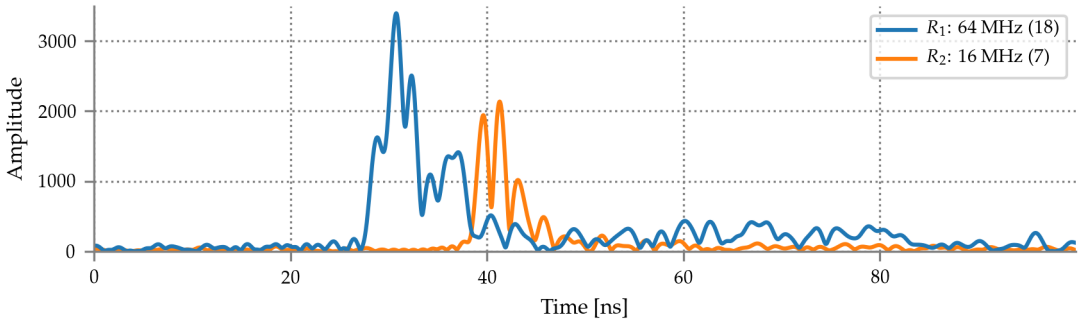
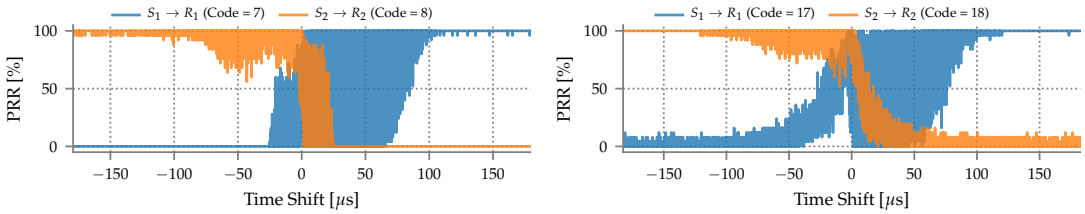


Figure 2.7: CIR with concurrent transmissions using different *PRFs* and $\Delta t = -20.513\,\mu s$.

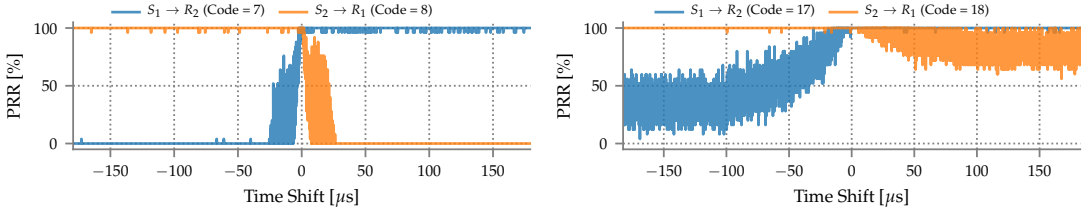
ing $PRR = 99.42\%$. When the sender is closer to the receiver (Figure 2.5b), only 17 and 103 packets were lost for *PRF*64 and *PRF*16, respectively, yielding $PRR \geq 99.97\%$. We obtain similar results with other preamble code combinations on channel 2 and 4. The higher reliability of *PRF*64 is the result of the higher amount of pulses per preamble symbol [74]. This comes, however, at a cost in terms of energy. The few packet losses obtained are mostly the result of SECDED and RS errors, i.e., non-correctable bit errors in the PHY header or in the payload. Overall, we observe that concurrent transmissions through different *PRFs* are reliable regardless of the time (de)synchronization and the physical arrangement of the network.

Ranging Reliability and Error. Figure 2.6 shows the ranging reliability (top) and the ranging error (bottom) over the applied time shifts for the two weak links ($S_1 \rightarrow R_1$ and $S_2 \rightarrow R_2$) on channel 4. With *PRF*64, we obtain an average ranging error $\mu = 1$ cm with standard deviation $\sigma = 5$ cm. *PRF*16 yields $\mu = 0.2$ cm with $\sigma = 3$ cm. These results are in accordance with the baseline in isolation (§2.4.1), indicating that performing ranging concurrently with two different *PRFs* has no impact on accuracy. With *PRF*16, however, we observe a minor overestimation of the ranging distance for time shifts $\Delta t \in [-50\,\mu s, 50\,\mu s]$. On channel 2, *PRF*16 also presents some extreme, although rare, outliers in the same region. The error was > 1 m for only 0.04% of the ranging samples. For both channels and *PRFs*, we obtain a ranging reliability



(a) *PRF*16. **(b)** *PRF*64.

Figure 2.8: *PRR* on the *weak* links (channel 4) and different preamble codes for *PRF*16 (2.8a) and *PRF*64 (2.8b). Concurrent transmissions with different preamble codes introduce significant packet loss, especially for the late transmission.



(a) *PRF*16. **(b)** *PRF*64.

Figure 2.9: *PRR* on the *strong* links (channel 4) and different preamble codes for *PRF*16 (2.9a) and *PRF*64 (2.9b).

$\geq$ 98.27%. *PRF*16 is slightly less reliable than *PRF*64. The minimum reliability for a given time shift was 60% with *PRF*64 and channel 2. The minor loss in reliability w.r.t. the *PRR* in the communication experiment is expected as each SS-TWR exchange requires *two* packets. Figure 2.7 shows the CIRs measured by R_1 and R_2 with both senders transmitting concurrently and $\Delta t = -20\,\mu s$. Both CIRs exhibit a clear line-of-sight path followed by some strong MPC, without impact from concurrent transmissions. As a result, both receivers can distinguish the first path and measure distance accurately.

2.4.3 Concurrent Transmissions with Different Preamble Codes

Communication reliability. Figure 2.8 and 2.9 show the *PRR* of each *weak* and *strong* pair, respectively, for each applied time shift. In contrast to the case with different *PRFs*, concurrent transmissions with different preamble codes introduce significant packet loss, decreasing reliability across the time shifts applied and RF settings studied to $42\% \leq PRR \leq 53\%$. We observe, however, that the *early* packet is likely to be successfully received at the intended destination, especially if the packet is sent $\geq$100 µs earlier that the interfering packet. In this case, the end of the preamble or the data portion of the early packet overlaps with the beginning of the preamble of the late packet, bearing reduced impact in the successful reception of the first. This observation could be exploited, e.g., to give priorities to different packets, allowing high priority transmissions to start sufficiently early. The low cross-correlation between preamble codes allows both receivers to synchronize with the early preamble,

Table 2.1: RX errors on a misconfigured link.

Error	PRF16	PRF64
SFD t/out	0.28%	0.03%
SECDED	81.55%	86.45%
RS	18%	13.29%
CRC	0.16%	0.22%

decreasing the probability of reception for the late packet.

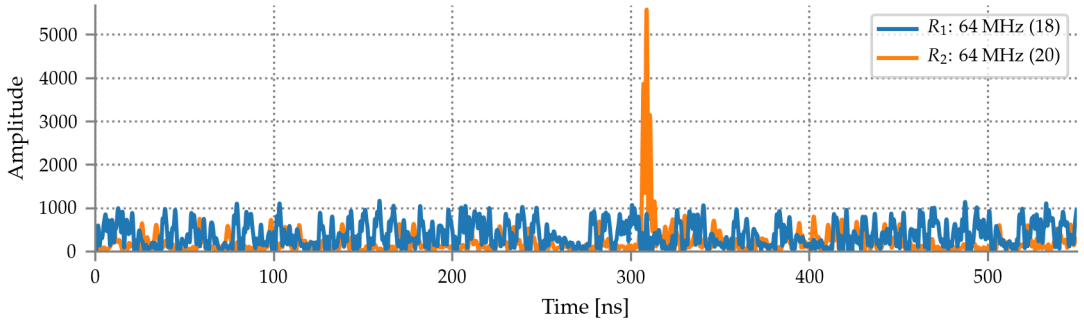
When the sender is close to the receiver and the signal is stronger (Figure 2.9), transmitting synchronously with the interfering signal or slightly earlier provides high reliability, which underlines the importance of the relative signal strength among concurrent transmissions (Q4). If the interfering signal is weaker and frames are precisely synchronized, we obtain high *PRR* for each link. With *PRF*64 and $\Delta t < 10\,\mu s$, the overall *PRR* was 99.81%. We noticed a clear asymmetry (Figure 2.9b) between the two links; packets transmitted from sender S_2 are more likely to be received than those from S_1. This is the result of the slightly different distance among nodes (Figure 2.4); we verified it by temporarily moving nodes to the same distance, obtaining more symmetric performance.

CIR analysis. We resort to the measured CIRs to understand the reasons behind the performance degradation w.r.t. the case with different *PRFs*. Figure 2.10 shows the CIR estimated by each receiver for various time shifts. When S_2 transmits sufficiently early (Figure 2.10a), the intended receiver (R_2) receives numerous preamble symbols without any interference, accumulating enough energy for the line-of-sight peak to emerge from other minor MPC and noise. The same occurs, reversed, with link $S_1 \rightarrow R_1$ (Figure 2.10c). In these cases, the early preamble can be easily detected and the packet is likely to be received correctly. The late transmission, however, suffers strongly from interference of the other, yielding minor peaks throughout the entire CIR span that hinder precise synchronization and first-path estimation. This effect is exacerbated in Figure 2.10b, where transmissions are more synchronized and it is more difficult to discern the right peak.

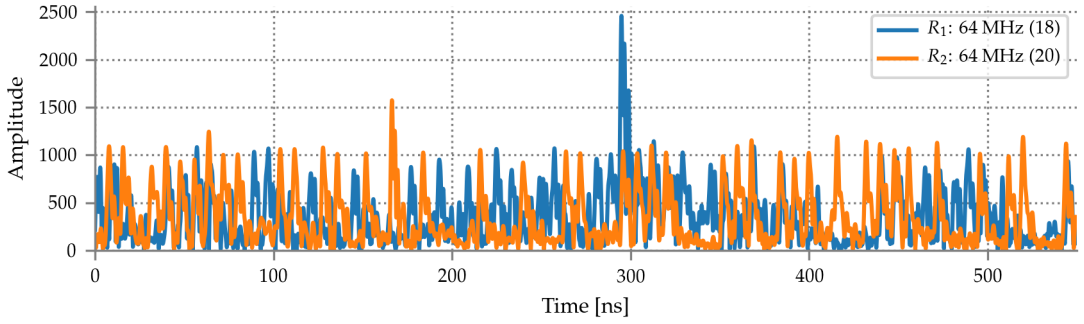
Understanding packet loss. Table 2.1 reports results from an isolated link we misconfigured to use different preamble codes for sender and receiver. As expected, no packet was received correctly, but with counter-intuitive sources of error (§2.1).

Given the different codes and CIR signals (Figure 2.10) we expected mostly SFD timeouts or no preamble detection. Instead, 99% of the errors (with either *PRF*) are due to SECDED or RS parity checks, i.e., non-correctable bit errors in the PHR and data payload, respectively. This suggests that, despite the low cross-correlation of preamble codes, the receiver synchronizes to the preamble sent with the different code and is even able to detect the SFD. If preamble codes were fully orthogonal, a receiver would be able to distinguish the different preamble codes, rejecting or ignoring the mistaken preamble.

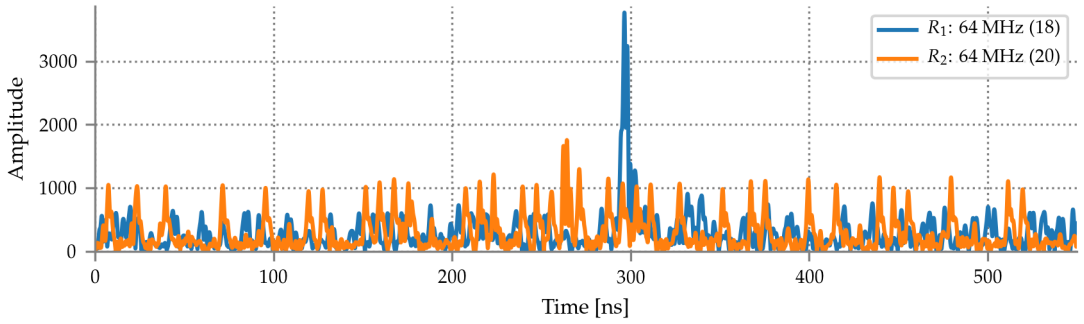
Ranging reliability and error. From the results on communication, it follows that



(a) $\Delta t = -41\,\mu s$.



(b) $\Delta t = 10\,\mu s$.

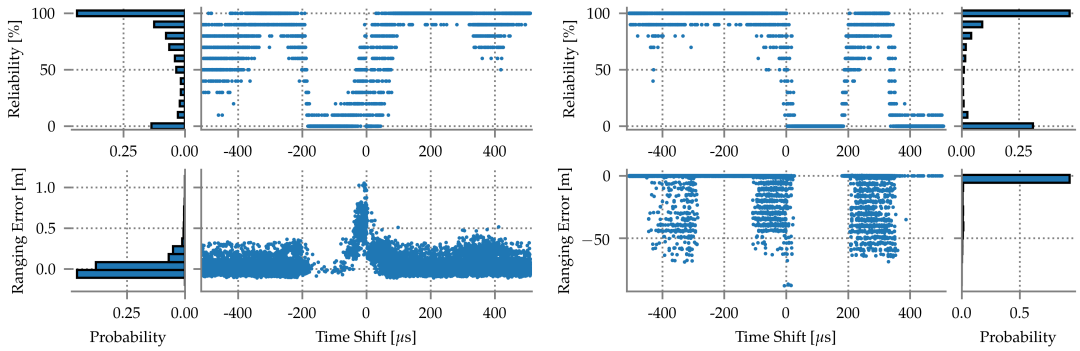


(c) $\Delta t = 41\,\mu s$.

Figure 2.10: CIR for various time shifts Δt with concurrent transmissions using different preamble codes.

many ranging rounds cannot be completed because nodes detect the first preamble transmitted, even if it is the one used by the other link. Figure 2.11 illustrates the reliability of ranging rounds and the measurement error, depending on the time shift. The success rate follows similar patterns for all configurations. Note that initiators transmit the POLL, switch off their radio, and wake up just in time for the expected RESPONSE. For $S_1 \rightarrow R_1$ (Figure 2.11a), when S_2 initiates the ranging exchange earlier than S_1, the responder R_1 misses the POLL of S_1 because it is receiving the preamble of the POLL from S_2. This mismatch causes the failure of rounds in the range $[-200\,\mu s, 0\,\mu s]$, where the responder cannot recover from the RX error fast enough to receive its intended packet. This behavior is mirrored for S_2 (Figure 2.11b) in $[0\,\mu s, 200\,\mu s]$. The other relevant drops in reliability are caused by similar interactions w.r.t. RESPONSE.

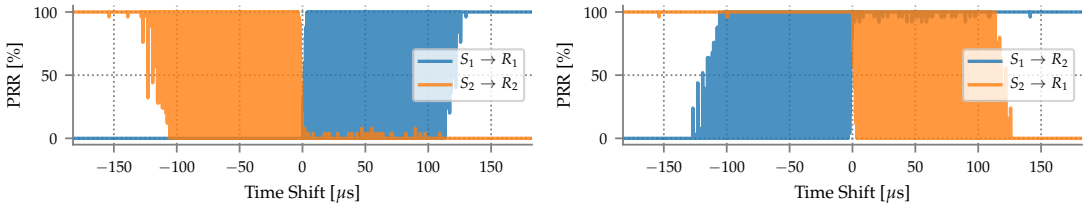
As for ranging error, *PRF64* generally achieves accurate distance estimates. Instead,



(a) Channel 4, *PRF*64, code 17: $S_1 \rightarrow R_1$.

(b) Channel 4, *PRF*16, code 8: $S_2 \rightarrow R_2$.

Figure 2.11: Concurrent ranging with different preamble codes. Significant outliers appear especially with *PRF*16. Many ranging rounds are lost due to interference and RX errors.



(a) *Weak* links: the intended signal is weaker than the interfering one.

(b) *Strong* links: the intended signal is stronger than the interfering one.

Figure 2.12: *PRR* with the same RF configuration (channel 4, *PRF*64, code 17).

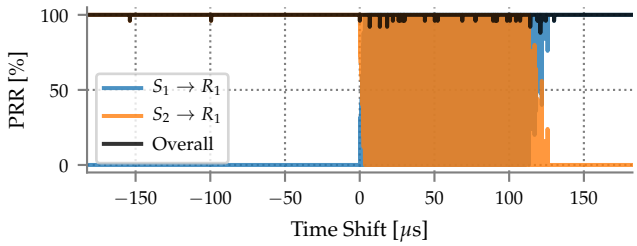


Figure 2.13: *PRR* in a single-receiver scenario (channel 4, *PRF*64, code 17).

*PRF*16 yields a meter-level error standard deviation, due to the magnitude of outliers whose position is nonetheless well-delimited (Figure 2.11b). The time shifts associated to outliers are those for which the SFDs of the link at stake overlap with the preamble of another frame, including both POLL-POLL and POLL-RESPONSE conflicts.

2.4.4 Concurrent Transmissions with the Same RF Configuration

Communication reliability. In this case, each receiver might get either the frame sent by the intended sender, the competing one addressed to the other receiver, or none if collision occurs.

Figure 2.12 shows results for both weak and strong links, with the overall average

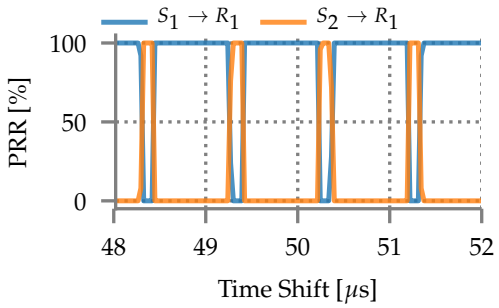


Figure 2.14: Zoom-in of Figure 2.13.

PRR remaining at 45% and 54.6% respectively. The charts are symmetric w.r.t. the zero-shift axis, reflecting the fact that the links are equivalent. When S_2 transmits earlier (left side of the charts), its packet is likely to be received by the intended target. If its signal is stronger than the interfering one (Figure 2.12b), its packet is received with nearly 100% probability. Instead, when the delayed interfering signal is stronger (Figure 2.12a), the *PRR* for S_2 covers the whole 0–100% range, achieving $\sim$ 90% on average.

Same receiver. When studying the areas of the charts where the *PRR* fluctuates, we noticed that they complement each other, i.e., when a receiver misses a frame from its intended sender, it likely receives the interfering frame instead. To see it clearly, we visualize the data differently (Figure 2.13) by focusing on a single receiver R_1 and plotting, for every time shift, the amount of packets R_1 receives from either S_1 or S_2, and the total. First, we note that the overall *PRR* remains $\sim$ 100% throughout the tested range, witnessing a very low rate of collisions in a situation with two transmitters competing on the medium to reach the same receiver. Further, we confirm that if the early signal is stronger ($S_2 \rightarrow R_1$) this is the one received (left side). Instead, when it is weaker ($S_1 \rightarrow R_1$) the radio often "switches" to the stronger one (right side) when it comes during the preamble of the weaker one.

As visible in Figure 2.14, this switching occurs roughly every 1 µs and lasts for $\sim$136 ns. Its periodicity matches the duration of a single preamble symbol, suggesting that the radio is able to ignore the stronger frame and keep receiving the weaker one if the symbols of the two preambles are displaced enough. Otherwise, if they coincide, the radio switches to the stronger frame. Interestingly, there are no fluctuations when the absolute shift exceeds 140 µs, i.e., when the later (even if stronger) preamble arrives after the SFD of the first frame was received.

The role of SFD timeouts. We ran these experiments by configuring the radio SFD timeout to be larger than the duration of two preambles. A lower value causes significant packet loss as the radio often *i)* synchronizes with the weaker preamble and starts accumulating preamble symbols *ii)* switches to the stronger delayed preamble *iii)* misses the weak SFD because it is "overridden" by the stronger preamble, and *iv)* eventually times out before having a chance to receive the stronger SFD. Therefore, the SFD timeout is crucial for concurrent transmissions when all nodes share the same RF configuration. However, we verified that it bears no influence in the previous cases with different *PRF* and/or preamble code.

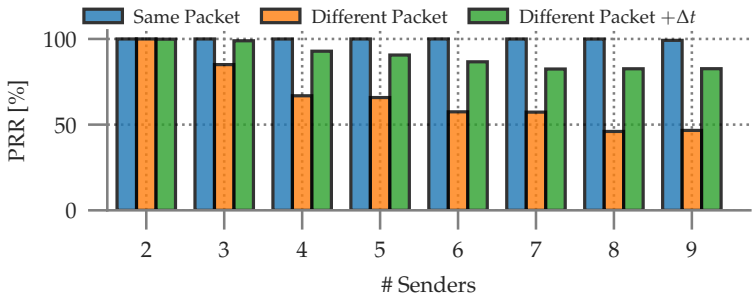


Figure 2.15: *PRR* vs. number of concurrent senders (ch.4, *PRF*64, code 17).

Multiple transmitters. Many modern protocols for low-power wireless networks build on tightly synchronized network floods. In these protocols, multiple nodes may transmit the same packet (or even different ones) concurrently with the receivers being able to reliably decode one of them. Therefore, we study the performance of UWB receivers when multiple transmitters broadcast concurrently, and extend our experimental setup with more nodes. One of them was configured as the unique receiver, while up to 9 others served as concurrent senders; we ensured they remained tightly synchronized. The *PRR* is computed based on successful reception of *any* of the concurrent packets.

First, we looked at the worst case where all senders are arranged in a circle, 2 m away from the receiver. In this scenario, almost no communication occurred when senders used *different* packets. This was expected because all arriving signals had similar strengths and timing, and therefore the receiver was unable to discern them. However, when all senders used *identical* packets, this same-distance network yielded highly variable results. The choice of senders (among the 9 available) and their relative distance affected the results tremendously, with the *PRR* varying in 0–100% even with only 2 senders. This indicates that slight variations in signal propagation paths, indoor reflections, and manufacturing differences among the nodes (e.g., clock drift, radiated power) may cause destructive interference.

In practical network deployments, however, the distances among nodes are never exactly the same. Thus, we repeated the experiments by placing the nodes along a corridor of our building. We did *not* compensate the signal propagation delay as in the aforementioned protocols all neighbors are potential receivers, with different distances.

Figure 2.15 shows the *PRR* as we add more concurrent transmitters in sequence, starting from the two closest to the receiver to the farthest one. When the whole network transmits the same packet, it is received in >99% of the cases. When different packets are transmitted synchronously, instead, *PRR* decreases as the senders increase, down to $< 50\%$ for 9 senders. Interestingly, the results are better ($PRR > 85\%$) when the transmissions are intentionally scattered by adding a random jitter within 20 µs; even if packets are different, the time gap between them enables the receiver to synchronize with the first one and stick to it till the end of the reception. *PRF*16 shows the same relative trends but with worse absolute values, e.g., $PRR < 35\%$ for 9 senders synchronously transmitting different packets and 81% with the jitter added.

Ranging. Although one of the concurrently transmitted packets is likely to be received, the inherent non-determinism severely undermines ranging as nodes may receive, e.g., the RESPONSE from the wrong responder, depending on the conditions discussed in this section. Not only the receiver may fail to detect the expected packet, it may even decode it correctly but associate it to a wrong timestamp. Since concurrent transmissions share the same preamble code, a peak associated to the interfering transmission will always appear in the CIR, but in another position that depends on the relative time alignment of preamble symbols. The timestamp is extracted from the CIR based on the first peak with an amplitude above a given threshold, that is not necessarily the peak from the decoded packet (Figure 2.16).

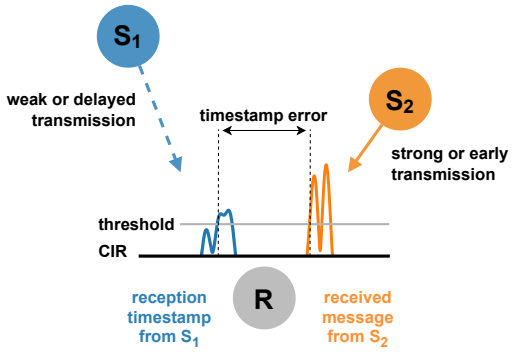


Figure 2.16: Wrong timestamp associated to the received packet due to an interfering transmitter affecting the CIR.

This observation is reported in [75], which explores the feasibility of a concurrent ranging scheme where the individual "peaks" from simultaneous responders are recovered from a single CIR at the initiator. But, unless ranging is performed with dedicated techniques, the potential mismatch between the timestamp and the received packet makes ranging in the same RF configuration impractical.

All systems in which the integrity of the timestamp needs to be ensured must take these issues into account. They will become relevant again for the localization systems presented in the second part of the dissertation (see §5.2 and §7.6.2).

2.4.5 Combined Settings

We showed (§2.4.2) that links with different *PRFs* barely affect each other. We also showed (§2.4.4) that one of the packets from multiple senders with the same RF settings can be received with high probability. The question is whether the two properties can be exploited at the same time; this would enable, e.g., to run non-interfering instances of the same flood-based protocol over different complex channels.

To this end, we doubled the 10-node deployment in §2.4.4 by pairing each of its nodes with another one configured to use the same frequency channel (4) but a different *PRF*. We also synchronized the two networks ensuring that all nodes transmit the same packets simultaneously.

Table 2.2 shows the *PRR* for the two receivers in each network vs. varying number of senders, compared to the baseline obtained with the two networks isolated. Although the reliability of the complex channel with *PRF*16 is clearly affected by the increase in senders, it nonetheless remains >84%, making it a useful design choice. On the other hand, this setup confirms the reliability of *PRF*64, yielding a $PRR > 99\%$ even under the heaviest load.

Table 2.2: *PRR* for 2 co-located networks with different*PRFs*.

#senders	**PRF16**	**PRF64**
One network active (baseline)		
9	–	99.98
9	98.35	–
Both networks active		
2+2	96.08	99.98
3+3	95.41	99.92
5+5	84.65	99.96
9+9	86.30	99.20

2.5 Discussion

Table 2.3 summarizes our findings. In contrast to DecaWave's claims (p. 15, [68]), we found that there is some interference between different *PRFs*. *PRF*16 is more affected, especially when the interferer is close to the receiver. In practice, complex channels with different *PRFs* are almost independent for both communication and ranging, and can be used to deploy co-located yet separate networks (e.g., to increase the scalability of localization systems), or to enhance parallelism in multi-channel protocols like TSCH.

Different preamble codes with the same *PRF*, unfortunately, do not provide independent channels due to cross-code interference. The rate of collisions can be significantly reduced by synchronizing well the senders, and communication was highly reliable when the interferer was farther than the sender. Arguably, these properties are likely to find application only in very specific, niche application cases.

Finally, our tests with concurrent transmissions in the same complex channel yielded very positive outcome for protocols relying on synchronous transmissions or medium access based on contention. Indeed, the very successful reception (99%) of simultaneous transmissions of the *same* packet from multiple nodes in principle enables techniques like those in A-MAC [57], Glossy [56] and others [59] on UWB. Further, the fact that *different* packets rarely collide destructively if shifted by a tiny jitter enables the techniques in Crystal [60] and Chaos [61].

Table 2.3: Summary of findings.

	Communication	Ranging
Different PRFs	Concurrency is possible with high reliability for both *PRFs*. *PRF*16 is slightly affected by the interference, showing minor packet loss.	Ranging is reliable and measurements are precise. In channel 2, using *PRF*16 results in some (rare) outliers.
Same PRF, different codes	Preamble codes do not provide independent channels. Only the first frame sent is likely to be received. Reliability increases if frames are synchronized, especially when the interference source is far from the destination.	Ranging exchanges may fail or give imprecise estimates depending on how POLL and RESPONSE overlap. *PRF*16 is susceptible to extreme outliers.
Same PRF and code	One of the concurrent frames is likely to be received regardless of the way they overlap. If payloads are identical, sending them synchronously ensures near-perfect reliability. Instead, different packets must not be precisely synchronized to avoid collisions. SFD timeout should be increased.	Similarly to the case with different preamble codes, ranging exchanges may fail depending on the time shift between initiators. Both *PRF*64 and *PRF*16 are affected by extreme outliers.

2.6 Related work

Early work on UWB investigated, mostly theoretically, methods enabling multiple access to the wireless medium, e.g., different time-hopping codes [64, 65] or orthogonal pulse shapes [76, 77]. The IEEE 802.15.4 standard [44] is based on the former although, as shown in §2.4, it results in unreliable performance unless codes use different *PRFs*, as observed by DecaWave [68]. Our work goes beyond this observation, and offers quantitative evidence from real-world experiments about the expected performance using different *PRFs* and codes, informing the design of communication and ranging schemes, to seize the opportunities offered by concurrent transmissions.

Few works investigated these opportunities. SurePoint [78] employs a Glossy-like flooding primitive to schedule ranging exchanges between mobile tags and anchors; however, it is neither detailed nor evaluated as it is not the focus of the paper. Concurrent ranging [75] exploits tightly-synchronized transmissions to measure the distance to several devices on a single TWR exchange by analyzing the CIR signal information, offering experimental results geared towards the specific technique proposed. Neither work analyzes the performance of concurrent transmissions w.r.t. different *PRFs* and preamble codes, nor evaluates the impact of time (de)synchronization or different signal power—key aspects of general applicability, and addressed in this chapter.

Our experimental analysis is inspired by the more established work on concurrent transmissions in low-power narrowband radios with several protocols providing efficient and reliable multi-hop communication. These protocols exploit, in the same RF channel, the PHY-level properties that allow the radio successfully decode data when multiple senders transmit identical or even different packets simultaneously. Our results in §2.4.4 suggest that similar benefits can be seized in UWB, potentially inspiring a new wave of research on UWB communication protocols based on concurrent transmissions.

2.7 Conclusions

Concurrent transmissions are a powerful tool for the designers of communication protocols, and have been successfully exploited in different ways by many works in IEEE 802.15.4 narrowband radios. Unfortunately, the same does not hold for UWB radios, whose peculiar ability to combine high-rate communication and accurate distance estimation is placing them at the forefront of IoT scenarios. Indeed, the guidelines in the IEEE 802.15.4 standard conflict with the recommendations for the most popular UWB chip, the DW1000.

We analyzed the conditions under which concurrent transmissions can be reliably exploited under different radio settings, to a depth and extent hitherto unreported in the literature. The high-level findings we distilled can be immediately exploited by protocol designer, potentially inspiring a new generation of UWB networking and ranging protocols exploiting the advantages of concurrent transmissions.

3

Concurrent Transmissions for Multi-hop Communication on Ultra-wideband Radios

Ultra-wideband (UWB) radios are rapidly becoming a prominent player in the ever-changing landscape of Internet of Things (IoT). They jointly provide accurate distance estimation (ranging) and high-rate wireless communication, therefore reuniting in a single radio transceiver two key functions of many IoT scenarios.

Nevertheless, a staple network stack for UWB is still missing. This is partly explained by the fact that the interest in UWB, at its peak about a decade ago and largely forgotten thereafter, renewed only recently, fueled by new chips (e.g., the popular DecaWave DW1000 [63] described in Chapter 2) that yield high ranging accuracy and yet are small, cheap, energy-savvy, and standard-compliant. In contrast, during the same decade, research in academia and industry generated numerous protocols, systems, and real-world deployments targeting a variety of traffic patterns, operating conditions, and stack layers. Among these, the approaches based on *concurrent transmissions* on the same radio channel, as popularized by Glossy [56], have proven a very effective building block for protocol design. Several protocols (e.g., [56, 59, 60, 79, 61, 80]) embraced this technique and its variants, ultimately pushing the envelope of IEEE 802.15.4 radios by achieving low latency, high reliability, low energy consumption—all at once.

Besides published works, the fact that almost all of the teams (and *all* of the top ones) in the four editions of the EWSN Dependability Competition relied on Glossy-

This chapter revises our publication [2]: D. Lobba, M. Trobinger, D. Vecchia, T. Istomin, and G. P. Picco. "Concurrent Transmissions for Multi-hop Communication on Ultra-wideband Radios". In *Proceedings of the* 17^{th} *International Conference on Embedded Wireless Systems and Networks (EWSN)*. 2020.

like systems is another witness that concurrent transmissions are the state-of-the-art technique in IEEE 802.15.4 narrowband radios. Therefore, it is natural to investigate whether they are applicable also to UWB radios. We provide a concise primer of narrowband concurrent transmissions in §3.1.

Goals. In Chapter 2 we elicited the conditions for successful UWB concurrent transmissions, both on the same channel (as in this chapter) and on different channels (not of interest here). Moreover, in the UWB localization system in [78], the use of Glossy-like concurrent transmissions is reported as a means to coordinate ranging exchanges.

The study presented in this chapter exploits some of the findings in these works, towards its broader objective to:

1. determine whether *different flavors* of concurrent transmissions can be embodied in a *full-fledged protocol and system* and, in the process,
2. highlight *similarities and differences* w.r.t. their narrowband counterpart in terms of both *implementation complexity and system performance*, and ultimately
3. provide a *reference implementation* of concurrent transmissions protocols that can be *directly* used and improved by the research community at large, fostering the adoption of this technique on UWB radios.

Which type of concurrent transmissions? As mentioned, several "flavors" of concurrent transmissions exist. Glossy was originally designed to support a single network-wide flood, triggered at an *initiator* node; all nodes disseminate the *same* packet via a tightly synchronized schedule of alternating RX and TX slots, until the desired number N of packet (re)transmissions are performed.

In this chapter we consider two other variants, representative of the state of the art. On one hand, in the last editions of the above EWSN competition several systems [81, 82] achieved very high performance by changing Glossy to exploit only the single initial RX slot necessary to receive the packet, followed by N consecutive TX slots catering for its re-transmission. Given that the RX energy costs of the popular UWB platform we use are almost twice than TX ones, this TX-centric operation is definitely worth investigating. On the other hand, several works [60, 61] observed that floods are quite reliable even when *different* packets are concurrently transmitted by different initiators in the original Glossy scheme, offering another dimension to our study.

Methodology and contribution. Our investigation is system-driven, and relies on *complete protocol implementations* of the variants above as well as *testbed experiments* on multi-hop topologies. We use the popular DW1000, and specifically the EVB1000 boards, as our target UWB platform, and develop software atop Contiki, exploiting the availability of drivers for the DW1000 [83]. This methodology is in contrast with that of Chapter 2, whose results rely on micro-benchmarks with few nodes in the same neighborhood, and also with [78], whose very short description of their Glossy-like component is insufficient to ascertain its actual performance or guide further developments.

This system-driven emphasis enables us to directly face the opportunities and challenges in exploiting UWB concurrent transmissions as well as to highlight key differences w.r.t. the corresponding narrowband implementations. One prominent example is the ability of the DW1000 radio to precisely schedule transmissions, which greatly simplifies implementation. Further, it also allows us to confirm, and sometimes disprove, some of the findings in [1, 78], ultimately contributing to a better system-level understanding of UWB concurrent transmissions.

We re-implemented Glossy and its TX-based variant from scratch, motivated by key differences in the radio operation and configuration. However, we also used the publicly-available codebase of Crystal [60, 84], a recent protocol that exploits both classic, single-initiator, same-packet floods as well as multiple-initiators, different-packet ones. As these are combined in a single protocol, Crystal serves as a sort of "catch-all" protocol enabling us to experiment with different types of concurrent transmissions in a single system. Further, the fact that we *reuse* the original Crystal codebase allows us to ascertain the extent to which this higher-level protocol built atop a narrowband Glossy layer can work when the latter is replaced with our UWB-based one. Our analysis shows that only minimal changes are required, suggesting that existing Contiki implementations of other higher-level protocols [59, 61, 85, 80] may be similarly reused for UWB radios, with minimal changes.

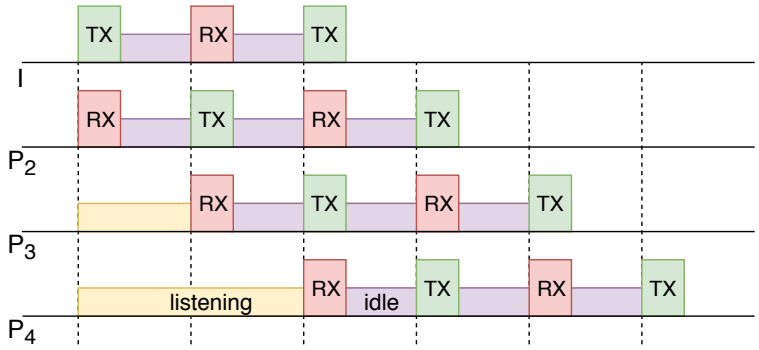
We illustrate the salient details of our implementations of the two Glossy variants (§3.3) and of Crystal (§3.4) hand-in-hand with their evaluation in a 23-node indoor testbed at our premises, which enables us to experiment at scale on multi-hop topologies. Results show that UWB concurrent transmissions yield benefits similar to narrowband, achieving near-perfect reliability, and very low latency and energy consumption across the 4 hops in our testbed. Further, due to the high-accuracy clock and the high data rate, they also enable order-of-magnitude improvements in network-wide time synchronization. Nevertheless, as our experimental results are inevitably biased by the peculiarities of our testbed, we also manipulate artificially our setup to investigate the conditions under which UWB concurrent transmissions may fail, validating or disproving earlier findings [1, 78].

The chapter ends by distilling findings and lessons learned (§3.5) that will hopefully inspire further work on the topic, before ending with brief concluding remarks (§3.6). We argue that our results pave the way for the exploitation of concurrent transmissions in UWB, which we foster by releasing our systems as open source [73], enabling their immediate use and improvement by researchers and practitioners.

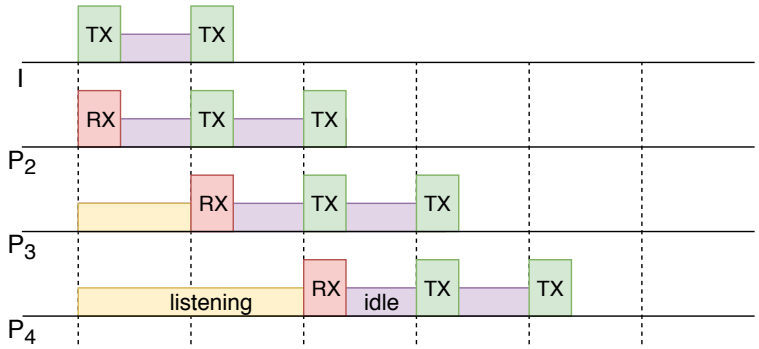
3.1 Concurrent Transmissions in Narrowband Radios

We provide a concise primer on the known enabling factors for concurrent transmissions and how they can be exploited, looking back at narrowband research.

Protocols based on concurrent transmissions rely on phenomena characteristic of IEEE 802.15.4 narrowband [86] when multiple senders simultaneously transmit towards the same receiver(s), on the same RF channel. The first phenomenon, *constructive interference*, occurs when the *same* packet, transmitted by different senders,



(a) GLOSSY.



(b) GLOSSYTX.

Figure 3.1: The two Glossy variants in a 4-hop network.

arrives at the receiver with a maximum time displacement of 0.5 µs, the duration of a bit (chip) in the transmitted chip sequence obtained by the direct-sequence spread spectrum (DSSS) encoding of the original message. These condition cause the union of concurrent identical signals, which actually improves the reliability of packet RX. The *capture effect*, instead, occurs even for *different* packets, as long as they arrive with a relative shift of no more than 160 µs, i.e., the duration of the synchronization header. The shift does not necessarily cause a collision, even more so since the radio is likely to switch RX from a weaker signal to the stronger one. In this case, one of the packets is received, with probability depending on the density of neighbors and the difference between the strength of the received signal and the sum of all other signals, including noise. This required difference is termed *capture threshold* and has been found to be around 3 dB for IEEE 802.15.4 narrowband in the 2.4 GHz band. The reported timing and signal strength requirements are different for other radios, such as IEEE 802.15.4 sub-GHz narrowband or Bluetooth [87]. Finally, the reliability of narrowband concurrent transmissions has been linked to the *carrier frequency offset* (CFO) [88] between overlapping signals, which can cause destructive interference.

3.1.1 Network Flooding with Glossy

Glossy. Originally designed for multi-hop time synchronization, the Glossy [56] protocol exploits the two phenomena above to achieve fast, energy-efficient, and reliable network floods. Figure 3.1a illustrates the concept. The *initiator* begins a flood

by broadcasting a packet. As the rest of the network is assumed to be already listening on the channel, the packet is received and immediately rebroadcast by neighbors, yielding concurrent transmissions. After (re)transmitting, the nodes go back to receiving, thus repeating the RX/TX sequence up to N times; the value of N is key to determine the balance between reliability and energy consumption. Another important factor affecting energy consumption is the duration of the slots, which must be long enough to accommodate either a packet TX or RX, including some guard times and software delays; nevertheless, when a packet is *not* received, a node listens for the entire slot, potentially wasting energy.

GlossyTX. A Glossy[1] flood unfolds by alternating RX and TX slots (Figure 3.1a); actually, a node is allowed to TX a packet *only* after a successful RX. This choice was originally motivated [56] by the use of CC2420 radio events as a means to enforce tight synchronization. However, it has drawbacks; a node that receives a packet in a RX slot and loses it in the next one is forbidden from rebroadcasting the (same!) packet in the subsequent TX slot, wasting time and energy, and possibly decreasing reliability. Glossy partially mitigates these problems by allowing only the initiator—i.e., the synchronization source—to transmit in a TX slot regardless of the outcome of RX ones, improving the flood progress in some unlucky situations.

Figure 3.1b shows an alternative scheme in which each node, after the initial successful RX, performs its N retransmissions in consecutive TX slots. This approach, hereafter called GlossyTX to distinguish it from the original, was first introduced by the winners of the 2^{nd} EWSN Dependability competition [81], and exploited by other teams in following editions. A major drawback of GlossyTX is that its implementation, relying solely on timeouts, makes it more challenging to ensure tight synchronization of concurrent senders on TelosB-like devices. Further, more nodes transmit concurrently, possibly increasing the probability of collisions [86, 61]. On the other hand, GlossyTX unlocks several potential advantages by: 1. solving the problem above induced by the original Glossy scheme, therefore potentially improving latency and/or reliability 2. enabling significant energy savings, by shortening the radio-on time by removing the unnecessary RX slots or, dually, 3. enabling reliability improvements, by replacing them instead with up to $N - 1$ TX slots.

The fact that Glossy and GlossyTX strike different trade-offs would already be enough motivation to consider them both. However, an even more compelling reason is the fact that, in the popular DW1000 UWB radio we use, the RX current draw is almost twice than the TX one, making GlossyTX preferable, at least in principle.

3.1.2 Higher-level Abstractions: Crystal

The effectiveness of Glossy gave rise to protocols that are built directly atop the original implementation [59, 60]. Among these, Crystal is particularly suited for the study presented in this chapter because 1. it does not require modifications to Glossy, therefore allowing us to explore the extent to which the original narrowband can be

[1]Hereafter, we use "Glossy" to refer generically to the system in [56], and "Glossy" to refer to the specific scheme derived from it (Figure 3.1) and implemented here.

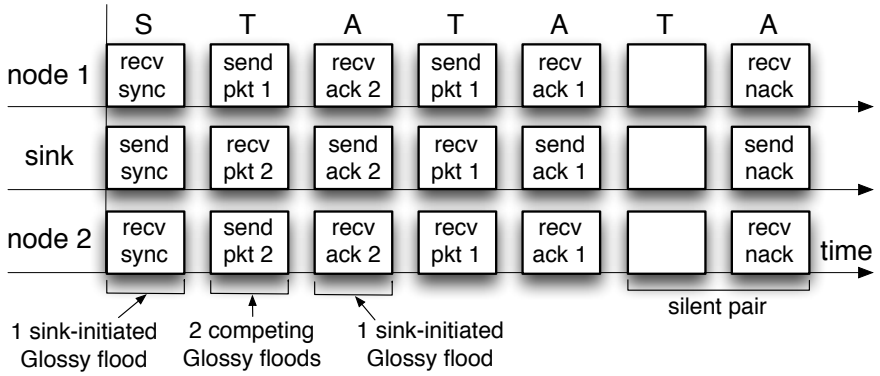


Figure 3.2: A Crystal epoch (from [84]).

replaced by our UWB implementations, and 2. it exploits concurrent Glossy floods containing *different* packets along with conventional, isolated ones, as described next.

Crystal [60] targets scenarios with aperiodic data collection and sparse traffic (e.g., those induced by data prediction, which provided the original motivation [60]) where relatively long periods of inactivity are interleaved with simultaneous data reporting from several nodes towards a sink. These scenarios require a careful balance between the need to minimize energy consumption during the inactive periods and to guarantee timely and reliable delivery of data whenever needed. Crystal achieves this balance by 1. dividing time into periods (*epochs*) that define the granularity of reporting, 2. exploiting the reliability of Glossy floods even when concurrently disseminating *different* packets, and 3. dynamically scheduling them as needed during the active time of the epoch, and putting the radio to sleep during the rest of it.

A Crystal schedule unfolds at the beginning of the epoch (Figure 3.2) and is composed of three phases, each corresponding to a Glossy flood:

- the initial S phase, starting from the data collection sink, ensuring time synchronization for Glossy slots and the next active period;
- the T phase, used by concurrent senders to disseminate their data. It is therefore the crucial phase, where *different packets* compete within concurrent Glossy floods originating from *different initiators*. Typically, due to the capture effect, one flood propagates successfully towards the sink overpowering the others;
- the A phase, directed from the sink to all the nodes. It is performed in isolation and provides a network-wide acknowledgment of sorts, enabling each sender to determine whether a retransmission—another Glossy flood in the next T phase—is needed or not for the packet they hold.

Termination occurs at each node when an empty T phase followed by an A phase containing no acknowledgment are observed for a number R of times.

Crystal's reliability and energy consumption can be tuned by means of the number of retransmissions inside the various Glossy floods, N. The total duration of Glossy phases, W, is then set depending on N and the maximum number of hops in the network.

3.2 Related Work

Concurrent transmissions, pioneered by Glossy, have been a breakthrough in narrowband low-power networking, showing unprecedented performance and leading to numerous follow-up works. It is therefore not surprising that researchers have begun investigating their applicability to other radio technologies, e.g., Bluetooth Low Energy (BLE) [89]. However, bringing techniques and results from narrowband to the impulse-radio UWB is non-trivial, due to the significantly different characteristics of the PHY layers.

In Chapter 2, based on single-hop micro-benchmarks, we experimentally verified that concurrent transmissions are possible on UWB links. This holds with *identical* frames, but it was observed that also *different* frames can be supported under certain conditions, namely de-synchronization and signal strength disparity. These findings lay the foundation for our study, where we exploit concurrent transmissions in actual full-fledged systems for multi-hop data dissemination and collection. Moreover, to further investigate the limitations of the schemes we employ, we analyze the synchronization requirements for correct reception on the time scale of a single BPM-BPSK data symbol, unlike in our previous study.

Another study [78] reported the use of a Glossy-like protocol to support an UWB localization system, i.e., the main contribution. The work described two necessary conditions for the correct operation of Glossy: preventing data symbol collisions and ensuring signal coherency of concurrent transmissions. As concurrent transmissions were not the main focus, however, the authors provide very few implementation details and no performance evaluation.

UWB concurrent transmissions have also been recently applied for concurrent ranging [75], in which all receivers of a single ranging request reply together. The authors show that the channel impulse response (CIR) available on the DW1000 can be exploited to collect multiple time-of-flight measurements at once. However, the system is not designed for communication, and the reliability of reception is only tested for the purpose of ranging in a single-hop scenario.

3.3 Glossy on UWB

We first illustrate our implementation of Glossy and GlossyTX on UWB (§3.3.1), focusing on how we exploit the opportunities offered by the DW1000 chip. We then quantitatively evaluate the performance of both variants (§3.3.2), drawing parallels with their narrowband counterparts. Finally, we analyze potential threats to the correct operation of our implementations and critically revisit some of the findings reported in the literature (§3.3.3).

3.3.1 Implementation Highlights

The original implementation of Glossy [56] targeted the TelosB motes (CC2420 radio and TI MSP430F1611 MCU) and was technically complex due to the lack of hardware support for precise timestamping of received packets and scheduling retrans-

missions. The clocks of the radio and the MCU run asynchronously, which causes a random jitter in the transfer of digital signals between these two components. Therefore, it is difficult to guarantee that the MCU issues the TX command to the radio at a designated time precisely enough, due to the non-deterministic time that elapses between a detected radio event and the invocation of the corresponding interrupt service routine (ISR). Moreover, the original Glossy avoided using the platform timers to schedule transmissions, because the stable 32kHz clock does not provide a sufficient resolution and the 4MHz DCO clock is not stable enough. Therefore, all actions of the protocol are triggered solely by radio events (e.g., *end-of-RX, end-of-TX, SFD*), further complicating the implementation.

The implementations described in this chapter are for the DecaWave EVB1000 board, equipped with the DW1000 UWB radio and STM32F105 ARM Cortex M3 MCU. Other MCUs can be easily supported, however their clock speed and the data rate of SPI bus connecting MCU and radio can affect the timing of Glossy floods.

The DW1000 simplifies the Glossy implementation on many accounts. First and foremost, the DW1000 gives access to its internal clock; this can be used to 1. timestamp received frames with sub-ns precision, and 2. schedule delayed frame TX with an 8 ns granularity. Both opportunities simplify the implementation tremendously. Random delays in ISR execution are no longer a problem, as the radio can be instructed to begin TX at an *exact* time in the future. Further, there is no jitter or non-determinism, because a single component—the radio—both timestamps the RX and triggers the TX using the *same* built-in clock.

Two variants of Glossy. As mentioned in §3.1, our Glossy and GlossyTX implementations have different purposes. Glossy is a faithful re-implementation of the original system in [56], where we exploit the precise timestamping and TX scheduling of the DW1000. Notably, we retain the original scheme in which a TX can be performed only for the packet received in the immediately preceding RX slot, except at the initiator (§3.1). This constraint was motivated in [56] by the need to obtain accurate timing information, and is made superfluous by the DW1000 features. Nevertheless, we preserve it to avoid changing the protocol too much, with the intent to have a yardstick enabling *direct* comparison with the body of literature on narrowband Glossy.

Indeed, if one were to allow a TX of a received packet regardless of the outcome of the preceding RX slot, the purpose of the latter would become unclear. A more efficient protocol would be one where, after the first successful RX, the packet is transmitted N times without other RX slots. This is exactly what the GlossyTX variant does, for which our implementation takes full advantage of the DW1000 features. Direct access to the stable clock of the radio greatly simplifies implementation. The latter was actually the major hurdle pointed out by the literature [81, 82], which however lacks in-depth evaluations comparing GlossyTX vs. Glossy.

Anatomy and duration of a slot. A Glossy slot must account for the time necessary to: 1. read/write the frame payload from/to the radio via SPI, 2. transmit/receive the

Table 3.1: Operation durations for UWB packets (µs).

Frame (bytes)	SHR	PHR & payload	SPI read & write	Other
15	73	45	∼36	∼250
127	73	178	∼304	∼250

Table 3.2: Slot durations for UWB and narrowband (NB) in µs.

Frame (bytes)	UWB	NB
15	404	887
127	806	4471

frame synchronization header, 3. transmit/receive the physical layer header and the payload, 4. perform various software and hardware operations required for packet processing and radio configuration. Table 3.1 shows approximate durations of these steps in the EVB1000, for the two packet lengths we experiment with. By summing up the duration of all the steps, we obtain the actual slot sizes. Their comparison with corresponding slot sizes of narrowband Glossy (Table 3.2) shows a key advantage of UWB: the higher data rate (6.8Mbps vs. 250kbps on the CC2420) allows for slots that are 2.1x and 5.5x smaller, with evident benefits in latency.

On the other hand, concerning the first step above, the DW1000 does not support writing/reading the frame payload during its TX/RX, a feature of the CC2420 which increases parallelism. The DW1000 does allow uploading the payload in parallel with transmitting the preamble; however, we could not exploit this feature because, for the preamble setting we used, the former is slower than the latter.

Dynamic clock frequency calibration. The radio clock of our platform is very stable. Even though DW1000 tolerates up to ±20 ppm [70] frequency drift (§2.1), the EVB1000 platform we use integrates a ±10 ppm oscillator, individually calibrated (trimmed) by the manufacturer to achieve ±3 ppm in normal conditions. Nevertheless, temperature and voltage variations may cause its frequency to drift within the full ±10 ppm range.

The authors of [78] report that this drift may undermine the reliability of concurrent transmissions. Therefore, we implement a dynamic frequency calibration of the radio clock of the receivers, relative to that of the flood initiator. Inspired by [78], the calibration is achieved by observing the time offset between the expected and actual arrival of consecutive floods, and adjusting the radio oscillator frequency of every receiver appropriately. This is done by trimming the oscillator with a hardware-defined step of 1.45 ppm. By choosing the value closest to the desired frequency, we ensure that the frequency offset of any device w.r.t. the flood initiator is within ±0.725 ppm. For any pair of non-initiator devices, their relative frequency offset stays within 1.45 ppm.

This dynamic calibration requires the radio clock to remain active in between floods,

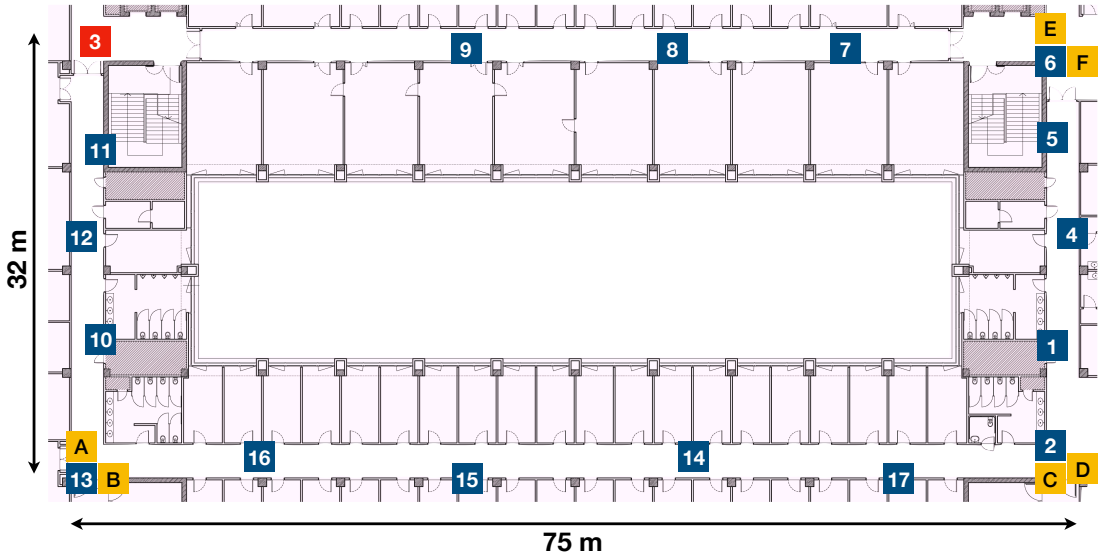


Figure 3.3: Experimental testbed. Out of the 23 nodes available, 22 were running the protocols under study; node 3 served as a sniffer.

with the radio in idle mode. This has relatively high power consumption (Table 3.5); however, the calibration is in general infrequent. In §3.3.3 we further elaborate on the impact of this technique via experiments that provide additional insights beyond what reported in [78], whose results are based on a custom hardware design achieving higher synchronization accuracy.

3.3.2 Evaluation

We evaluate several aspects of Glossy and GlossyTX, highlighting similarities and differences between them and w.r.t. their narrowband counterparts.

Experimental setup and radio configuration. We tested our implementation of Glossy in a 23-node testbed deployed in the corridors of an office building (Figure 3.3). The communication range normally extends through the entire length of each straight segment, with at least one link with $PRR \geq 90\%$ for every pair of adjacent corners. However, exceptions exist where shorter links are less reliable, e.g., node 17 cannot communicate directly with 13. Further, we verified that node 11 cannot communicate directly with 9; therefore, we do not use the node 3 in between them, and set node 9 to be the initiator, achieving a network diameter of 4 hops.

As for the radio configuration, we use channel 4, 6.8 Mbps data rate, 64 MHz *PRF*, 64 µs preamble, and the transmission power of `0x9A9A9A9A` recommended [63] for the combination of channel and *PRF* we use, corresponding to −41.3 dBm/MHz or −11.75 dBm.

Flood reliability. One of the main benefits of protocols based on concurrent transmissions is their ability to achieve near-perfect reliability. The latter strongly depends on the number N of retransmissions (§3.1). However, long packets are also known

to be detrimental to reliability in narrowband [56]. For these reasons, we experiment with $N \in \{1, 2, 4, 8\}$ and 1. short packets of 15 B, allowing for 8 B of payload as commonly used in the literature, and 2. long packets of 127 B, the maximum allowed by the standard. For every combination of these values, we report results aggregated from 12000 floods.

Table 3.3 shows that, in our experiments, both variants always achieved perfect reliability with short packets, even with N=1. Instead, with long packets this is achieved only by Glossy and only with N=8; further, GlossyTX is systematically less reliable for $N \geq 2$, although it achieves a reliability $\geq$99% in all cases. Interestingly, the reliability of Glossy increases with N, as expected, while this is not always true for GlossyTX. For $N \geq 2$, in GlossyTX the number of TX slots at each node is higher than RX slots, increasing the number of nodes transmitting simultaneously and therefore the chance of occasional collisions among the (long) packets.

Table 3.3: Glossy vs. GlossyTX: reliability.

Protocol	Frame (bytes)	Average flood reliability, %				Minimum node reliability, %			
		N=1	N=2	N=4	N=8	N=1	N=2	N=4	N=8
Glossy	15	100	100	100	100	100	100	100	100
	127	99.91	99.997	99.9992	100	99.5	99.95	99.991	100
GlossyTX	15	100	100	100	100	100	100	100	100
	127	99.91	99.97	99.95	99.997	99.5	99.8	99.0	99.95

Latency. These trends are mirrored by the *first relay count*, i.e., the number of slots elapsed at a node before the first successful RX slot, effectively an indirect measure of latency. The average number of slots for each configuration is shown in Table 3.4. The values for this metric are identical for the two variants in the case of short packets or N=1, but are slightly higher in the other cases, meaning that the flood is slightly delayed due to lost packets and consequent retransmissions.

Table 3.4: Glossy vs. GlossyTX: latency.

Protocol	Frame (bytes)	Mean first relay counter			
		N=1	N=2	N=4	N=8
Glossy	15	1.32	1.32	1.32	1.32
	127	1.49	1.37	1.46	1.49
GlossyTX	15	1.32	1.32	1.32	1.32
	127	1.49	1.48	1.56	1.46

To investigate the *maximum* latency, Figure 3.4 focuses on node 11, the farthest from the initiator. In most cases, the flood reaches this node exactly after 4 hops, with

sporadic outliers in case of short packets. With long packets, the maximum latency is still very stable, though bigger, due to larger Glossy slots needed. However, the 99^{th} percentile shows increase in latency corresponding to 1–2 slots. Overall, there is a weak tendency for latency to grow when N increases, because of a higher chance of collision, as discussed before.

Compared to narrowband [56], our UWB implementation provides smaller latency due to the shorter slots used, with a 52% reduction for short packets and 82% for long packets.

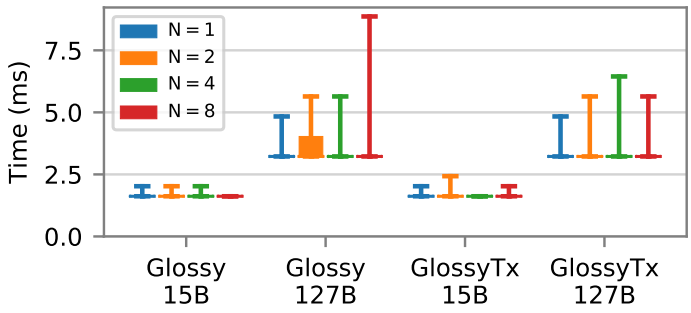


Figure 3.4: Latency of node 11, the farthest from the initiator (4 hops). Bars denote minimum/maximum values; boxes denote the 25–75% percentile.

Energy consumption. Unlike narrowband radios like the CC2420, for which TX and RX have similar energy costs, the RX current draw of the DW1000 chip is almost twice than the TX one (Table 3.5). This motivates investigating the energy consumption of the two Glossy variants, as they exploit very differently these two radio states. However, this energy unbalance prevents us from using radio-on time as an energy metric, as commonly done by the narrowband literature. We therefore resort to *modeling* directly the energy costs, as the structure of Glossy protocols is simple and largely deterministic. Specifically, we study the energy cost of Glossy and GlossyTX as a function of the hop distance from the initiator; however, we neglect the contribution of collisions, as these are generally rare and in any case dependent on the specific target environment and network topology.

The drain of electric charge of a node during a flood is:

$$Q = T_{TX}I_{TX} + T_{RX}I_{RX} + T_{listen}I_{listen} + T_{idle}I_{idle} \tag{3.1}$$

where T_i and I_i are, respectively, the time spent and corresponding current draw in a given state i (Figure 3.1). From this, energy can be computed easily as $E = Q \cdot V$, with knowledge of the voltage supply (Table 3.5).

The current draw for each state is shown in Table 3.5 for both the DW1000 and the CC2420 used in the original implementation of Glossy. In the latter case, the values are retrieved from the datasheet [62]. For the DW1000, the datasheet does not contain information specific to the radio configuration of our experiments; therefore,

Table 3.5: Nominal current draw and voltage supply.

		Current draw (mA)				
	Frame (bytes)	I_{RX}	I_{listen}	I_{TX}	I_{idle}	**Voltage (V)**
CC2420	any	18.8	18.8	17.4	0.426	3.0
DW1000	15	114.9	113.0	71.5	18.0	3.3
	127	116.5	113.0	61.1	18.0	

we use the current draw reported for the most similar one, i.e., the one with the same parameters but channel 2, which has the same central frequency of channel 4 but smaller bandwidth. Note that radio states with a lower power than the idle one cannot be exploited, due to the large time required by the radio to exit from them (e.g., up to ∼3 ms for the DW1000).

Interestingly, (3.1) also models the consumption of *narrowband* Glossy and GlossyTX, albeit with a few caveats. Indeed, the original implementation for CC2420 reads and writes frame data via SPI directly during RX and TX, respectively, avoiding inter-slot processing delays and the need for putting the radio to the idle state between RX/TX slots. Therefore, to apply our energy model (3.1) to narrowband, we 1. consider as T_{idle} the time (192 µs) needed by the radio to switch from RX to TX and the software delay (23.3 µs) required by the MCU to trigger a TX, and 2. account for it as if the radio were in RX. This, along with the fact that $I_{listen} = I_{RX}$ for narrowband (Table 3.5) leads to the simpler expression for *narrowband* variants:

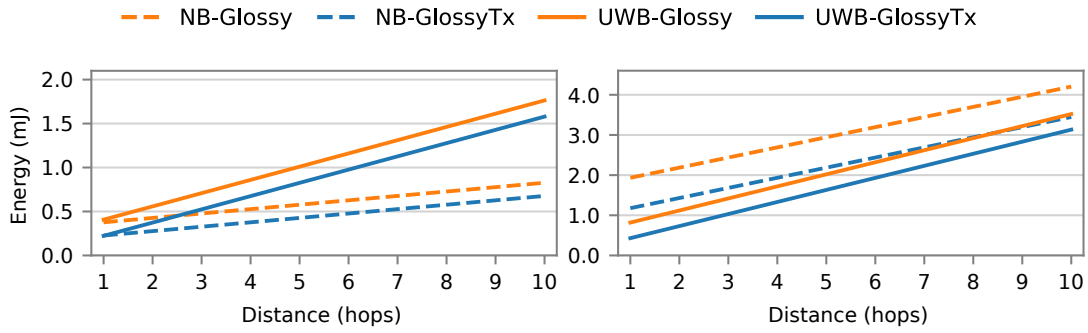
$$Q = T_{TX} I_{TX} + (T_{RX} + T_{listen} + T_{idle}) I_{RX} \tag{3.2}$$

The values of T_i can instead be determined as a function of N, of the slot duration (T_{slot}), of the radio time to TX or RX a frame (T_{frame}), and of the first relay counter (C):

$$\begin{aligned} T_{TX} &= N \cdot T_{frame} \\ T_{RX} &= N_{RX} \cdot T_{frame} \\ T_{listen} &= C \cdot T_{slot} \\ T_{idle} &= (N_{RX} + N - 1) \cdot (T_{slot} - T_{frame}) \end{aligned} \tag{3.3}$$

where $N_{RX} = N$ for Glossy, and $N_{RX} = 1$ for GlossyTX.

Figure 3.5 shows the resulting energy estimates for N=4; different N values exhibit similar trends. As expected, GlossyTX is more energy-efficient than Glossy both in narrowband and UWB. By scheduling only TX slots after the first successful RX, GlossyTX reduces the flood duration, sparing energy. This difference increases with N. As for the tradeoffs between narrowband and UWB, with short packets the former clearly outperform the latter. Interestingly, roles are reversed when long packets are transmitted. Despite the higher energy cost of both TX and RX for the DW1000 chip (Table 3.5), UWB GlossyTX is the most efficient solution. At the first hop it consumes nearly one third of its narrowband counterpart, and 4.5x less than narrowband Glossy. However, the gap decreases with hop distance.



(a) 15B frame. **(b)** 127B frame.

Figure 3.5: Energy consumption: Glossy vs. GlossyTX for narrowband (NB) and UWB. Note the difference in scale between the y-axes.

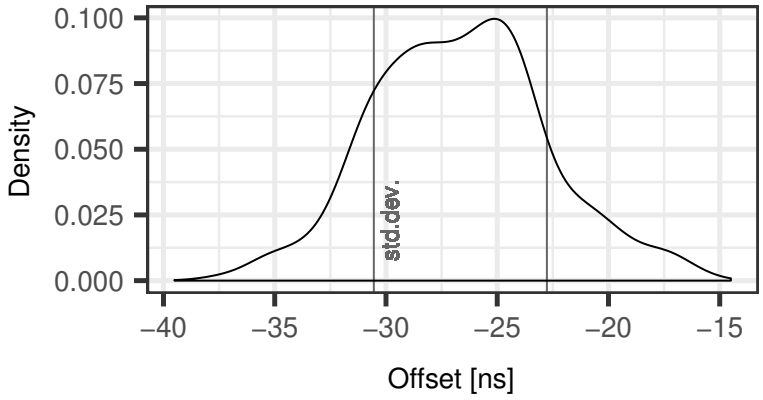


Figure 3.6: Time synchronization error.

The reason behind the higher energy efficiency of UWB-based solutions with larger payloads is twofold: 1. the data rate of CC2420 is 27x smaller than DW1000 (250 kbps vs 6.8 Mbps); to TX (or RX) 127B of data, narrowband radios stay active ∼18x longer than UWB, and 2. the processing delays required by the UWB implementation do not bear a significant negative impact on energy consumption as the radio remains idle; after the first successful RX, nodes spend in idle >60% of the time, saving considerable energy.

Accuracy of time synchronization. Finally, we recall that Glossy was originally proposed for time synchronization [56]. It is therefore interesting to investigate this aspect, especially given that the UWB platform we use provides access to its highly accurate clock. To study the synchronization accuracy, we 1. rely on the privileged position of node 3 and use it as a "sniffer", capable to hear and timestamp the RX of packets sent by node 9, the initiator, and node 11, the farthest from it, and 2. analyze their difference w.r.t. the first TX of the initiator (relay count of 0) and the first TX of node 11 (relay counter of 4).

As Glossy was, by design, unaware of the distance among nodes, the reference time at

the receivers is always biased by the signal propagation delay, ∼333 ns every 100 m. Knowing the overall distance the signal travels in our setup, we subtract that bias and determine the error distribution. This yields a setup similar to the original in [56], where nodes were all on the same desk and propagation time essentially negligible. Results show an underestimation of ∼6.5 ns per hop, resulting in an average offset of −26 ns at 4 hops (Figure 3.6). We attribute this bias to an imprecise antenna delay calibration. Overall, the error distribution covers an interval of 22 ns, essentially due to the 8-ns precision in the DW1000 TX scheduling, accumulating over 4 hops. In the worst case, the TX scheduling error is always exactly 8 ns, yielding a theoretical maximum error of 32 ns for our setup; in practice, the random variations of TX times often cancel each other. In any case, the standard deviation of the error is 3.89 ns, i.e., *almost three orders of magnitude smaller than in narrowband*, reported in [56] to be 2.5 µs over 4 hops.

3.3.3 Exploring the Limits

As we observe the occurrence of packet loss in our testbed setup, we investigate the conditions that can hamper concurrent transmissions in our UWB implementations of Glossy. To this end, we collect empirical evidence in a different, smaller-scale setup where we can precisely control the overlapping of signals. We position a receiver in between 2 synchronized transmitters, at 1 m distance from each, and evaluate the effect of data symbol misalignment and crystal accuracy. This placement is particularly challenging because the strength of concurrent signals is similar at the receiver. However, it allows us to derive stronger conclusions regarding the limitations and the ideal conditions of concurrent transmissions, as well as provide evidence that the signal strength and the number of available receivers play a role in the robustness of Glossy.

Payload collisions. UWB data symbols are divided in two halves for binary burst position modulation (BPM, §2.1). In principle, a node may fail to receive correctly when two concurrent transmissions are shifted by more than half a symbol duration, and one of the pulse bursts occupies the wrong BPM location. The ability to ensure that different transmissions occupy the same BPM locations has been reported as a *necessary* condition to prevent collisions [78]. Specifically, pulse bursts should remain within the T_{BPM} duration, i.e., 64.105 ns for the 6.8 Mbps data rate used in our setup, resulting in severe limitations on the position of nodes.

However, our earlier experiments in Chapter 2 hint at the fact that this is actually not at all crucial. To verify this hypothesis, we trim the clocks of the DW1000 for a short duration before packet transmission to schedule TXs with a resolution of ~ 1 ns (circumventing the 8 ns resolution limitation). We then delay one of the transmitters to cause different degrees of signal overlapping along the symbol duration, in steps of 1 ns. Figure 3.7 shows that the receiver enjoys a $PRR \geq 98\%$ even when the delay we artificially introduce is > 64.105 ns, causing pulses to occupy the opposite side of the data symbol. This shows that the DW1000 radio is able to decode the packet correctly even in the presence of pulse bursts in erroneous locations. We speculate that this is due to the coherency mechanisms built in the DW1000 receiver, that allow

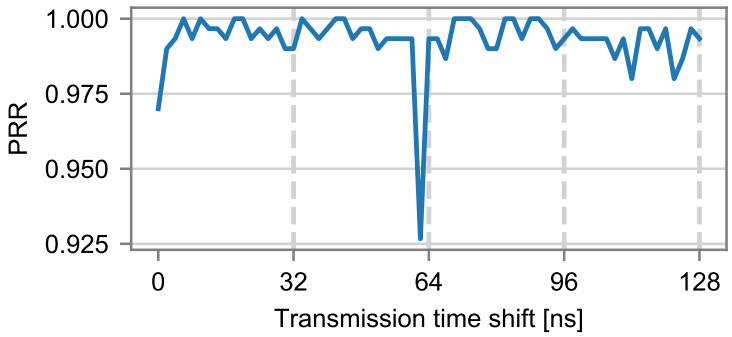


Figure 3.7: *PRR* when a transmission is shifted over the symbol duration (short 15 B frame).

correct decoding based on the phase of the signal alone. On the other hand, Figure 3.7 also shows that the decoder is affected by concurrency when pulses are really close to ~ 64 ns, i.e., when they occupy a location matching the time-hopping sequence on the opposite side of the BPM-BPSK symbol. However, this constraint is unlikely to happen and even less likely to disrupt a Glossy flood, thanks to spatial diversity and inherent variations of TX times.

Our findings significantly relax previously reported requirements [78] that, by indirectly affecting the physical location of network nodes, would otherwise hamper the practical applicability of concurrent transmissions over UWB.

Frequency offset. Another necessary requirement reported in [78] is about coherency of two overlapping signals throughout the whole frame. In other words, the phase drift caused by the oscillator frequency difference of two concurrent transmitters should never go beyond half the oscillation period within the frame transmission. In [78], this is translated into a maximum clock frequency offset (CFO) of 1.39 ppm for 33B frames. The same calculation, applied to the maximum-length packets of 127B allowed by the standard and considered in §3.3.2, yields $0.5/3.4944 \times 10^9/252 \times 10^{-6} = 0.57$ ppm, where 252 µs is the packet TX time with the 64 µs preamble we use. This value is quite far from the 1.45 ppm we achieve with frequency calibration (§3.3.1). Indeed, when transmitting long packets in our controlled (and challenging) small-scale experiments above, we observe a rather unstable *PRR*, in many cases $< 10\%$ regardless of whether the calibration is used or not. Unfortunately, further improvements are possible only with a custom hardware platform, different from the popular EVB1000 we use in our experiments.

On the other hand, to elicit further insights about this constraint as well as assess its impact on short packets, we use a small-scale setup where we concurrently transmit 12200 rounds of short packets and simulate the presence of a transmitter with poor crystal accuracy by artificially altering the oscillator frequency, albeit within the ± 20 ppm tolerance required by the DW1000 [70]. First of all, we confirm that high CFO is not a problem in the case of *isolated* transmissions. In our experiments, a single transmitter with the artificial frequency offset of 10 ppm yields $PRR \geq 99\%$, as expected given that this offset is within the DW1000 tolerance. However, when

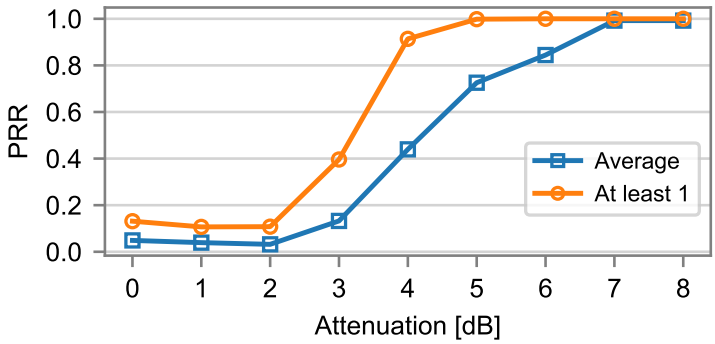


Figure 3.8: Reliability with 4 receivers and 2 transmitters with relative TX attenuation (long packets): average and at-least-one *PRR*.

multiple concurrent transmitters are present and one is configured with the same artificial 10 ppm offset, we obtain $PRR \geq 81.17\%$, while dynamic frequency calibration (§3.3.1) yields $PRR \geq 96.74\%$. This confirms that 1. the frequency offset matters even for short packets, and 2. dynamic frequency calibration is effective in improving their reliability.

On the other hand, we observed these drops in reliability only when *artificially* introducing a frequency offset, as witnessed by the perfect reliability shown in §3.3.2 for both Glossy variants. In practice, the EVB1000 platform we use, factory-trimmed at 3 ppm, guarantees perfect reliability for short packets even without dynamic frequency calibration. However, the latter may play a role in deployment environments harsher than the indoor one where we performed our experiments.

Receiver redundancy and TX power. The number of available receivers at a given hop and the relative TX power of transmitters are important factors in the reliability of our UWB Glossy variants, similarly to the narrowband ones [86]. This aspect, largely neglected by related work [1, 78], would deserve a more exhaustive analysis than what is possible here. Nevertheless, we offer empirical evidence about it in our small-scale setup with 2 co-located nodes transmitting long packets—the most unreliable—and 4 receivers, again in the most challenging placement where they are at essentially the same distance from transmitters. Further, we configure the transmitters with a relative TX attenuation of 0–8 dB. Figure 3.8 shows that when the transmitters use the same TX power, the average *PRR* is very low; nevertheless, it nearly doubles if computed by considering a reception successful when it occurs on *at least one* of the 4 receivers. Further, it also shows that the *PRR* rapidly grows with the difference in TX power. An attenuation of 7 dB ensures that *all* receivers get the packet; 5 dB are sufficient to ensure reception by at least one of them. These considerations are important, as one successful receiver is enough to enable a Glossy flood to progress. On the other hand, the dual also holds; a topology in which progress is ensured by a single forwarder is obviously very brittle. Incidentally, this is the reason why we placed nodes $A - F$ in the corners of our testbed, that is, to eliminate these single-receiver bottlenecks that should anyway be avoided in real deployments.

3.4 Crystal on UWB

The previous section confirmed that our implementations of Glossy for UWB radios provide benefits comparable to those known from the narrowband literature. We now turn our attention to a different research question, namely, whether the results from higher-level abstractions and protocols built atop the Glossy layer also transfer to UWB. To provide an answer, we focus on the Crystal protocol [60, 84] described in §3.1. We discuss the few changes our Crystal implementation for UWB required w.r.t. the original one, followed by the results of its evaluation in our testbed.

3.4.1 Implementation Highlights

We used the publicly-available code for Crystal, and kept the overall protocol logic unchanged; we disabled channel hopping [84], as it is not the focus of this chapter. However, a few minor modifications were necessary, motivated by the different operation of the underlying radios.

Crystal detects termination based on the absence of received packets; it is therefore crucial to tell apart absent transmissions from failed receptions. In narrowband, noise detection enabled Crystal to defer termination if no packet is received but strong noise is detected. However, this mechanism relied on clear-channel assessment (CCA), not present in UWB. Further, it did not provide direct evidence of a failed RX, but only of the *possibility* of one, due to noise.

On the other hand, the DW1000 offers rich informationabout RX errors, which we exploit in our implementation. This information is signaled when the radio detects a preamble but fails to decode either the SFD or the data portion of the packet, due to Reed-Solomon, SECDED or CRC errors (§2.1). A "spontaneous" preamble detection may still happen without any TX, but is highly unlikely in practice. On the other hand, the mere presence of a preamble signals that one or more nodes are sending packets but their data cannot be decoded, probably because of collisions. We verified that these techniques significantly improve the reliability of our UWB implementation of Crystal.

Finally, our UWB implementation relies on the MCU 32 kHz timer of the EVB1000 board to schedule its activities; data TX in the shared T slots may therefore overlap within 30 µs. Relying on the more accurate radio clock would require significant changes in the code, in contrast with our desire to minimize them. Further, it would likely bring little or no benefits, given that a slight de-synchronisation of transmitters is shown in Chapter 2 to increase the reliability of concurrent transmissions of *different* packets. Following another insight from our previous study, we also increase the SFD timeout by 32 µs, to account for possible frame offsets caused by the timer resolution.

3.4.2 Evaluation

We evaluate Crystal in our testbed, with the same configuration of §3.3.2; further, node 9 is the sink, maximizing the network diameter. We are interested in the overall reliability in delivering packets at the sink, but also in ascertaining the underlying

raw reliability of the floods disseminating *different* packets and competing in the same shared Crystal slot. In doing so, we experiment with both variants of the underlying Glossy, as well as with short and long packets.

We consider two key parameters influencing performance: the number U of concurrent updates and the number N of TX in the Glossy flood inside a shared Crystal slot. U determines how many nodes transmit a packet in each Crystal epoch, defining the degree of concurrency. We explore $U \in \{1, 2, 5, 7, 10\}$, i.e., up to half of the network, chosen at random among non-sink nodes; we also consider the extreme case where all the U=21 non-sink nodes transmit concurrently. N defines the degree of redundancy at the Glossy layer; we explore $N \in \{1, 2, 4, 8\}$ as in §3.3.2. We set a default of U=5 and N=2 when exploring the other parameter.

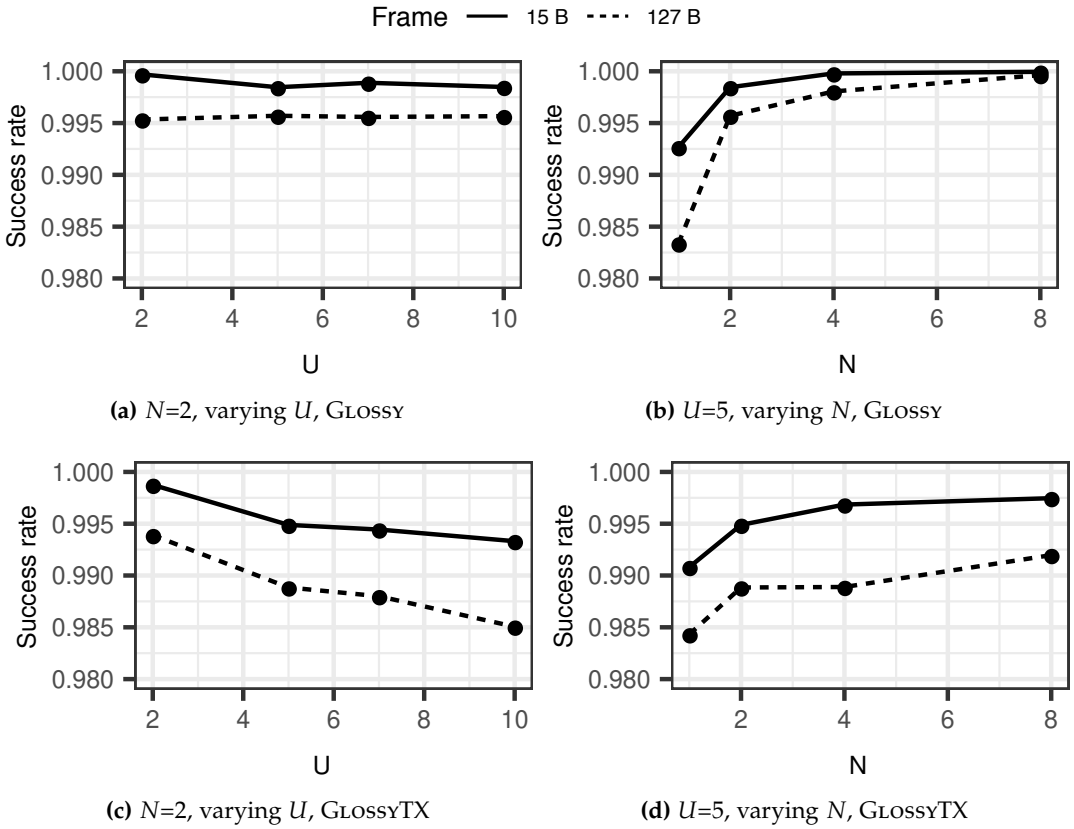
For each combination in this space we collect traces of 1000 Crystal epochs, i.e., $1000 \times U$ packets transmitted.

Overall reliability. Our UWB implementation of Crystal ensures remarkable reliability with both short and long packets. With the short 15B packets, Crystal correctly delivered *all* the $>$ 150000 packets transmitted, regardless of the parameter configuration, and notably even when U=10 nodes (half of the network) were concurrently transmitting. Moreover, even with the longest 127B packets Crystal achieves >99.9% reliability. This near-perfect reliability is fully in line with the results originally reported in [60], therefore confirming that the performance Crystal achieves in narrowband can be harvested also in UWB.

Reliability of shared slots. This remarkably high reliability is achieved in Crystal via mechanisms that *mask* the packet losses in the underlying concurrent Glossy floods. Therefore, in the light of ascertaining the extent to which concurrent transmissions of *different* packets can be used as a building block for other protocols [59, 61, 85], we now focus on the performance of shared slots in isolation.

Specifically, we look at the first T slot of each Crystal epoch, when there are exactly U nodes transmitting simultaneously. For each node, we define the success rate metric as the ratio between the number of floods when the node received *any* packet over the total number of floods when the node was listening in the first T slot, as in [60]. Figure 3.9 shows the average success rate, and Figure 3.10–3.11 the distribution of this metric across nodes via the complementary empirical cumulative distribution function (CCDF).

The charts exhibit clear trends. First of all, Glossy systematically outperforms GlossyTX across all configurations. In particular, the reliability of Glossy remains nearly constant w.r.t. the increase in the number U of concurrent senders, while GlossyTX shows a marked decline. This is a consequence of the fact, already pointed out in §3.3.2, that the density of transmissions in GlossyTX is much higher than in Glossy, and obviously exacerbated as U increases. Long frames degrade reliability of both variants, again consistently with §3.3.2, and with a more marked effect on GlossyTX, as per the observations above. In any case, the worst success rate recorded across all these many experiments was 95%, which is still very good.



(a) N=2, varying U, Glossy

(b) U=5, varying N, Glossy

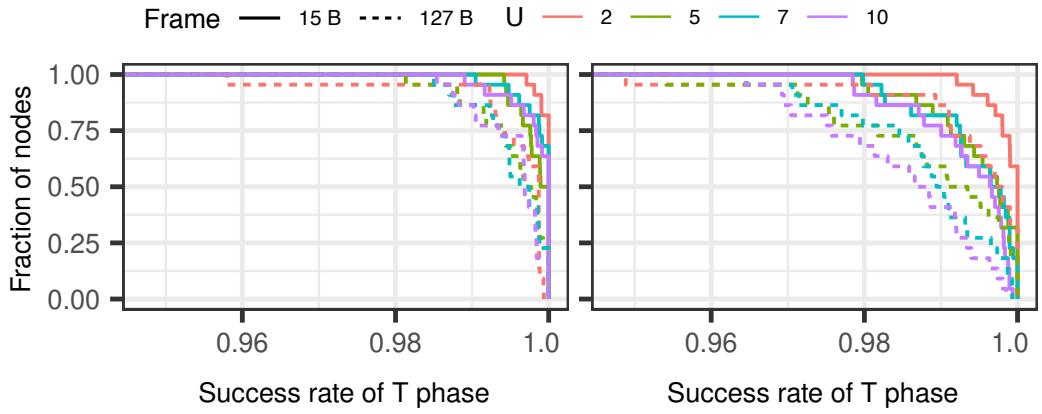
(c) N=2, varying U, GlossyTX

(d) U=5, varying N, GlossyTX

Figure 3.9: Average success rate in the first T shared slot, for different values of N and U.

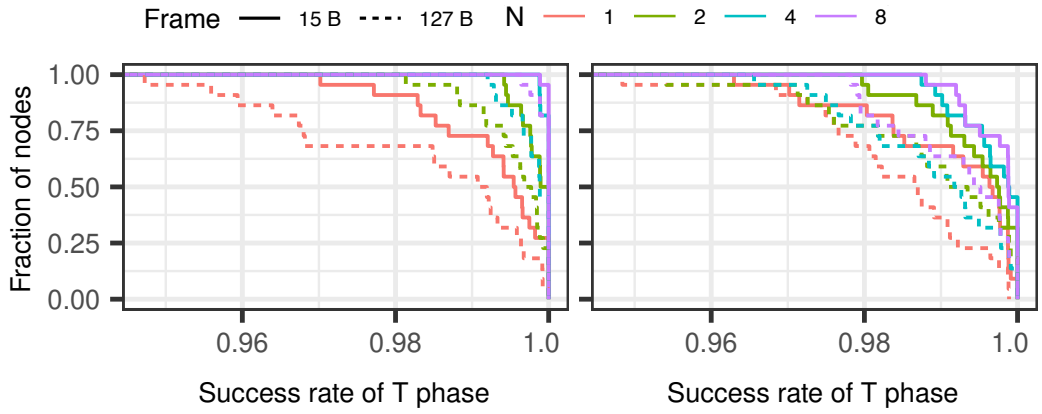
Finally, reliability increases with the degree of redundancy induced by N. At the highest value tested, N=8, the average reliability of Glossy reaches 99.994% with short packets and 99.96% with long ones, caused by packet losses at a single node. As for GlossyTX, the average reaches a plateau of 99.7% for short packets and 99.0% with long ones. These figures are remarkable, considering that 1. they are achieved with concurrent transmissions of *different* packets, and 2. *without* the reliability mechanisms of Crystal, which are nonetheless key to spare the steep energy costs induced by a high value of N.

Extreme case. We also tested Crystal in the extreme scenario where *all* 21 non-sink nodes transmitted in all epochs (U=21). With N=2, the success rate of the T phase drops to as low as 80% even with short packets; nevertheless, the overall reliability at the sink remains at 99%. As expected, long packets exhibit worse reliability, with a success rate of the T phase at ∼75% and an overall reliability of ∼95%. At the other extreme, N=8 yields a near-perfect overall reliability for short packets, with several runs at 100%, and ∼98% for long packets, despite an underlying success rate at ∼88% and ∼79%, respectively. Overall, these results confirm once again the effectiveness of the "safety net" provided by Crystal's reliability mechanisms, as already observed for narrowband in similar extreme scenarios [60, 84].



(a) GLOSSY **(b)** GLOSSYTX

Figure 3.10: CCDF for T success rate vs. U (N=2).



(a) GLOSSY **(b)** GLOSSYTX

Figure 3.11: CCDF for T success rate vs. N (U=5).

3.5 Discussion

We distill salient findings from our results and offer some considerations that may inspire future work on the topic.

Similarities vs. differences. First and foremost, our experiments demonstrate that Glossy-like mechanisms achieve in UWB benefits similar to narrowband, i.e., low latency, high reliability, low energy consumption——all at once. Energy consumption has slightly different tradeoffs than in narrowband, due to the significant imbalance between RX and TX in the DW1000. On the other hand, the latency achievable in UWB is significantly lower than in narrowband, as a consequence of the high-accuracy clock and the higher data rate. The former also enables *a three order of magnitude improvement* in the accuracy of network-wide time synchronization, which was actually the original motivation for Glossy.

Interestingly, our in-depth analysis of §3.3.3 reveals that the key PHY-level mechanism enabling Glossy appears to be more *non-destructive interference* than a con-

structive one. Indeed, the DW1000 is capable of decoding concurrent pulses—and therefore packet TX—even when severely misaligned. On the other hand, due to the encoding based on short pulses rather than long waves, the crystal frequency offset is more of an issue for UWB than it is for narrowband, with a stronger impact on the tradeoff between the packet length of the concurrent transmissions and their reliability. However, as for narrowband, a difference in the signal energy of concurrent transmissions and multiple receivers improve performance considerably even for long frames.

Glossy or GlossyTX? A first observation is that the question is actually an open one for narrowband. Indeed, the potential superiority of GlossyTX is rather anecdotal, as it derives from the ad hoc setup of the EWSN Dependability Competition and has never been rigorously analyzed across different system parameters.

For UWB, our study shows that GlossyTX is more energy-efficient than Glossy; for long packets, it achieves a consumption even lower than narrowband. Therefore, it would seem obvious to always use it in place of Glossy. Nevertheless, our study shows that this is not always necessarily the case. Indeed, the other side of the coin is that the aggressive re-transmission policy of GlossyTX is prone to increasing the number of collisions, affecting reliability. This behaviour can be observed for both same-packet and different-packet floods. However, the actual impact ultimately depends on how the Glossy layer is used in the specific traffic profile and/or higher-level system (e.g., Crystal, in our case).

The dual argument is that Glossy appears slightly more reliable, due to the alternating pattern of TX and RX slots that reduces the "density" of concurrent senders and thus the probability of collisions. An opportunity for future work is to find a scheme striking the right balance between the back-to-back transmissions of GlossyTX and the sparser transmissions, yet rigidly alternating with receptions, of Glossy.

Transferring results from narrowband to UWB. As repeatedly mentioned, there is a substantial literature on concurrent transmissions for narrowband, including systems that built atop the original Glossy to support alternate network functionality [59, 61, 60, 85].

Our experience with "porting" Crystal to UWB and the related evaluation (§3.4) shows that the effort required is relatively small while, on the other hand, the benefits that can be attained are entirely in line with those shown for narrowband. Of course, it would be a leap of faith to claim that the same can be done for all other higher-level abstractions in the literature. However, our experience hints at the fact that this may actually be the case for several of them, especially those that run atop an unmodified Glossy layer, e.g., notably including LWB [59].

We argue that pursuing this question is actually important, to amplify the impact (and awareness of) the body of literature on concurrent transmissions on other radio technologies and hence research communities, and concretely demonstrate that it applies to a far more general scope than the hardware niche it was originally developed for.

3.6 Conclusions

We explored the extent to which concurrent transmissions, made popular by Glossy for IEEE 802.15.4 narrowband, can be exploited in UWB via a full-fledged, readily-available system, and ascertained what is the corresponding performance. Overall, the answer is very positive: both variants of Glossy we consider, as well as Crystal, the higher-level abstraction building atop of it, yield in UWB a reliability similar to the one observed in narrowband. Further, the higher clock accuracy and data rates in UWB unlocks significant latency improvements, which become order-of-magnitude ones for time synchronization, the original motivation of Glossy. We provided a detailed account of the opportunities this UWB platform enables for an efficient implementation of concurrent transmissions, an analysis of the threats to performance, as well as investigated the effort required to exploit Crystal atop the Glossy layer.

Beyond the qualitative lessons learned and quantitative results reported here, we also release the systems we described as open source [73], enabling their immediate use and improvement by researchers and practitioners, and generally inspiring future work on the topic.

4

One Flood to Route Them All: Ultra-fast Convergecast of Concurrent Flows over UWB

In the last decade, concurrent transmissions (CTX) have catalysed the attention of the low-power wireless community. As discussed in Chapters 2 and 3, the term refers to the fact that tightly-synchronized simultaneous transmissions do not necessarily result in a collision; instead, under some conditions determined by the underlying PHY radio layer (§4.1), one of the concurrent packets is received with very high probability.

Motivation. This phenomenon was exploited in IEEE 802.15.4 narrowband where, popularized by the Glossy system [56], it rapidly became a state-of-the-art asset in designing protocols supporting various traffic patterns [59, 61, 60]. Recent work has shown that the same principle is applicable to other radios, notably including Bluetooth Low Energy (BLE) [89]. Our previous work demonstrates the feasibility of the technique in UWB.

This surge of interest is motivated by the fact that protocols based on CTX enable unprecedented performance by achieving at once near-perfect reliability and very low latency and energy consumption. Nevertheless, most of these protocols are built atop the network-wide flooding offered by Glossy, by simply scheduling its floods in different ways depending on the traffic pattern and goals at hand. In other words, Glossy is used to a large extent as a *monolithic blackbox*, with little or no modifications by higher-layer protocols. This is a reasonable design decision enabling faster and

This chapter revises our publication [3]: M. Trobinger, D. Vecchia, D. Lobba, T. Istomin, and G. P. Picco. "One Flood to Route Them All: Ultra-fast Convergecast of Concurrent Flows over UWB". In *Proceedings of the* 18^{th} *ACM Conference on Embedded Networked Sensor Systems (SenSys)*. 2020.

reliable prototyping, given the system-level complexity of Glossy. On the other hand, this decision stifles the exploration of alternate, finer-grained designs that *directly and individually* exploit CTX, free from the rigid, predefined scheduling strategy of Glossy.

Exploiting the full potential of CTX: Weaver. This second approach is precisely the one we follow in this chapter, in which we retain network-wide flooding as the main communication mechanism, but fundamentally change its purpose and operation.

A Glossy flood is *entirely* dedicated to disseminating a *single* packet across the entire network. A global scheduling of transmission (TX) and reception (RX) slots, whose redundancy is governed by the user-defined parameter N, achieves the aforementioned excellent performance. Nevertheless, flooding is still a wasteful, network-wide operation, exacerbated by the fact that reliability is directly tied to the redundancy factor N; the higher its value, the higher the number of times a packet is (concurrently) retransmitted and, therefore, the higher the (network-wide) energy consumption. Crucially, the value of N, and hence the duration of the flood, is fixed before execution, therefore intrinsically prone to over- or under-provisioning, hampering lifetime and reliability, respectively.

In contrast, the system we describe here, called Weaver, is expressly designed to *concurrently disseminate towards a receiver different packets from multiple senders in a single, self-terminating, network-wide flood*, significantly improving on latency, reliability, and energy consumption w.r.t. Glossy-based systems.

Weaver achieves these goals with several mechanisms, each built directly atop individual CTX. As in all Glossy-based systems, Weaver alternates (short) periods executing its network-wide flood with (long) periods of inactivity, all implicitly time-synchronized by system operation. Each node, including the sink, executes a time-slotted sequence formed by a TX slot followed by *two* RX slots, that repeats until the flood self-terminates. Adding an extra RX slot to the TX-RX scheme of Glossy may seem a minor change; yet, it is crucial to unlock significant performance gains. Indeed, the resulting 3-slot structure, combined with the propagation of an initial message from the sink, staggers the (concurrent) TX and RX of nodes at different hop counts from the sink, enabling multiple flows to co-exist within the same flood without disrupting each other. Further, it enables Weaver to exploit a combination of *local*, 1-hop broadcast acknowledgments and *global*, sink-initiated ones that, together, adaptively 1. suppress unnecessary packet propagation or, on the contrary 2. trigger retransmission of packets that have been lost, therefore decreasing energy consumption and increasing reliability w.r.t. the fixed redundancy of Glossy-based approaches.

Goals, methodology, contributions. We discuss the design rationale and goals for Weaver (§4.2) and offer an analytical model confirming the intrinsically superior performance achieved w.r.t. Glossy-based state-of-the-art representatives, before delving into a more in-depth illustration of protocol details (§4.3).

Systems based on CTX are notoriously complex. This, however, is to a large extent a relic of a past when the lack of proper hardware primitives required complex designs

yielding timing guarantees. Nowadays, many radio transceivers offer rich primitives notably including the ability to schedule transmissions accurately, including the DecaWave DW1000 UWB radio [70] we focus on in this chapter. In addition to our Glossy implementation over UWB, other works [78] have shown that Glossy-based protocols can be effectively supported by its powerful features.

Moreover, we observe that a staple communication stack, and in particular a convergecast one, is currently missing for UWB, in stark contrast with the plethora of protocols resulting from more than a decade of work on IEEE 802.15.4 narrowband wireless sensor networks. By providing a fast and efficient data collection embodied by our Weaver prototype we aim at fostering adoption of UWB also for sensing and communication, besides ranging and localization.

Once the leap is made from coarse-grained, rigid Glossy floods to finer-grained alternatives based on individual CTX, many solutions are possible, catering for different requirements. Our *modular* implementation (§4.4) sharply decouples the mechanics of accurately scheduling TX and RX slots, delegated to a Time Slot Manager (TSM) component, from their higher-level orchestration in Weaver, which can be easily replaced by alternate designs. A component estimating energy consumption is also provided. We release these reusable components as open source [73] contributing to further developments in the fast-growing UWB research community.

We evaluate Weaver in a 36-node testbed at our premises. We first analyze the impact of key design decisions with dedicated experiments. Next, we compare directly the performance of Weaver against Crystal [60, 84] a state-of-the-art Glossy-based convergecast protocol with an UWB implementation [2]. Our results confirm the trends observed in the analytical model, e.g., showing that Weaver can deliver at the sink 30 concurrent flows in $\sim$100 ms, achieving a reduction of $\sim$70% in both latency and energy consumption w.r.t. Crystal while achieving near-perfect reliability due to the lower contention induced by the finer-grained, adaptive use of CTX. Moreover, the ultra-fast dissemination achieved by Weaver, along with the inherent redundancy offered by CTX, makes our system resilient to topology changes, e.g., induced by mobility.

Finally, although our Weaver prototype targets UWB, its protocol design does not rely on features specific of this PHY layer. Therefore, it can be applied to other radios, amplifying the impact and contribution outlined here and pushing the envelope of what CTX can achieve in low-power wireless communications at large. We concisely discuss these and other follow-up opportunities (§4.6) before ending the chapter (§4.7) with brief concluding remarks.

4.1 Background and Related Work

We offer the necessary background on CTX in narrowband and UWB, along with a concise survey of the most relevant approaches.

4.1.1 CTX Protocols in IEEE 802.15.4 Narrowband

CTX protocols can support various traffic patterns using Glossy as the underlying communication engine, but examples exist that rely on finer-grained structures.

Glossy as a reusable building block. Glossy [56] was the first to exploit CTX into a reliable, efficient, and publicly-available system providing network flooding and time synchronization. Several others exploited the low-latency, high-reliability, low-consumption, network-wide flooding of Glossy as a building block for higher-level abstractions. LWB [59] supports different traffic flows (many-to-one, one-to-many, one-to-one) by properly scheduling them as individual Glossy floods from a single initiator. Crystal [60, 84] supports many-to-one convergecast via phases in which Glossy floods from multiple initiators compete, followed by others in which the sink alone has the opportunity to acknowledge the received packet. Other systems [90, 91, 92, 93] explore variants of these concepts.

Reusability vs. degrees of freedom. Interestingly, in all these systems Glossy is reused as a *monolithic blackbox*, with little or no modification. Indeed, only few systems make relatively small modifications to Glossy that, however, are not geared to change its core functionality, rather to improve its performance, e.g., increasing reliability via channel hopping [85] and/or reducing latency [81, 82].

The direct reusability of Glossy actually fueled research on CTX, enabling researchers to experiment with new protocols while avoiding the intricacies of the CTX implementation. Nevertheless, at the same time it also fossilized research on CTX to a large extent.

Indeed, a Glossy flood is entirely dedicated to disseminate a *single* packet from a *single* initiator, with the intent to exploit constructive interference among forwarding nodes. LWB and others (e.g., [90, 92]) rely directly on this feature. Crystal and others (e.g., [91]) push Glossy further by having *multiple* initiators compete within the same flood, relying on the capture effect; however, the final outcome is still a single packet from only one of the initiators.

Weaver: Back to individual CTX. In contrast, we take a significantly finer-grained perspective and free ourselves from the mechanics of Glossy, exploiting CTX directly and individually.

Only few systems hitherto resorted to a similar approach, and always to support many-to-many communication. Chaos [61] realizes network-wide aggregation of data from multiple initiators via competing floods. Mixer [94] and CodeCast [80] exploit network coding to improve performance and reliability. In all of them, the TX-RX scheme of Glossy is replaced by a sequence of indistinct slots in which a node dynamically decides whether to TX or RX; in Chaos, a TX happens deterministically when the node observes new information affecting the global aggregate, while in Mixer and CodeCast the decision includes also a probabilistic component.

Our research endeavor differs from the above on two accounts. First, it explores a strategy in which slots are not indistinct, rather they have a preassigned role, as in Glossy. However, differently from Glossy, our scheme is capable of deterministically

intertwining multiple flows from multiple initiators whose dissemination and termination we govern with a novel, adaptive strategy as described in §4.2 and §4.3. Second, instead of many-to-many, we tackle convergecast traffic. This communication pattern is arguably more popular, thanks to its use in monitoring and data collection applications, yet hitherto dominated by systems relying on monolithic Glossy floods. By "breaking" this unit of communication and achieving better performance via individual CTX we exemplify the power of this alternate design mindset, possibly paving the way to exploration of alternate schemes catering for this and other traffic patterns.

Finally, to facilitate this exploration by others, and simplify our own system development, we follow a recent trend [95, 85, 94] and sharply separate the fine-grained CTX engine from the Weaver protocol built atop it, while taking advantage of features provided by modern transceivers that greatly simplify programming.

4.1.2 CTX Protocols in IEEE 802.15.4 Ultra-wideband

The remarkable performance achieved by CTX in IEEE 802.15.4 narrowband led researchers to study whether and how the same concept applies to other radios, e.g., Bluetooth Low Energy (BLE) [89] and IEEE 802.15.4 UWB. The answer is not for granted, given the significantly different characteristics of the PHY layers. However, these studies have shown that both same-packet and different-packet CTX can be exploited, and their benefits still stand. In this chapter we focus specifically on the Decawave DW1000, arguably still the most popular and widely-available UWB transceiver today.

Towards a communication stack for UWB. Our focus on UWB is motivated by the following reasons. First, UWB is increasingly popular, as it offers both accurate distance estimation and communication. Although UWB is not as pervasive as other radios, its inclusion in smartphones by Apple and Samsung hints that this may change soon. Second, a staple UWB communication stack is currently missing, in contrast with the many available for IEEE 802.15.4 narrowband after almost two decades of work on wireless sensor networks. By focusing on convergecast, a staple feature of the latter, and leveraging the superior performance of CTX, we aim at concretely showing the effectiveness and applicability of UWB to sensing and communication scenarios, accelerating its adoption beyond localization-centric ones.

Glossy on UWB. A third motivation is that Glossy-based baselines already exist. The work in [78] reports the use of Glossy on UWB for network-wide coordination in a localization system. However, as communication is not the main focus, implementation details are scarce and no performance evaluation is provided. Our work in Chapter 3 details instead several design opportunities enabled by the DW1000 and evaluates Glossy on UWB in a testbed. Moreover, it also shows how Crystal, a representative higher-level protocol supporting convergecast (§4.1.1), can be "ported" to UWB with minimal modifications, yielding remarkable performance akin to the state-of-the-art one observed for narrowband [60]. Hereafter, we use the publicly-available UWB implementation of Crystal as the baseline we compare Weaver against.

These systems are all enabled by nearly-simultaneous packet transmissions, similar to narrowband albeit with slightly different temporal and environmental constraints. For an in-depth analysis of the latter, and a comparison with narrowband, the reader is referred to Chapter 2. Nevertheless, we reassert that the main contribution of this chapter lies in the novel idea of merging multiple packets flows in a single network-wide flood—a notion that is not tied to UWB and therefore applicable to other radios, as further elaborated in §4.6.

4.2 Design Goals and Principles

We designed Weaver to tackle inherent inefficiencies of data collection protocols that rely on Glossy floods as the only communication primitive. To better understand the crux of the matter, we focus on Crystal and analyze critically its operation, in particular its reliance on Glossy and the related shortcomings (§4.2.1). Motivated by this analysis, we then provide a concise overview of the key design decisions and goals at the core of Weaver (§4.2.2). We conclude the section by providing a quantitative argument, supported by analytical models, showing that our fine-grained CTX-based design is intrinsically superior to Glossy-based ones (§4.2.3).

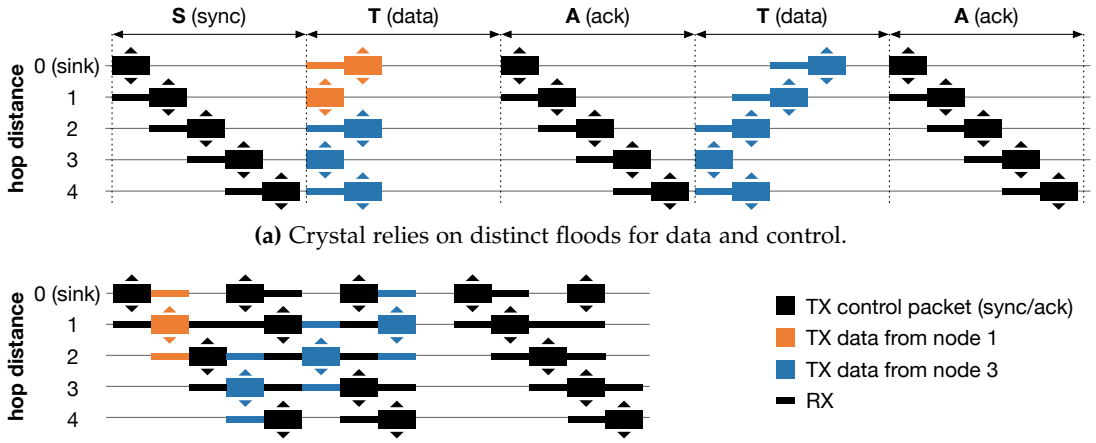
4.2.1 The Drawbacks of a Glossy Legacy

Crystal [60], introduced in Chapter 3, targets scenarios with aperiodic data collection and sparse traffic, using a schedule (Figure 4.1a) composed of three phases, each executing a Glossy flood. Crystal operations can be summarized as: 1. the initial S phase for time synchronization; 2. the T phase, where concurrent floods for different packets compete, and one is received at the sink with high probability, e.g., the orange one in Figure 4.1a; 3. the A phase that originates at the sink, for network-wide acknowledgment informing senders of whether their packet has been received; in Figure 4.1a, this enables retransmission of the blue packet in the next T phase. The alternation of T and A phases (TA pair in Crystal jargon) continues until all pending packets are received and acknowledged, and a TA pair without data is observed for a pre-defined number of times. For further details, the reader is referred to §3.1.2.

Although Crystal already achieves remarkable, state-of-the-art performance, it is inherently limited by its direct reliance on Glossy, as many others (§4.1).

A first problem is that *each Crystal phase is a* Glossy *flood that must complete before a new one is started*. The schedule on each node must allocate enough time for flood propagation, determined with knowledge of the network diameter and number of retransmissions. This causes a significant increase in the total time required for data collection.

The problem is exacerbated by the fact *that the number N of retransmissions is also fixed*, yielding other inefficiencies: 1. retransmissions are performed regardless of whether a packet has already successfully propagated, therefore hampering latency, lifetime, and possibly reliability due to unnecessary contention, and 2. the fixed value of N cannot dynamically cater for transient sources of unreliability, common in wireless



(a) Crystal relies on distinct floods for data and control.

(b) WEAVER weaves data and control into the same flood.

Figure 4.1: Sample executions of Crystal and WEAVER. Termination phase is not shown.

communications; either the worst case is assumed, hampering lifetime in the normal case, or the latter is assumed, hampering reliability when disturbances occur.

These problems are shared by all Glossy-based approaches (§4.1). Specific to Crystal is the sink-initiated, network-wide acknowledgment in the A phase. The latter has been shown (in [60, 84] and EWSN competitions) crucial to achieve near-perfect reliability even with aperiodic, bursty traffic and heavy interference. This superior reliability motivates our use of Crystal as a baseline instead of, e.g., LWB or derivatives, besides the lack of UWB implementations. However, the asset brought by A phases also bears drawbacks, again inherited from Glossy. One directly descends from the problems above: successful propagation of a packet requires (at least) two phases, transmission (T) and acknowledgment (A), both fixed-length and strictly separated, wasting energy and time.

A less obvious problem is that *the A phase is oblivious of the reason why packets are not received*. In Crystal, the common case is that transmissions from U initiators compete in the same T phase; the A phase is crucial to inform senders of whether their packet should be re-sent. Nevertheless, the A phase also counters packet losses due to collisions. These are often concentrated in network "pockets" where packet transmissions violate the constraints for successful CTX, yielding a collision. This situation stems from a combination of neighbor density, relative signal power, and environmental conditions, extremely hard to predict yet likely to repeat due to the periodic operation of the protocol. Unfortunately, in Crystal the only option is to inform the network about the missed packet, and hope that somehow the problem solves itself.

4.2.2 Weaver: The Power of Fine-grained CTX

We tackle the limitations above at their core by removing the dependency on Glossy, therefore regaining the full degrees of freedom available once the unit of communication becomes an individual CTX rather than a monolithic Glossy flood. This finer-grained design mindset enables us to bring to the table several techniques that,

together, improve significantly the already remarkable performance of Crystal, in its role of representative of Glossy-based approaches. The most significant point of departure is that WEAVER collects and acknowledges multiple packets *within a single flood*, where the different flows coexist without disrupting each other.

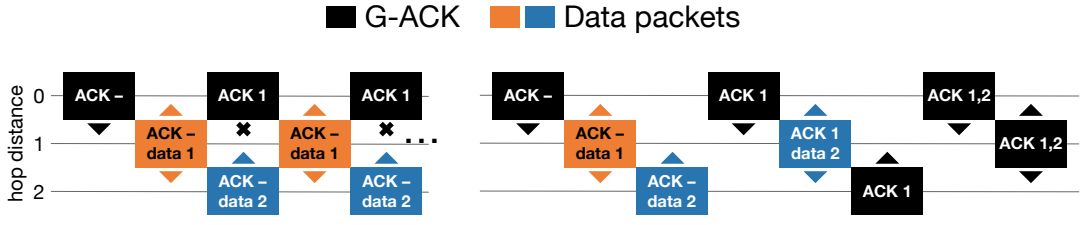
On the surface, WEAVER resembles other CTX-based convergecast protocols, e.g., Crystal or LWB. Time is divided in rounds (*epochs*) of fixed length, each containing a time-slotted communication schedule. The *sink*, i.e., the node collecting data, periodically starts a new epoch by broadcasting a synchronization packet; nodes repropagate it concurrently, exploiting CTX, and align their slots to the one of the sink, beginning execution of the global schedule.

Epoch bootstrap: Acquiring one-shot topology information. Differences begin with this first step, which in WEAVER is exploited not only to enable nodes to time-synchronize, but also to *learn their hop distance from the sink*. This topology information is exploited by a node to relay only packets from nodes at the same hop distance, or higher, from the sink, i.e., favoring packets in need to make progress and quenching those already ahead, reducing contention.

This *topology information is not explicitly maintained*, as in conventional route-based approaches, *rather passively learned* during the initial dissemination, hereafter termed as *epoch bootstrap*. In this respect, the ultra-fast dissemination achieved by WEAVER doubles as a fundamental asset for its operation. As we show later (§4.5), WEAVER disseminates 30 flows over 6 hops in only ∼100 ms. Therefore, *during this very short time span the network can be effectively considered as static*, and the topology learned during bootstrap safely assumed to persist throughout the entire flood, even in scenarios with node and/or sink mobility, as we investigate in §4.5.4.

Weaving packet flows. WEAVER merges flows from multiple senders (*initiators*) into a single flood where data *implicitly* follows the upward gradient towards the sink established by the epoch bootstrap, and acknowledgments flow downwards towards initiators.

This goal is intrinsically at odds with the classic GLOSSY schedule alternating a TX slot with a RX one, which causes floods two hops apart to *systematically* compete. A node in RX mode hears TX from both nodes on the next and previous hop towards the sink, with the latter potentially halting propagation of data upstream. For instance, in Figure 4.2a, the TX of the ACK from the sink for the orange packet is performed concurrently with the TX of the blue packet. If the latter prevails, the ACK is lost and the orange packet must be retransmitted, as shown. Otherwise, the ACK prevails



(a) TX-RX schedule (Glossy). **(b)** TX-RX-RX schedule (WEAVER).

Figure 4.2: Weaving flows of data and acknowledgments.

and delays the TX of the blue packet. Which one occurs depends on the vagaries of CTX and is therefore unpredictable, ultimately making it impossible to distinguish between data and acknowledgments.

Weaver replaces the 2-slot TX-RX structure of Glossy with a 3-slot TX-RX-RX one (Figure 4.2b). This simple addition, combined with information gathered during the epoch bootstrap, decouples the TX from nodes at different hop distances, enabling each node to consistently receive 1. in the first RX slot, data flowing upwards, i.e., from nodes *farther* from the sink and to be forwarded towards it, and 2. in the second RX slot, acknowledgments flowing downwards from nodes *closer* to the sink, to be forwarded to initiators.

The effect is clearly visible in Figure 4.2b: *CTX from nodes at different hop distances no longer interfere*. During the first RX slot, a node at hop h can receive only from senders at $h+1$, nodes at $h+3$ from senders at $h+4$, and so on. Receivers then relay data concurrently in their next TX slot, providing forwarding progress. Therefore, different data flows coexists within a single flood and proceed towards the sink in an orderly fashion. If these non-overlapping flows carry packets from different initiators, collection speed increases dramatically, as shown by comparing Figure 4.1b with Figure 4.1a.

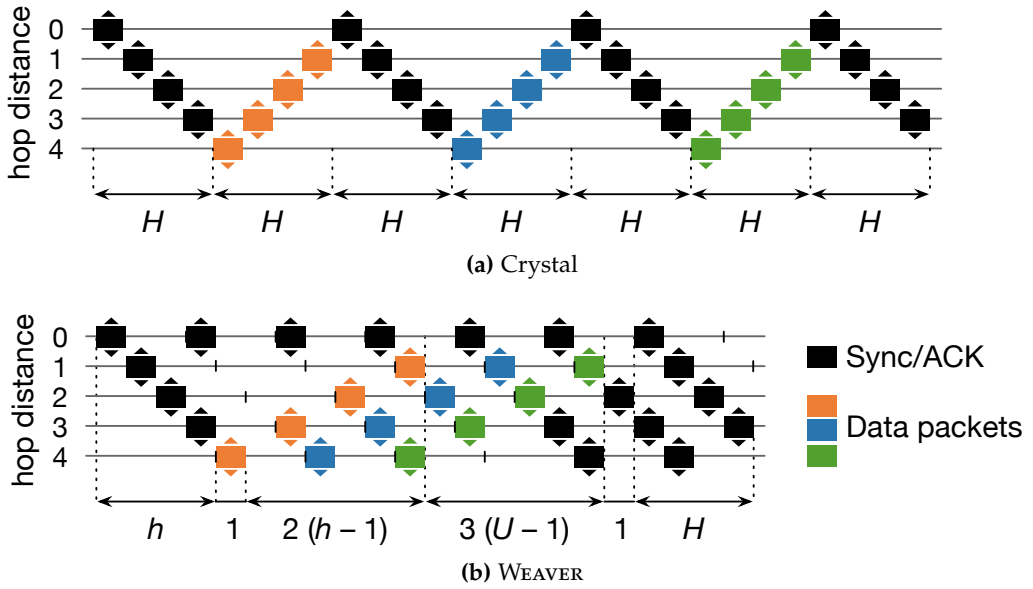
The role of (different) acknowledgments. An obstacle remains on the path to the sink: packets from same-hop initiators *compete* towards the next hop. The use of CTX ensures that one is decoded at the receiver with high probability, but what happens to the others?

Weaver solves the problem with two types of ACKs, both piggybacked on data packets whenever possible. A *local acknowledgment* (*L*-ACK) is sent by a receiver to its 1-hop neighbors to (temporarily) suppress their TX for a packet that has already made progress upwards, therefore leaving room for the propagation of other packets still behind. A *global acknowledgment* (*G*-ACK) is instead sent by the sink upon receiving a packet, and re-propagated by each node, informing the whole network that retransmissions are no longer needed for this packet, and it can be discarded.

The two ACKs are implicitly related. A node whose TX is suppressed by a *L*-ACK should eventually receive a *G*-ACK; otherwise, the packet has been lost on the way to the sink, and its TX should be resumed. The crucial question then becomes: How long should the node wait before resuming TX? The answer comes, again, from the topology knowledge accrued during the epoch bootstrap that, by informing the node of its hop distance from the sink, enables an accurate estimation of the number of slots expected to elapse between a *L*-ACK and the corresponding *G*-ACK for the same packet.

4.2.3 Is It Worth? An Analytical Model

A full understanding of Weaver entails several details (§4.3) whose treatment we postpone to first offer evidence that our strategy achieves significant improvements w.r.t. the state of the art.



(a) Crystal

(b) WEAVER

Figure 4.3: Latency models for Crystal and WEAVER with $U = 3$ initiators (orange, blue, and green) all at maximum hop distance H. Termination phase is not shown.

We achieve this goal with simplified models for Crystal and WEAVER, where we assume that 1. all data packets originate from U initiators placed at the same hop distance h, whose value $h = H$ is the worst-case maximum distance from the sink 2. packet collisions never occur, i.e., one of the packets concurrently transmitted is *always* received, and 3. we do not consider the overhead induced by protocol termination, present in both protocols.

Our models compute the number of slots required to disseminate U data packets under these assumptions, offering a proxy for latency and energy consumption. This allows us to *directly* compare Crystal and WEAVER in an abstract setting, eliciting their *intrinsic* differences, independent of the PHY layer or other system factors.

Assuming the most energy-efficient (but least reliable) configuration with a single Glossy retransmission ($N = 1$), Crystal requires

$$L_C = H(2U + 1) \tag{4.1}$$

CTX slots to deliver and acknowledge all U packets. H slots are required for the initial, synchronization phase (S), followed by one TA pair with $2H$ slots for each of the U initiators.

This is exemplified in Figure 4.3, which also shows how WEAVER significantly increases the parallelism of the U data flows and their ACKs. In the worst-case scenario we consider, the nodes $h = H$ hops away from the sink must wait h slots before they can TX data, to first receive the epoch bootstrap packet from the sink. On the other hand, differently from Crystal, TX can begin immediately after, as in WEAVER CTX occur free from the many constraints of Glossy floods. The first packet reaches the sink after $1 + 2(h - 1)$ slots, as it takes 2 slots to relay a packet one hop upwards. The remaining $U - 1$ packets reach the sink once every 3 slots, completing after $3(U - 1)$

slots. Finally, the network-wide G-ACKs triggered by the sink upon receipt of each packet account for $1 + H$ slots each, yielding

$$L_W = 3(h + U - 1) + H \tag{4.2}$$

as the total number of slots utilized by Weaver.

Figure 4.4 compares the protocol performance based on our simplified models, for several values of network diameter H and initiators U. Crystal is more efficient in the (degenerate) case of 1-hop networks, as it uses a 2-slot schedule instead of the 3-slot one used by Weaver and, for the same reason, latency is marginally better with a single initiator ($U = 1$) at $h = H$. However, in a multi-hop network, the number of slots required by Crystal is directly proportional to the network diameter H. This is not the case for Weaver, which is also faster and more scalable as U increases due to its ability to parallelize flows, up to ∼2x faster for a 4-hop diameter and ∼4x for a 7-hop one with $U = 30$ (Figure 4.4).

Although the magnitude of the performance gap between the protocols is evident, and sufficient to confirm the validity of our design choices, there are obviously several aspects that are not captured by our simplistic models. Specifically, they do not cater for system and environmental factors affecting reliability and, in turn, latency and energy consumption. These can be ascertained only with real-world experiments, which we present in §4.5 after further detailing our protocol and its implementation.

4.3 Protocol Details

We now complete the description of the Weaver protocol with additional, important details.

Epoch bootstrap. The bootstrap packet sent by the sink and re-propagated throughout the network at the beginning of each epoch is key to provide nodes with a common time reference and topology information. The bootstrap packet transmitted by any node includes the number of hops from to the sink. The sink transmits with hop count 0. Its neighbors increase the hop count by 1 before re-propagating, and so on.

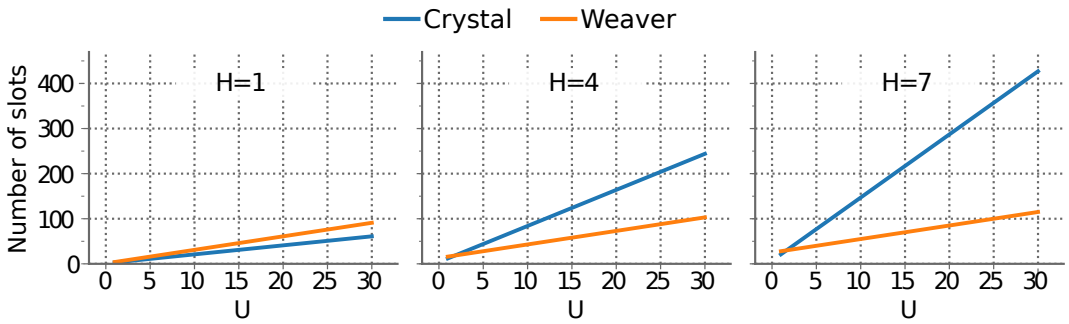


Figure 4.4: Estimated number of slots required to deliver U data packets in a network of H hops.

Therefore, after the epoch bootstrap, every node knows its distance from the sink in terms of hops, which is used to align the TX-RX-RX slot structure as in Figure 4.1b. Initiators can *immediately* transmit data inside the re-propagated bootstrap packet; unlike Glossy-based systems, there is no need to separate data collection from the initial synchronization.

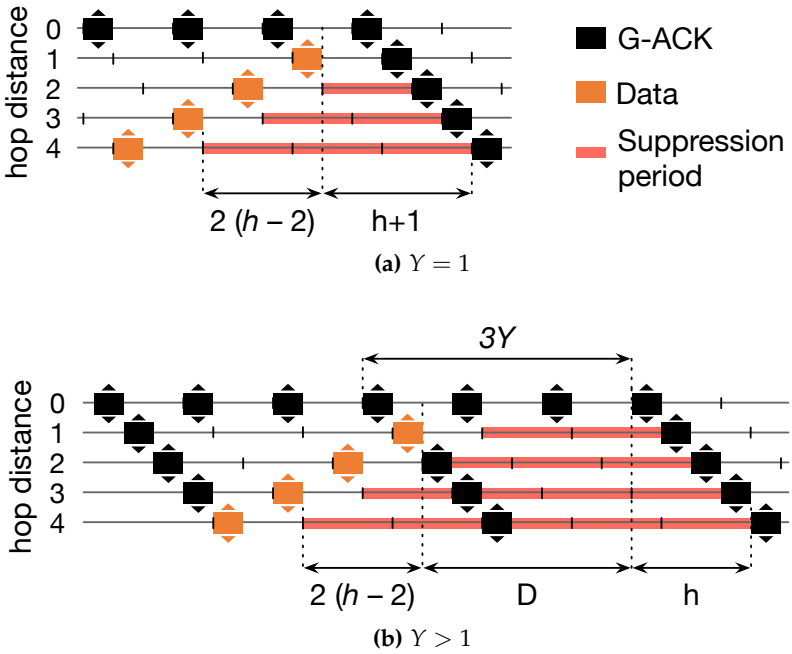
In theory, one bootstrap network-wide flood is enough; in practice, it may not reach all nodes due to collisions. When this happens, functionality is impaired; nodes that missed the bootstrap packet are unaware of their hop distance from the sink and do not know how to realign their schedule, preventing reliable operation. This iswith respect less of a problem with several initiators, as the hop distance included in all packets gives nodes that missed the bootstrap multiple chances to realign; however, it is crucial with few initiators.

The nodes that missed the bootstrap packet can reuse the information learned during the previous epoch. Often, this information is unchanged and can be refreshed in the next epoch, if collisions are rare. However, this may be not enough to accommodate the vagaries of wireless communication, or scenarios encompassing mobility. A simple and more reliable solution is to retransmit the bootstrap packet a pre-defined number B of times. We analyze the impact of the value of B on reliability and energy efficiency in §4.5.2.

Local acknowledgments (*L*-ACKs). Upon packet TX, nodes embed the initiator ID of their last heard packet in the Weaver header. When received at another node in the second RX slot, the one devoted to communication from upstream nodes, this ID indicates that the corresponding packet has already made progress towards the sink. Therefore, the original data packet doubles as a *L*-ACK for previous packets at downward nodes waiting to TX the same old data; these nodes can suppress this unnecessary packet TX and replace it with one for a new packet, if any, speeding up propagation of the latter and avoiding unnecessary contention due to the former. Nodes without new data listen during TX slots to hear same-hop neighbors and help them deliver their packets.

Global acknowledgments (*G*-ACKs). The *G*-ACKs sent by the sink contain a bitmap with one bit for each node in the system, signaling whether a packet from the corresponding node has been delivered at the sink during the epoch. *G*-ACKs are interwoven with data collection; they are received in the second RX slot from upward nodes and subsequently propagated downwards in the next TX slot. As with *L*-ACKs, nodes piggyback the *G*-ACK bitmap on data packets, if any, as part of the mandatory Weaver header. Nodes without data to send re-propagate the *G*-ACK as a no-payload packet only if it contains new bits, to reduce contention.

Linking the two ACKs: Suppression period L. The reception of a *L*-ACKs by a downward node suppresses the TX of the corresponding packet. Nevertheless, the latter must eventually be acknowledged by the sink via a *G*-ACK; if this does not happen, the packet never reached the sink and dissemination must be resumed. This combination of acknowledgments exploits spatial diversity and, as we verified experimentally, is more reliable than triggering retransmissions only upon a missed



(a) $Y = 1$

(b) $Y > 1$

Figure 4.5: Determining the suppression period L.

L-ACK, which is prone to packets remaining stuck in areas with weak links towards parents.

The availability of topology information passively gathered during epoch bootstrap enables an accurate estimation of the number L of slots expected to elapse between the RX of an L-ACK for a packet and the G-ACK (Figure 4.5a). Indeed, for a G-ACK to be sent, the corresponding packet must first be received; for an initiator at h hops from the sink, this requires $2(h-2)$ slots after RX of the L-ACK. At the sink, because of the 3-slot scheme, an additional slot elapses between packet reception, in the first RX slot, and the next TX. Finally, in the latter slot the sink disseminates the G-ACK, which travels back to the initiator, requiring additional h slots. Therefore, upon receiving a L-ACK, a node computes a *suppression period*

$$L = 2(h-2) + h + 1. \tag{4.3}$$

If the suppressed packet is not acknowledged by the sink after L slots, the node resumes its transmission.

Both types of ACK are not immune from packet loss due to collisions, potentially causing wasteful TX that nonetheless rarely affect reliability. A missed L-ACK prevents packet suppression. As for G-ACKs, when a node receives a packet already known to be acknowledged by the sink, it resumes the piggybacking of the G-ACK bitmap, to cater for nodes that may have missed it.

In summary, 1. L-ACKs avoid wasteful retransmissions of packets, hampering the progress of others 2. G-ACKs achieve the same goal definitely and globally 3. together, as determined by L, they avoid that a packet stuck in a "dead end" area is lost and forgotten.

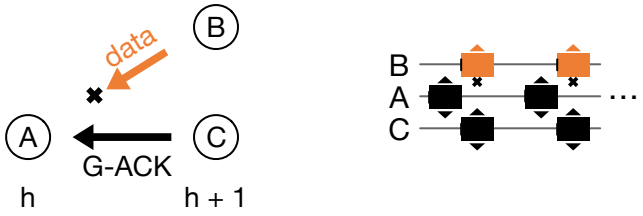


Figure 4.6: Example scenario in which *G*-ACKs block the propagation of data packets.

Tuning Weaver: Batching *G*-ACKs. Acknowledgments bring several benefits, but also cause their share of problems.

Consider the example in Figure 4.6. Nodes B and C are at the same hop distance from the sink; their schedule is aligned and their packets, whether containing data or ACKs, compete in the same TX slot. A problem arises if one node enjoys better link quality than the other(s) towards their upward node A, e.g., because C is physically closer to A. In this case, the *G*-ACKs re-broadcast by C are likely to suppress the data TX from B at A, and do so repeatedly due to the periodicity of schedules, until *G*-ACK propagation ends.

To counter situations like this, which do occur in practice, we introduce a *batching period* Y for *G*-ACKs at nodes other than the sink. Instead of immediately re-propagating a *G*-ACK upon RX, nodes send a cumulative *G*-ACK once every Y executions of the 3-slot TX-RX-RX pattern, i.e., every $3Y$ slots (Figure 4.5b). When the TX of a *G*-ACK occurs on a node, its bitmap is up-to-date w.r.t. *G*-ACKs received during this period. Therefore, the same information is delivered to the network, but $3(Y-1)$ slots are now free from data/ACK interference like the one in Figure 4.6. However, if data packets are transmitted in the meanwhile, the *G*-ACK bitmap is still piggybacked on them, as this does not cause problems.

As this technique changes the mechanics of *G*-ACK propagation, we revisit the earlier definition (4.3) of the suppression period L as

$$L = 2(h-2) + h + D \tag{4.4}$$

accounting for the additional D slots (Figure 4.5b) introduced by *G*-ACKs batching. As *G*-ACKs are issued with a predefined, globally-known period, nodes can autonomously determine the value of D upon receiving a *L*-ACK for a packet and before its next *G*-ACK.

Termination. WEAVER targets fast, reliable, and energy-efficient data collection. This entails quickly turning off the network upon detecting absence of data packets while ensuring that key nodes do not leave before all packets have been delivered to the sink.

Every node terminates and enters low-power mode (sleep) after accumulating in a termination counter T a given number of inactive slots in which no new data is received. RX errors are considered an attempt from neighbors to transmit new information, and reset T. Similarly, the RX of *G*-ACKs informs a node that the sink is still

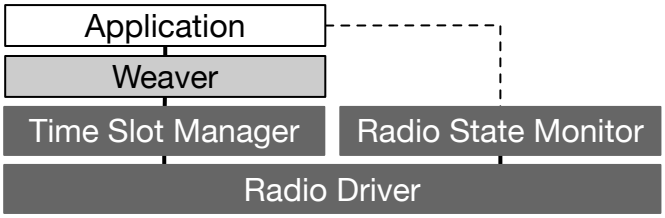


Figure 4.7: System architecture.

active and whether it is aware of the data the node already transmitted; if it is not, the node postpones its termination.

The value of T depends on the protocol phase. During the epoch bootstrap, $T = 3H + 3B$, where B is the number of bootstrap packets sent and H the maximum hop distance of nodes from the sink. Indeed, 1. H slots are required for the bootstrap packet to reach the farthest possible initiator and enable its packet TX; 2. $2H$ slots are required, due to the 3-slot scheme, for a packet from this worst-case initiator to reach the sink; however, 3. this packet competes with the B bootstrap packets rebroadcast by neighbors in consecutive TX slots; therefore, in the worst case where these are always received upstream instead of the packet, additional $3B$ slots must elapse.

If the sink does not receive any data packet T slots after sending the bootstrap packet, it enters sleep. Otherwise, an alternate counter is defined and reset every time a data packet is received. The sink waits $T = 3H + 3$ slots at the end of every G-ACK batching period. Similar to the above, 1. H slots are required for the G-ACK to propagate downwards and enable the TX of a new packet, 2. $2H$ slots are required to collect it at the sink, plus 3. 3 slots to account for the worst case where the re-propagation of the G-ACK by a same-distance neighbor blocks the data TX. The suppression period L after a L-ACKs was defined precisely to allow transmissions to resume timely for packets that did not receive the G-ACK, giving them another chance to reach the sink before termination.

Once the sink decides to terminate, it floods a special packet to shutdown the entire network before entering sleep; there is no point in keeping nodes awake if the sink is not. However, nodes are also capable of entering sleep autonomously, as they maintain the same termination counter T as the sink; this serves as a fallback ensuring node termination when the shutdown packet is lost.

4.4 A Modular Implementation

Weaver relies on the ability to individually manage CTX. This, however, involves low-level radio programming, time slot management and synchronization, i.e., tedious and repetitive work that complicates and distracts from the high-level protocol logic and, worse, must be largely modified when the latter changes.

To simplify our iterative development, and enable other researchers to build their own protocols atop fine-grained CTX (§4.6), we designed our prototype to sharply separate these two layers. We implement the low-level functionality necessary to

CTX in generic and reusable way, available to protocol designers via a simple yet expressive API (§4.4.1): the Time Slot Manager (TSM). The actual Weaver protocol is implemented as a thin veneer atop it, easily replaceable and modifiable. The architecture of our prototype (Figure 4.7), implemented on Contiki [72], is completed by an optional module enabling accurate estimation of energy consumption (§4.4.2).

4.4.1 Time Slot Manager: A Flexible CTX Engine

Our goal is to avoid complexity when implementing simple things while giving fine-grained control to the higher layer, when needed.

Enabling factors. The decoupling of protocol logic is enabled by new capabilities of modern radio chips, allowing access to internal high-precision timers for timestamping radio RX events and scheduling TX/RX operations at specified times.

Without these capabilities, meeting the strict timing requirements of concurrent transmissions forced protocols to trigger the next action right within the handler of the previous radio event and keep the duration of event processing constant for all nodes, packets, and protocol states to guarantee that all nodes trigger the next operation at the same time. This approach limited code branching and therefore the complexity of the higher-level protocol logic. Instead, if the next operation is scheduled directly via the internal timer of the radio, autonomously from the MCU, the protocol logic can become more rich and dynamic, expanding through levels of abstraction and indirection. The only requirement is that event processing finishes within the predefined deadline, leaving enough time for the radio to initialize and perform the next operation at the scheduled time.

Basic principles. We observe that all concurrent transmissions systems (§4.1) share the same time structure. They organise communication in rounds (epochs), placing the radio to sleep between them. Each round consists of multiple fixed-duration time slots associated with a TX or RX operation (sometimes neither) and related data processing; the number of slots per round may vary. The transmitted messages contain the current slot index within the round, enabling receivers to establish the round reference time (i.e., its beginning) from the start-of-frame delimiter (SFD) timestamp of the received packet.

We delegate to TSM all this common bookkeeping related to synchronization, i.e., computing the round reference time, executing radio operations at the right times, and updating the synchronization information in the header of TX packets to allow reference time to propagate over multiple hops. Instead, we leave it to the higher layer (e.g., Weaver) to decide what action (TX, RX, or none) to perform in each slot, the data payload for each TX slot, and when to stop the current round and enter sleep until the next begins.

A node joins the network via the `TSM_SCAN` operation, instructing TSM to listen to the channel until a packet is received. When this happens, TSM automatically synchronizes with the network and starts the slotted operation. Optionally, the higher layer can instruct TSM to adjust its reference time upon any successful RX; it is wise to do so periodically, to counter clock drift. Unlike Glossy, TSM is agnostic of node

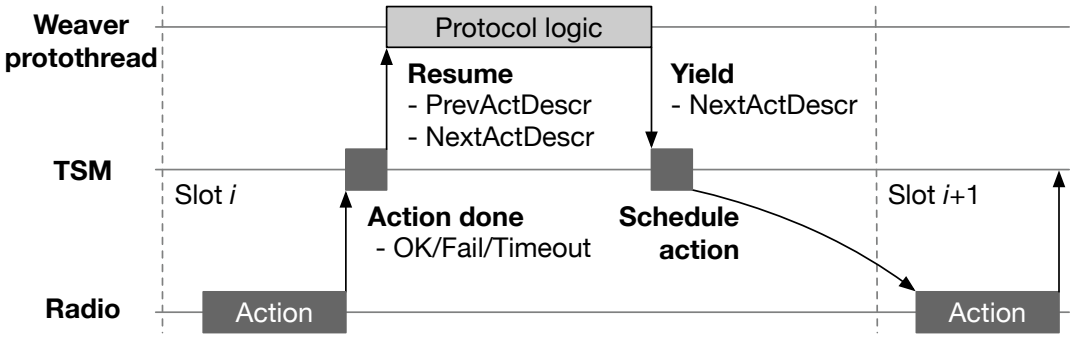


Figure 4.8: Flow of control.

```
while(1) {
  TSM_SCAN(buf); // scan until RX
  if (prevActDescr.status == SUCCESS) {
    // forward the packet in the next slot
    TSM_TX(buf, prevActDescr.data_len);
    TSM_RESET(ROUND_PERIOD); // skip to the next round
}}
```

Figure 4.9: Glossy forwarder logic ($N = 1$) atop TSM.

roles, leaving it to the higher layer to determine which node(s) provide the authoritative time reference, but provides all the necessary time calculations, adjustments, and scheduling.

API and control flow. A protocol built atop TSM begins with a `TSM_START` call providing the desired slot duration and a pointer to the slot handler function. The control flow is then driven by TSM, which automatically calls the latter function before every slot (Figure 4.8), passing as a parameter a special read-only structure describing the operation performed in the *previous* slot, if any, including the code of the action performed, its status (success or error code), the RX payload data and size (if any), and the slot index.

Another structure describing the *next* slot action is passed by reference, to be filled by the slot handler function. TSM pre-configures most fields with default values based on settings and context; only few must be set by the protocol. This next-slot structure includes the action to perform (`SCAN,RX,TX,RESTART,STOP`), a pointer to the TX payload or RX buffer, and other fields described later. After the slot handler function ends, control returns to TSM which uses the values set in this structure to schedule the next action.

In principle, a conventional function can be used as slot handler. However, TSM was designed to take full advantage of Contiki protothreads by providing convenience calls (C macros) that combine the configuration of the next action with protothread interaction, effectively mimicking a conventional blocking function (Figure 4.8). These convenience calls yield control to the system, letting the MCU perform other tasks or go to a low-power mode while waiting for the requested action

to complete; when this occurs, the protothread is resumed from the point where it requested the TSM action. This enables the description of a complex protocol logic as a sequential program with branching and loops, arguably more natural than the cumbersome event-driven style necessary with classic callbacks.

Figure 4.9 shows a naïve yet working implementation of the GLOSSY forwarder logic with $N = 1$, written atop TSM in only 6 lines of code. A full-blown Glossy re-implementation is outside the scope of this chapter; the code is meant to illustrate the simplicity induced by TSM, fully exploited in our WEAVER prototype. Each loop iteration is a GLOSSY round. The forwarder, i.e., any node other than the initiator, begins each round by scanning the channel for incoming packets. When one arrives, TSM automatically uses it to resume the protothread and begin slotted operation; in case of successful RX the node retransmits the exact same payload in the next slot via the `TSM_TX` call. TSM advances the slot index automatically and updates the TX packet header accordingly. After packet TX is finished, the `TSM_RESET` call instructs TSM to finalize the current round and sleep for the rest of the specified `ROUND_PERIOD`.

This example shows how TSM keeps simple things simple, by hiding all operations related to timing and slot scheduling, and allowing the higher layer to concentrate on the protocol logic. On the other hand, the fact that WEAVER, a significantly more complex protocol, was implemented atop TSM confirms that the abstractions in TSM are not only simple but also expressive.

Delayed TX. As reported in Chapter 2, concurrent transmissions perform significantly better in UWB if they are slightly de-synchronized, unlike in narrowband. TSM caters for this by allowing the definition of a small TX delay (ns to µs) on a per-slot basis. This requires adding the value of the delay used to the nominal slot reference in the TSM header, enabling receiving devices to compensate the delay when computing the round time reference. WEAVER exploits this feature by inserting a random delay before all TX, specified in the corresponding field of the next-action structure. Delay values are reported in §4.5.1.

RX timeouts and energy savings. Idle listening (preamble hunt) is the most energy-consuming operation of the DW1000. Therefore, we minimize the time the radio listens to the channel in RX slots. Since we expect nodes to be synchronized, the radio can begin listening shortly before we expect the frame preamble to arrive, and stop shortly after if none is received. We achieve this by setting a preamble detection timeout equal to the sum of 1. the initial guard time 2. the maximum TX delay senders could use, and 3. half the preamble duration. If no preamble symbols are detected before timeout, the radio switches to idle mode automatically and triggers an event

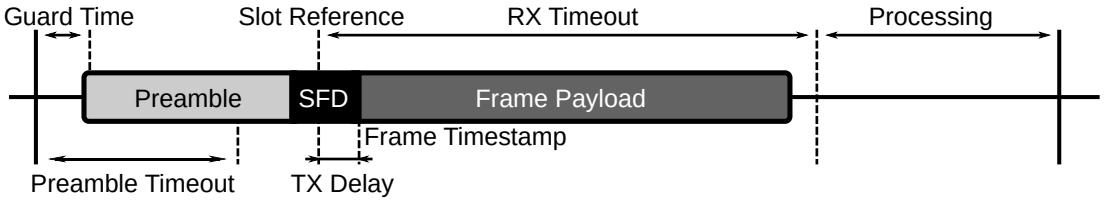


Figure 4.10: TSM slot structure.

to TSM. Moreover, we set a frame RX timeout to guarantee that any RX operation leaves enough time within the slot for the protocol layer to run its logic and prepare for the next slot. Figure 4.10 shows the resulting structure of a RX slot. In case a timeout triggers, TSM reports a failed RX action to the higher layer.

4.4.2 Monitoring Energy Consumption

An accurate estimation of energy consumption is crucial to validate the performance of our prototype. Systems built atop Contiki for IEEE 802.15.4 narrowband can rely on the well-known Energest [96] component. Unfortunately, no equivalent exists for UWB.

Therefore, we designed our own component to estimate the energy spent during radio operations. Our Radio State Monitor module (Figure 4.7) brings the core concepts of Energest to the more complex state machine of the DW1000 radio. This entails supporting several key features not present in Energest, e.g., delayed operations and timeouts, and using the precise timer of the radio. As in Energest, our module maintains several counters aggregating the overall time spent by the radio in the various states. However, differently from it, our module tracks separately different portions of the frame RX and TX for more accurate estimation of the energy spent, as these consume very different amounts of energy on the DW1000. Finally, the current drawn in idle mode is also accounted for.

Overall, the Radio State Monitor is a valuable contribution per se, not tied to CTX, that can be exploited by other researchers working on UWB at large to assess the energy consumption of their systems.

4.5 Evaluation

We evaluate Weaver in an UWB testbed at our premises, considering two topologies with different characteristics. We first provide an in-depth analysis of parameters B and Y, which control the reliability of the epoch bootstrap and the periodicity of G-ACK dissemination, and analyse their impact on performance. We then compare Weaver to the UWB implementation of Crystal [2] in the same conditions. For both protocols we report 1. the packet delivery rate (*PDR*) at the sink, 2. the per-epoch estimated energy consumption of non-sink nodes, and 3. the latency, defined as the time between the beginning of an epoch and the delivery of the *last* data packet at the sink. Finally, we experiment with mobile nodes to assess whether Weaver is suitable for use in dynamic topologies.

4.5.1 Experimental Setup

Hardware and testbed. We report experiments from a 36-node testbed installed on the ceiling above the corridors of an office building, over a 84×33 m^2 area (Figure 4.11). Each node includes a Raspberry Pi, a JTAG programmer, and a DecaWave EVB1000 board equipped with a DW1000 UWB radio and a STM32F105 ARM Cortex M3 MCU. A dedicated Ethernet infrastructure enables automated and remote control of experiments and collection of logs.

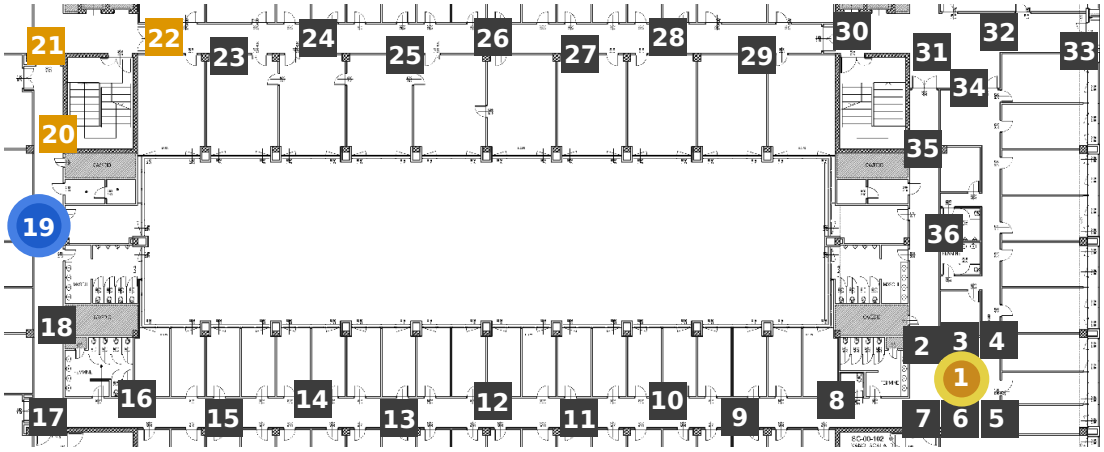


Figure 4.11: Testbed spanning $84 \times 33m^2$. In Floor, node 1 is the sink. Linear excludes node 20–22; node 19 is the sink.

Network topologies. We consider two topologies, called Floor and Linear, with different characteristics. In Floor, node 1 is designated as the sink, all nodes are active, and the network spans 3 hops. Data can flow along two paths—clockwise and counter-clockwise—providing spatial diversity. The sink is deployed in a dense area where 10 neighbors have near-perfect link quality towards it, and most (nodes 2–7) are placed at similar distances from it. As reported in Chapter 2, a similar scenario can be challenging for CTX-based protocols, and therefore intriguing to analyze, since multiple signals with similar strengths and timing are likely to reach the sink, increasing collisions especially with several and different packets.

In Linear, node 19 acts as the sink and nodes 20–22 (top left corner) are disabled, preventing communication between the sink and node 23. This 1. increases the maximum hop distance to 6 hops, and 2. forces all data flows to proceed along a single path, significantly reducing spatial diversity. Moreover, node 18 cannot communicate with any of nodes 8–16 on the bottom corridor; therefore, node 17 is the *only* connection between the sink and the remaining (large) part of the network. The absence of receiver redundancy, known to be detrimental for concurrent transmissions-based protocols, makes this topology particularly challenging, yet realistic in indoor environments.

Radio configuration. We use channel 4 with 64 MHz pulse repetition frequency (*PRF*). To minimize energy consumption, we choose the highest 6.8 Mbps data rate on the DW1000 and the shortest ∼64 µs preamble, in line with [2]. We set the TX power to the maximum recommended [63] for our channel and *PRF*. We exploit TSM to randomly delay all transmissions by up to 1 µs (i.e., roughly the duration of one preamble symbol) as we verified that this small de-synchronization is sufficient to significantly reduce the chance of collision.

Packet size and slot duration. Long packets are known to increase the chance of collision when transmitted concurrently [56, 2]. To assess how this impacts reliability and energy consumption, we perform experiments with both short (2B) and long (100B) payloads. We set the duration of Weaver slots to 813 µs, enough to

Table 4.1: Occurrence of failed bootstrap for any node and average energy consumption vs. number B of bootstrap packet retransmissions. Data acquired over $10,000$ epochs in two topologies with no initiator ($U = 0$).

Topology	% of failed bootstraps			Energy (mJ)		
	B=1	B=2	B=3	B=1	B=2	B=3
Floor	0.015	0.0003	0	2.49	3.00	3.35
Linear	0.0016	0.0006	0	3.19	3.46	3.73

accommodate the maximum IEEE 802.15.4 frame length.

4.5.2 Dissecting Weaver

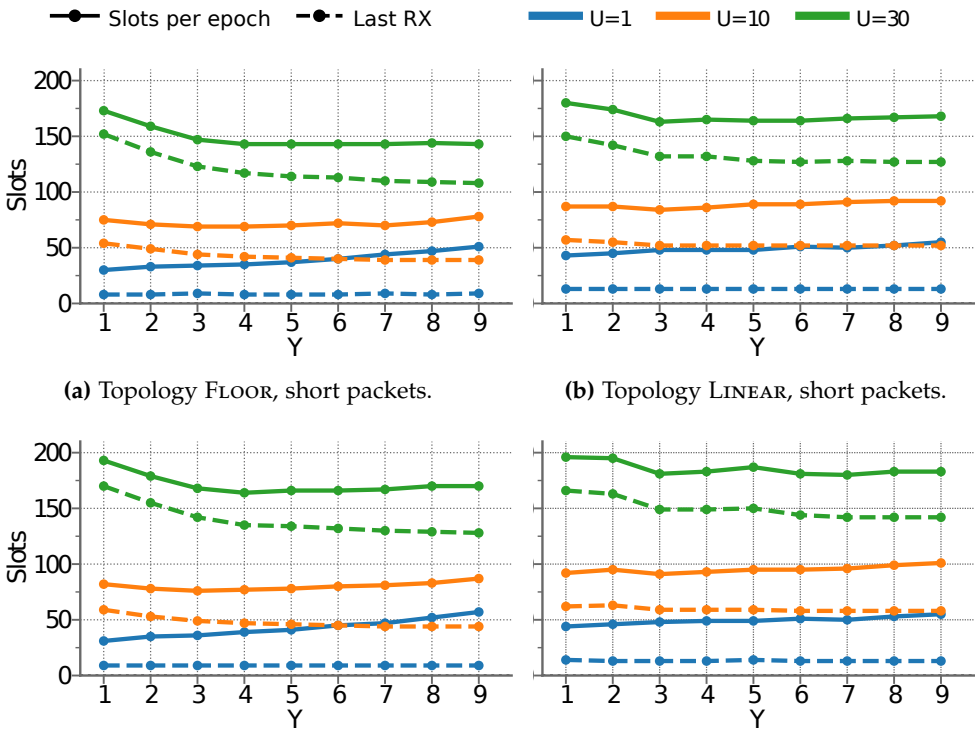
We study the impact of parameters B and Y on the performance of Weaver. The former impacts the reliability of epoch bootstrap, while the latter controls the trade-off between latency and energy consumption depending on the expected traffic patterns.

Reliability of epoch bootstrap. We explored $B \in \{1..3\}$. Table 4.1 reports the number of failed epoch bootstrap attempts across $10,000$ epochs, with no initiators ($U = 0$). In Floor, $B = 1$ yields 59 occurrences of a node missing the bootstrap packet (0.015%), while $B = 2$ yields only 1 occurrence (0.0003%). Linear is less prone to a failed bootstrap, with the same values of B yielding only 5 and 2 occurrences (0.0016% and 0.0006%), respectively.

The value $B = 3$ guarantees a correct bootstrap in all epochs for both topologies. However, this reliability comes at the cost of energy consumption (Table 4.1) whose increase is more evident without traffic ($U = 0$) as node termination directly depends on B (§4.3). In this case, consumption is 3.35 mJ and 3.73 mJ in Floor and Linear, a $+35\%$ and $+17\%$ increase w.r.t. $B = 1$. However, when traffic is present ($U > 0$), the influence of B is less marked as 1. the network remains awake for longer to collect all data, and 2. collection starts immediately, in parallel with bootstrap. Hereafter, we set $B = 2$, the best compromise between reliability and energy efficiency.

Impact of G-ACK batching period. The period Y used to disseminate G-ACKs upon data reaching the sink is the main knob to control Weaver, balancing timeliness in acknowledging packets via G-ACKs with their interference with data (§4.3). We analyze the impact of Y on the duration of the flood (Figure 4.12). For each combination of topology, packet size, $Y \in \{1..9\}$, and $U \in \{1, 10, 30\}$ the results are obtained by aggregating 1000 epochs.

The impact of Y on termination, while not high in relative terms, varies in function of the amount of traffic (Figure 4.12). In both topologies, Weaver shows a similar response to the increase of Y, although the trend is more evident in Floor. Similarly, packet size does not have a substantial impact, with longer packets causing only a slight increase in latency. With sparse traffic, increasing Y does not yield benefits as G-ACKs rarely interfere with data floods, making the duration of the collection phase independent from Y. Thus, $Y = 1$ is the fastest and the most energy-efficient



(a) Topology Floor, short packets.

(b) Topology Linear, short packets.

(c) Topology Floor, long packets.

(d) Topology Linear, long packets.

Figure 4.12: G-ACK batching period Y vs. number of slots required for termination and last packet collected at sink.

solution, as it minimizes the time a node waits in between the last packet RX and termination (§4.3). However, as the number U of initiators increases, a small value of Y becomes detrimental; with $Y = 1$, each packet reaching the sink triggers a new G-ACK, disseminated network-wide. This increases contention and the risk of interference between data and G-ACKs, slowing down the collection process. By increasing Y and therefore reducing the number of G-ACKs, we increase the chance to collect multiple packets in between two consecutive G-ACKs floods. The impact on the flood duration is clearly visible for $U = 30$. For instance, in Floor and with short packets a flood requires 174 slots to terminate with $Y = 1$, and only 143 with $Y = 4$ (-18%). On the other hand, increasing Y further does not pay off, as it forces the system to remain active for several slots after the last packet collected; this is very costly with sparse traffic and brings little to no improvement with a denser one.

The best choice of Y ultimately depends on the behavior of initiators. If the traffic profile is known beforehand, users can tune the value of Y to further reduce the latency and energy consumption of Weaver. Otherwise, Figure 4.12 shows that the impact is relatively limited anyway. In the rest of this section, we assume the application has no a priori knowledge of traffic and set $Y = 4$ as in our case this is a good balance across all dimensions.

Finally, Weaver achieved $PDR \geq 99.9\%$ independently from the value of Y. A more thorough study of reliability is described next.

4.5.3 Weaver vs. Crystal

We compare against Crystal [60], a state-of-the-art data collection protocol, using its publicly-available implementation for UWB [2].

Protocol configurations. We configure Weaver with $B = 2$ bootstrap packet retransmissions and a G-ACK batching period $Y = 4$, informed by our analysis in §4.5.2. Configuring Crystal entails tuning the underlying Glossy for every phase, by defining N and adapting the maximum phase duration W accordingly. Large values of N enhance flood reliability, by increasing the spatio-temporal redundancy of Glossy, but also increase energy consumption. In our analysis we consider $N \in \{1, 2\}$, exploring different trade-offs between reliability and energy efficiency. $N = 1$ is the most energy-savvy configuration possible, but also the most fragile. Table 4.2 reports a summary of the configurations. Following the methodology of [60], we dimension W for each phase to accommodate the maximum hop count H, plus a small slack to cope with possible flood delays due to collisions. Other Crystal parameters (e.g., number of empty TA pairs before termination) are unchanged w.r.t. [60, 2].

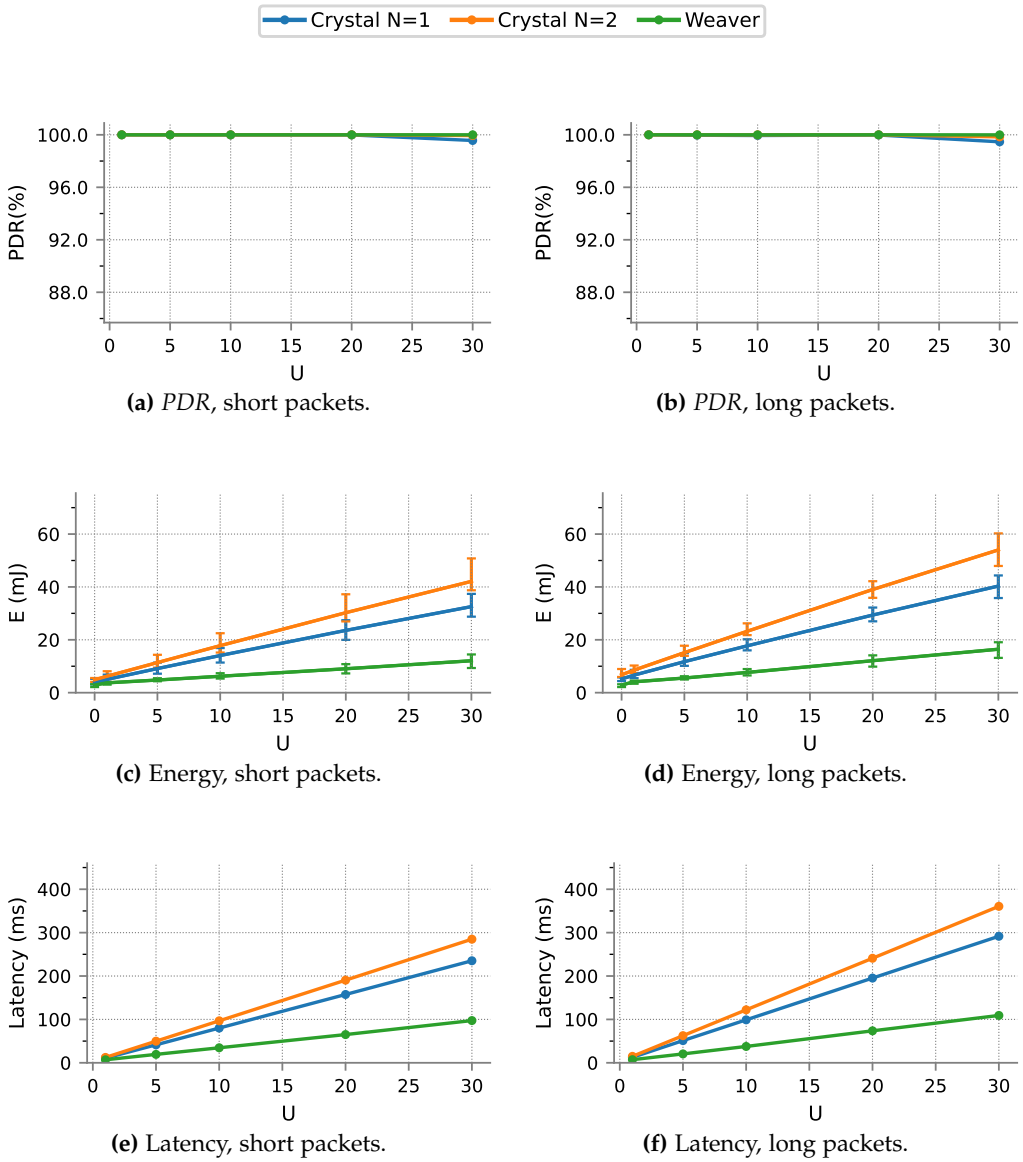
Results. For each combination of topology, number of initiators $U \in \{0, 1, 5, 10, 20, 30\}$, packet size, and protocol configuration, we collect execution traces of 5000 epochs for both protocols.

In Floor, Weaver is largely unaffected by packet size, achieving near-perfect reliability with both short and long ones (Figure 4.13a, 4.13b) even under heavy contention, with $PDR > 99.99\%$ when $U = 30$. Instead, the reliability of Crystal is significantly lower, especially with $N = 1$, and the negative impact of long packets is clearly visible as U increases, due to the higher chance of collisions.

We actually found an increased rate of collisions for long packets also in Weaver, by analyzing the RX error rate at the level of single slots. For instance, with $U = 30$ each node incurs in a RX error 5.83 times per epoch with short packets and 9.07 with long ones. However, Weaver can tolerate more collisions, as the continuous flood grants each node many chances to retransmit. Further, the number of retransmissions is not fixed beforehand, as in Glossy and therefore Crystal, rather it adapts to data traffic, as nodes keep attempting to forward packets upon collisions. Moreover, thanks to the

Table 4.2: Parameters used for the two configurations of Crystal considered. T_s and T_l are the duration of the T phase optimized for a short (2B) and long (100B) packet, respectively.

Topology	Retransmissions N (S,T,A)	Phase duration W (ms)			
		S	A	T_s	T_l
Floor	1	2.7	2.8	2.8	4.5
	2	3.6	3.7	3.6	6.1
Linear	1	4.0	4.1	4.0	6.0
	2	4.8	5.0	4.9	8.4



(a) *PDR*, short packets.

(b) *PDR*, long packets.

(c) Energy, short packets.

(d) Energy, long packets.

(e) Latency, short packets.

(f) Latency, long packets.

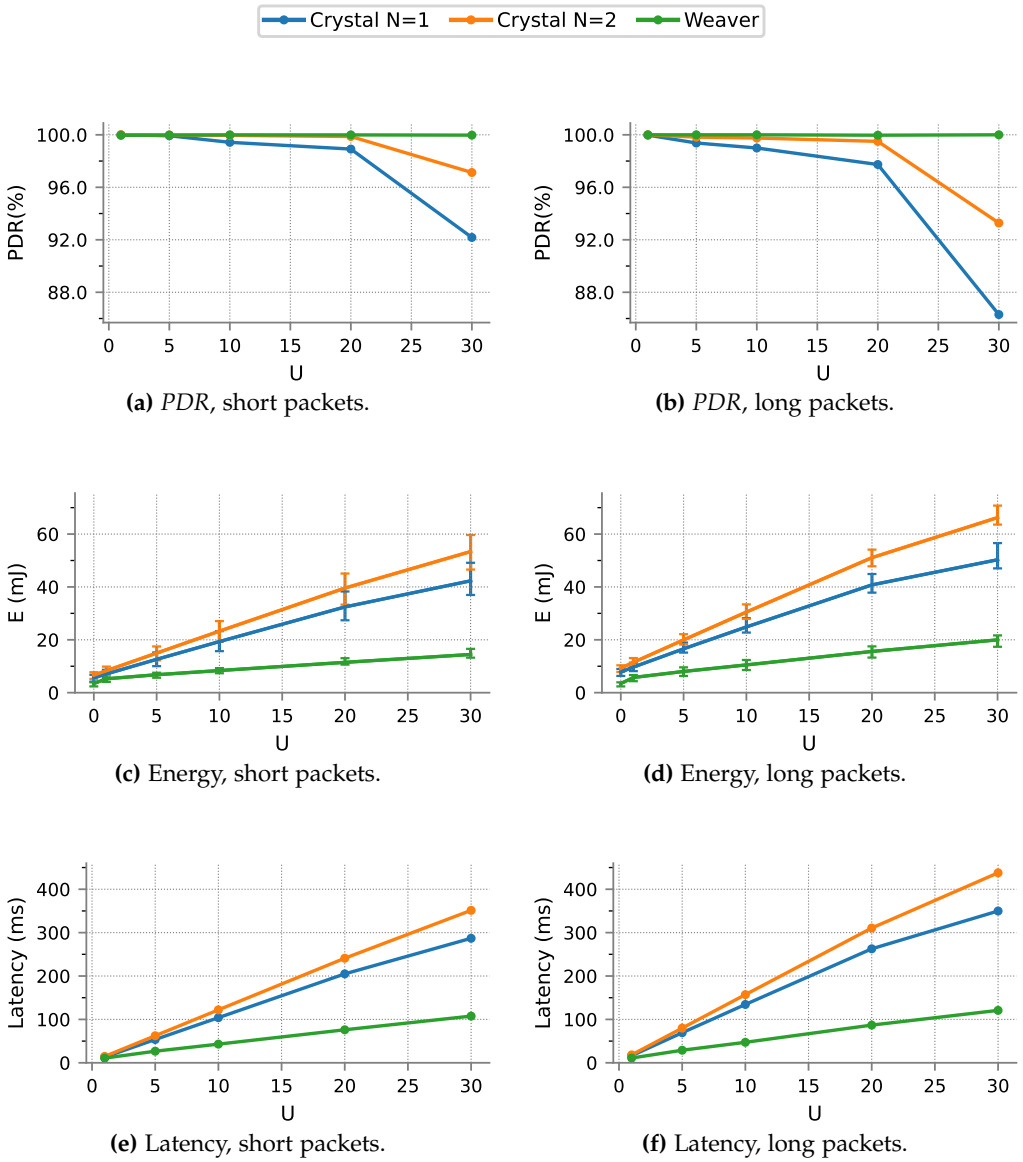
Figure 4.13: Weaver vs. Crystal in the Floor topology.

L-ACKs, Weaver can promptly suppress transmissions before the arrival of *G*-ACKs from the sink, quickly reducing contention.

High reliability often comes with extra energy consumption. This is not the case for Weaver, specifically designed to remove the inefficiencies of Glossy-based solutions. Indeed, the fast, reliable and contention-resilient operation of Weaver yields significant energy improvements w.r.t. Crystal (Figure 4.13c, 4.13d).

Without traffic ($U = 0$), Weaver consumes 40% and 57% less than Crystal with short and long packets, respectively. The benefits of fine-grained control over concurrent transmissions increase with U, as Weaver fully unleashes its ability to parallelize collection floods, *reducing energy consumption by* $\sim$70% *for* $U = 30$ *initiators regardless of packet size*.

In the Floor topology explored so far, multiple paths enable packets to reach the



(a) *PDR*, short packets.

(b) *PDR*, long packets.

(c) Energy, short packets.

(d) Energy, long packets.

(e) Latency, short packets.

(f) Latency, long packets.

Figure 4.14: Weaver vs. Crystal in the Linear topology.

sink, itself surrounded by many relays. In Linear, all data towards the sink must flow through the bottleneck of node 17; continued collisions at this node can lead to interruption of the flood and multiple packet losses. Crystal behaves poorly in these conditions; even with the more reliable $N = 2$, *PDR* decreases as U increases, down to ∼97% and ∼93% for $U = 30$ and short and long packets, respectively (Figure 4.14a, 4.14b). Weaver is affected to a much smaller extent, achieving $PDR > 99.9\%$ in all conditions.

Energy consumption increases for both protocols in Linear, due to the larger diameter. However, Weaver consumes a fraction of the energy required by Crystal (Figure 4.14c, 4.14d) similar to what observed in Floor; with $U = 0$, Weaver saves 40% and 63% with short and long packets, and ∼70% for both packet sizes with $U = 30$.

Weaver is highly reliable and energy efficient in both our topologies, despite Linear being quite challenging. The question is whether it is also faster, as predicted by our model (§4.2.3). Many definitions of latency are possible. We report the time needed to complete data collection (i.e., RX of last packet) because 1. the average latency incurred by a packet is roughly half the duration of collection, and 2. flood termination at the sink happens consistently a few slots after the RX of the last packet, making these two metrics redundant.

Our experiments confirm that Weaver is significantly faster than Crystal. With long packets and $U = 30$, Crystal receives the last packet after 361 ms in Floor, and 438 ms in Linear, while Weaver does the same in only 109 ms and 121 ms, respectively (Figure 4.13f, 4.14f). Interestingly, switching from Floor to Linear causes a 21% latency increase for Crystal, but only 11% for Weaver. Even with a single packet ($U = 1$) Weaver is faster at 7 ms and 11 ms, against the 15 ms and 18 ms of Crystal, also thanks to the ability to begin packet TX concurrently with the initial bootstrap. Weaver is faster than Crystal also with short packets, in both topologies (Figure 4.13e, 4.14e). For $U = 30$, its latency is 97 ms and 107 ms, against 285 ms and 351 ms for Crystal. For $U = 1$, Weaver incurs a latency similar to long packets, while the one of Crystal, reduced to 12 ms and 15 ms, remains higher than Weaver.

Optimizing the fixed slot duration (§4.5.1) for packet size leads to large improvements. For 2B packets, a shorter slot of 455 µs reduces latency and energy consumption respectively by 43% and 22% w.r.t. Figure 4.13–4.14, regardless of traffic and without affecting reliability.

4.5.4 Weaver and Mobility

Among the advantages of concurrent transmissions-based protocols is that they are agnostic of the underlying network; being resilient to topology changes they are suitable for scenarios with mobile nodes [59]. However, this is not entirely true for Weaver. Nodes learn their hop distance during the epoch bootstrap, and leverage this information to direct data and *G*-ACKs flows during collection; this potentially makes the protocol susceptible to topology changes.

Nevertheless, Weaver completes the collection of packets from 30 initiators over a 6-hop network in ∼100 ms (Figure 4.14e). During this time, a person walking covers ∼14 cm and a car traveling at 100 km/h covers ∼3 m. A Weaver flood is so fast that even when nodes are moving the topology inside it remains essentially static.

We ascertain whether this is true, and the applicability of Weaver to mobile scenarios, through experiments in which 3 people, each carrying a node, walk at brisk pace in the testbed area for the entire duration of the test. In these experiments, all 39 nodes of the testbed are active. As mobile nodes traverse the testbed, their links to other nodes degrade or even interrupt abruptly due to obstacles.

Table 4.3 shows *PDR*, latency, and energy consumption with a static or mobile sink. In the first scenario, node 1 is the sink, as in Floor, and all mobile nodes are initiators; in the second, one of them serves as mobile sink. The latter scenario is particularly challenging, as sink movement 1. alters the structure of the collection scheme, and

2. explores several topologies at once, including problematic ones like LINEAR. We run 2000 epochs for each $U \in \{10, 30\}$ and short packets, and observed no packet loss with $U = 10$ and $PDR > 99.9\%$ with $U = 30$, regardless of sink mobility.

Overall, the values in Table 4.3 are in line with those in §4.5.3; mobility appears to have little to no impact on performance. This confirms the resilience of WEAVER w.r.t. mobility, which we are evaluating more extensively as part of our future work.

4.6 Discussion and Outlook

We concisely elaborate on the potential impact of our work and how it could be extended and generalized by other researchers.

What did we accomplish? The evaluation we presented, along with the analytical model in §4.2.3, confirm that protocols based on fine-grained CTX rather than monolithic Glossy floods can unlock remarkable improvements over the already impressive performance achieved by the latter. The ability to weave and consolidate multiple floods into a single, coordinated one improves on latency, but also on reliability and energy consumption—i.e., all three metrics in which CTX excel. On the other hand, the very small latency also enables a novel way to exploit topology information, allowing protocol designers to treat the network as static even when it is not, as in scenarios encompassing mobility. We argue that these design principles are a contribution per se, which goes beyond the nonetheless remarkable performance of WEAVER.

What about other radios? Although we focused on UWB, we argue that our contribution is not limited to it, as neither WEAVER nor TSM rely on features specific to the PHY or radio chip we used.

The superior performance of WEAVER is intrinsically determined by its use of fine-grained CTX, as shown quantitatively in §4.2.3. Indeed, the efficient organization of multiple data flows in WEAVER builds solely on the assumption that receivers can successfully decode, with high probability, one among different packets transmitted concurrently. As mentioned, this assumption has been shown to hold for other popular PHY layers besides UWB. Therefore, we expect the principles of WEAVER, if not the exact protocol, to find direct application for these other radio technologies.

However, the extent to which our quantitative findings can be transferred to other radios is yet to be established experimentally, for which TSM provides a handy framework. We argue that it is simple to port TSM to any platform that, like DW1000,

Table 4.3: Performance of WEAVER with 3 mobile nodes.

Sink	*PDR* (%)		Latency (ms)		Energy (mJ)	
	U=10	U=30	U=10	U=30	U=10	U=30
Static	100	99.993	57.72	119.51	7.58	14.07
Mobile	100	99.95	61.79	123.58	8.22	15.24

supports timestamping and scheduling of packet TX and RX precisely enough to enable non-destructive interference of TX signals. As discussed in §3.1, the timing requirements depend on the radio technology. A short-term item on our research agenda is to port TSM and WEAVER to a modern IEEE 802.15.4 narrowband radio supporting these features, e.g., the CC2538 for which Contiki-based implementations of Glossy already exist [97], further simplifying the transfer of our results.

What about other traffic patterns? The role of TSM, however, is not limited to simplifying the transfer of our results to other platforms. On the contrary, our main motivation for its development was to sharply separate the *general* low-level mechanics of CTX from the *specific* higher-layer protocol exploiting them.

In this respect, WEAVER is only one of the possibilities, geared towards data collection. We argue that the benefits unlocked by the key insight of WEAVER, i.e., its fine-grained use of individual CTX instead of monolithic floods, can be reaped for other traffic patterns similar to what happened for Glossy, whose availability as a core communication primitive was exploited in many directions.

What about ranging and localization? Our focus on UWB opens intriguing opportunities. For instance, the work in [75] has recently shown that CTX in UWB enable concurrent distance estimation (ranging) towards multiple nodes at once, inspiring several follow-up works [98, 99, 100, 101]. The concepts in WEAVER, and the core building blocks in TSM, could therefore be exploited to rejoin the two perspectives of communication and localization enabled by UWB under a single framework efficiently enabling both.

4.7 Conclusions

CTX have been studied for about a decade, but largely within the perimeter of what enabled by the popular Glossy system. In this chapter, we show that an alternate design mindset is possible; one where the protocol designer regains control over all degrees of freedom enabled by using individual CTX as building blocks, significantly finer-grained than the monolithic one offered by Glossy. We offer analytical and experimental evidence that this alternate design paradigm brings remarkable advantages in the context of convergecast, and provide publicly-available, open-source software [73] enabling researchers to explore other ways to harvest its benefits.

Part II

Ultra-wideband Localization in Large-scale Areas

5

TALLA: Large-scale TDoA Localization with Ultra-wideband Radios

Ultra-wideband (UWB) radios have rapidly gaining popularity among localization technologies. A new generation of UWB transceivers, spearheaded by the DecaWave DW1000 [70], has redefined the tradeoffs associated to this technology with radio chips that are tiny, cheap, low-power, yet capable of accurate (<10 cm error) distance estimation.

Industry and academia seized this new opportunity, with initial efforts focused on the two-way ranging (TWR) scheme relying on the time of arrival (ToA) estimation. However, in TWR-based schemes [102], at least 2 messages are required to estimate the distance between a tag and an anchor; this must be repeated for each anchor and requires continuous neighbor discovery. Therefore, commercial and research systems increasingly rely on *time difference of arrival* (TDoA) estimation (§5.1), where a *single* message transmission from the tag suffices to compute the difference in reception time at *all* localization anchors. Therefore, TDoA is more scalable than TWR, in terms of mobile tags and/or sample update rate supported [103].

Goal: Large-scale, flexible TDoA operation. However, TDoA requires tight time synchronization of the anchors. This can be achieved with an out-of-band wired network, often already present but otherwise expensive to deploy. Alternately, in-band UWB wireless schemes exist [104, 105] that greatly increase deployment flexibility

This chapter revises our publication [4]: D. Vecchia, P. Corbalán, T. Istomin, and G. P. Picco. "TALLA: Large-scale TDoA Localization with Ultra-wideband Radios". In *Proceedings of the* 10^{th} *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. 2019. DOI: 10.1109/IPIN.2019.8911790.

and reduce cost. However, these have been applied only in small-scale settings (e.g., a room) with *all* anchors in range, where clock drift can be mitigated via one-hop wireless synchronization.

In contrast, the Talla (TDoA Localization for Large-scale Areas) system we present in this chapter 1. performs time synchronization in-band, using the wireless UWB link, and 2. is capable to *scale over large operational areas without sacrificing positioning accuracy*.

Contribution. Our design (§5.2) is based on TDMA, with time slots allocated for anchor and mobile tag transmissions. Each anchor is the potential (sub)ns-level time reference for TDoA localization; this feature enables a *continuous, multi-hop* operation of the wireless anchor infrastructure supporting localization. Further, reliance on TDMA prevents collisions and therefore improves reliability and update rate.

Anchors are allocated n slots in the TDMA frame, enabling their synchronization. Tags are allocated the remaining k slots for localization. These values are configured based on application and system requirements, e.g., catering for different update rates, anchor density and total number of tags. Distant anchors can safely share time slots, therefore n depends only on anchor density but is constant w.r.t. network size.

The evaluation of Talla is non-trivial; the verification of our claims about large-scale operation directly depends on the number of nodes and the size and geometry of the operational area they are deployed in. We evaluate our prototype in a 12-node UWB testbed in a 100×60 m^2 corridor area at our premises. This is larger than what is commonly reported [106, 99, 100]; yet, the number of nodes is relatively small, the network diameter is only 3 hops, and the geometry very challenging. For this reason, we adopt an evaluation methodology (§5.3) that exploits our prototype also to inform, via real packet traces, a simulation toolchain that faithfully reproduces the timing inaccuracies affecting TDoA positioning error (§5.4). This enables us to derive synthetic yet realistic traces for areas whose size is well beyond the one of our testbed, which we nonetheless use to validate our simulated results (§5.5). Further, this mixed simulation-testbed strategy also allows us to easily explore the parameter space, which is key to analyze the design and configuration choices germane to our approach.

The chapter ends by placing our work in the context of related ones (§5.6) before offering concluding remarks (§6.7).

5.1 TDoA Localization: Fundamentals

We describe the necessary background on TDoA localization with UWB radios, with wireless time synchronization.

Infrastructure. We assume a network infrastructure with at least $N = 4$ anchors in range, enabling 2D positioning. Each anchor reports each transmission (TX) and reception (RX) along with their corresponding timestamps to a server, where the actual localization computation takes place, through a backbone network (e.g., WiFi).

Mobile tags roaming in the covered area can be located with a single packet TX per tag.

Clock synchronization. To compare the RX timestamps for TDoA localization, the server must have a clock model per anchor to translate the *local* anchor timestamps to the same clock domain. To this end, a reference anchor periodically transmits a synchronization beacon every $T_s = 1/f_s$. This beacon is received by anchors in range, which report their RX timestamps to the server that, in turn, computes the k^{th} clock offset of each anchor i w.r.t. the reference as

$$\theta_{i,k} = (t_{i,k} - \tau_{i,ref}) - t_{ref,k} \tag{5.1}$$

where $\tau_{i,ref}$ is the known time of flight between anchor i and the reference anchor, determined based on the known positions $\mathbf{p_i}$ and $\mathbf{p}_{ref}$ and the speed of light in air c. The clock drift of anchor i w.r.t. the reference anchor can then be estimated based on the previous $k-1$ beacon TX/RX information as

$$\delta_{i,k} = \frac{\theta_{i,k} - \theta_{i,k-1}}{T_s} \tag{5.2}$$

Based on $\theta_{i,k}$ and $\delta_{i,k}$, the server can translate each local anchor timestamp t_i to the reference clock domain as $t_i - \theta_{i,k} + t_e(1 - \delta_{i,k})$, where t_e is the measured time elapsed since the last synchronization beacon RX.

Position estimation. The server can estimate the true position $\mathbf{p}$ of a mobile tag in 2D based on the RX timestamps of at least $N = 4$ anchors, including the reference. The server first translates the local timestamps to the reference clock domain and then computes the TDoA estimates $\hat{\Delta}t_{i,ref} = t_i - t_{ref}$, each representing the equation of a hyperbola

$$\Delta t_{i,ref} = \frac{\|\mathbf{p} - \mathbf{p_i}\| - \|\mathbf{p} - \mathbf{p}_{ref}\|}{c} \tag{5.3}$$

To estimate the tag position $\hat{\mathbf{p}}$ with $N - 1 \geq 3$ non-redundant TDoA estimates, the server solves a non-linear least squares problem by minimizing the squared difference between the measured TDoA estimates $\hat{\Delta}t$ and the theoretical ones Δt as

$$\hat{\mathbf{p}} = \arg\min_{\mathbf{p}} \sum_{i}^{N-1} \left(\hat{\Delta}t_{i,ref} - \Delta t_{i,ref}\right)^2 \tag{5.4}$$

5.2 Enabling TDoA over Large Areas

The wireless synchronization mechanism described requires all anchors to be in range of the reference, limiting scalability. We present our design to enable TDoA localization across large areas requiring multi-hop communication. The design is based on a time-slotted approach that follows a periodic schedule. This schedule allows each node in the network (anchor or tag) to transmit its packets without collisions based on TDMA and, at the same time, enables (sub)ns-level clock synchronization for TDoA localization at the server side, where each anchor can serve as a potential time reference. The schedule is also configurable to cater for various requirements in terms of localization rate as well as number of tags and anchors.

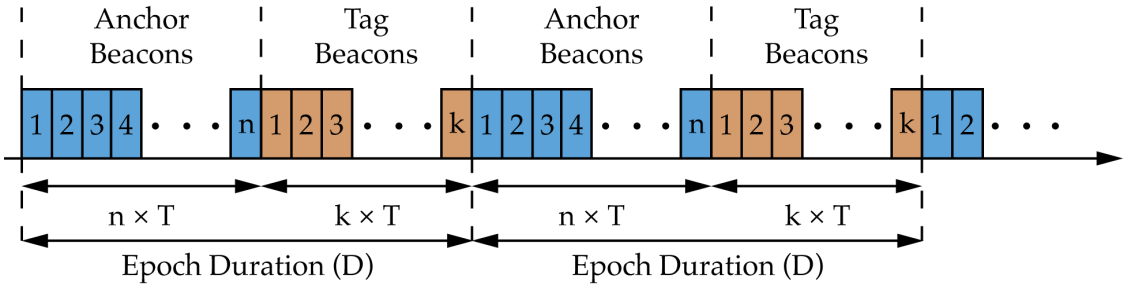


Figure 5.1: Example time-slotted schedule including n dedicated anchor slots for time synchronization and k slots for tag localization beacons.

Time-slotted schedule design. The schedule unfolds as a repetition of a pre-defined sequence of time slots, each of duration T ms, forming an *epoch* of duration D. The value of D is tied to the synchronization rate required to reliably compensate for clock drift. Within each epoch, the schedule provides each anchor and tag a dedicated time slot to transmit a beacon. Anchor transmissions serve to time-synchronize the network and enforce the schedule, while tag beacons provide the necessary TDoA measurements for positioning.

Figure 5.1 shows an example. Each of the n anchor slots is assigned to one anchor beacon, without repetitions. These beacons enable the network to build a time synchronization tree, described next, to avoid collisions among tags and/or anchors. They also enable the server to collect the information required to establish any anchor as a precise TDoA time reference. Tags listen periodically to anchor beacons to retain alignment with the schedule. Anchors are always listening except when they transmit their own synchronization beacons.

The remaining k slots are instead used by tags to transmit their localization beacon. Unlike anchor slots, the same tag can be assigned multiple slots. Tag beacons are received by anchors, which report the RX timestamps to the server, which in turn computes the tag position using the TDoA solver (§5.1).

Schedule definition. The schedule is defined at compilation time, based on the expected localization accuracy and update rate. However, it can be tailored to application needs, e.g., by assigning multiple slots to a tag requiring a higher update rate. Similarly, the n anchor slots need not be consecutive and can be spread throughout the epoch. In any case, their number n detracts from the number k of those available for tags; this can be problematic in large areas with many anchors.

Nevertheless, the problem can be easily solved by enabling non-interfering sets of anchors to re-use slots and transmit their beacons concurrently and safely, without possibility of collision. This requires a network connectivity assessment prior to operation, which can be performed by scheduling beacons from each anchor. While one anchor transmits, all others stay in reception mode. This enables each anchor to determine which other anchors are in its same interference domain, and this information can be reported to the server. The latter can use well-known graph coloring algorithms [107, 108] and communicate back to each anchor the actual full schedule

to use, therefore reducing n while retaining a reliable and robust synchronization infrastructure.

We stress the fact that the connectivity assessment for the purpose of slot re-use is *not* the same as determining the set of neighbors from which an anchor can correctly receive a packet. An anchor may never decode correctly a packet, while the transmitter of that packet may still cause interference and disrupt the reception timestamp (see the discussion on ranging in §2.4.4). We verified experimentally that non-interfering sets of anchors can be obtained considering as an interference source any anchor in reception range or causing a reception error. In other words, if the transmissions from one anchor causes an SFD timeout or a PHR uncorrectable error at another anchor, that is a sufficient condition to establish the edge between those nodes for the purpose of graph coloring. Whether or not the payload can be correctly received is not a factor if the goal to avoid interference in the estimation of the channel impulse response—where the timestamp for synchronization is extracted from.

Network-wide synchronization. To enforce the schedule and avoid collisions, the network must be time-synchronized. Anchor beacons are used to build a synchronization tree where a pre-defined node (e.g., anchor 1) serves as the global time reference. Note that to follow the schedule, µs-level synchronization is sufficient. To build the synchronization tree and avoid loops, nodes use a routing metric (e.g., hop count) to propagate the clock offsets, allowing nodes that are multiple hops away to follow the schedule. For anchors to be able to learn their time offset w.r.t. the reference, beacons must carry the node ID and metric. Based on the beacons received, nodes select an appropriate time source (e.g., the parent with the lowest hop count). Using the RX timestamp and the node ID of the parent, a node can learn its time offset and transmit its beacon in the corresponding time slot. As the schedule repeats itself, nodes can re-synchronize every epoch, reducing drastically the impact of clock drift on the schedule.

High-Precision synchronization for TDoA localization. Anchor beacons are also used to achieve (sub)ns-level time synchronization at the server, for TDoA localization. This requires each beacon TX/RX from an anchor to be precisely timestamped and sent to the server, which maintains a data structure containing the TX timestamp and sequence number of every beacon along with the ID of anchors that received it and their RX timestamps. Based on this information, the server can pick any anchor as a potential ns-level time reference to estimate the TDoA measurements required for positioning. With typical clock drifts, the synchronization frequency required to obtain accurate TDoA measurements must be $f_s \geq 1$ Hz. This in turn constrains the epoch duration, which should be kept short enough ($D \leq 1$ s) to reduce the positioning error.

Anchor selection and position estimation. The solver accuracy is closely related to the choice of anchors involved. A first issue is selecting the appropriate TDoA reference among the many available in the large-scale areas targeted by Talla. A tag beacon is received by many anchors, possibly synchronized with different candidate references; however, the anchors used for localization must be synchronized together. Our solver selects as reference the one that synchronized the highest number of an-

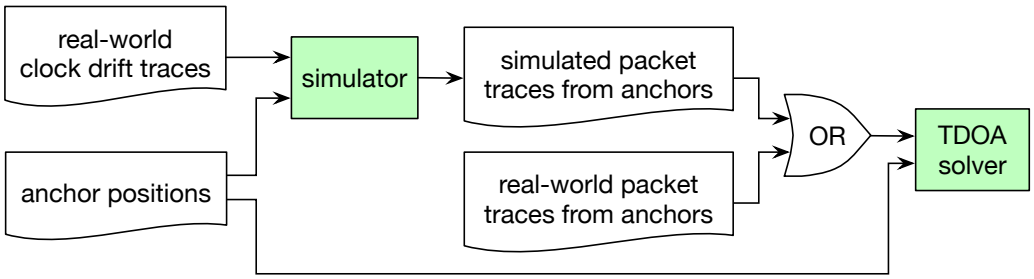


Figure 5.2: Prototypes and evaluation toolchain.

chors (among those that received the tag beacon) within the last epoch; in case of a tie, the anchor synchronized more recently is chosen. A second issue is validating the solver output. If the tag position is estimated at a distance from anchors greater than communication range, it is invalid. Localization is repeated with a subset of anchors, potentially exploring all combinations until a valid result is obtained.

5.3 Evaluation Methodology

The specific focus of TALLA demands that its performance is evaluated on *large-scale areas* containing many anchors, e.g., tens if not hundreds. Unfortunately, publicly-accessible UWB testbeds of this size are not available, and ad-hoc setups are prohibitively effort-demanding. Similarly, simulators with realistic performance are unreported in the literature.

Hence, we adopt an evaluation methodology that combines a mid-scale testbed with a simulation approach yielding realistic and accurate estimates. Both are supported by a single evaluation toolchain (Figure 5.2) relying on two key software artifacts: a *simulator* of timing inaccuracies and a *TDoA solver*.

Real-world testbed experiments are executed by initializing the TDoA solver with knowledge about the anchor positions and then feeding as input the packet traces these anchors collect during a run. The solver operates based on the principles outlined in §5.1. In our case, as further detailed in §5.5.1, the testbed is composed of 12 nodes deployed on the ceiling of corridors at our office premises. While the scale of this testbed is significantly larger than commonly reported setups [106, 99, 100], it only yields a network diameter of 3 hops. Further, the geometric characteristics of corridors are ill-suited to demonstrate the localization accuracy we can attain, due to anchors lacking position diversity along one of the localization axes.

Therefore, we complement our testbed experiments with simulation, whose accurate modeling of timing inaccuracies—the main source of localization errors in TDoA—is nonetheless directly informed by real-world traces. The latter have the sole purpose of gathering enough data about the timing behavior of UWB nodes in the target scenario, e.g., due to temperature gradients. The simulator receives as input these real traces along with the actual anchor positions of the intended deployment, and outputs synthetically-generated packet traces with desired duration and frequency that faithfully represent the real traces one would observe in the same conditions.

We now describe in more detail the techniques we use to perform this accurate modeling of timing inaccuracies at the core of our simulation approach, along with experimental results confirming their accuracy.

5.4 Modeling and Reproducing Timing Inaccuracies

Timing inaccuracy is one of the main challenges in TDoA systems, as timing must be determined with a very fine granularity: a 1 ns timestamp estimation error translates to a 30 cm distance estimation error. At this granularity, inaccuracies arise from two main sources. The first one is the clock drift of typical COTS oscillators, which changes in time and can amount to several ppm. The other one is the error introduced by the timestamping of packets upon reception.

Impact of packaging on clock drift. While performing the experiments in preparation of this work, we were using both the fixed nodes in our testbed infrastructure as well as some spare nodes that we could position in various configurations to test different topologies and anchor densities.

However, when analyzing the clock drift, we noticed that the two exhibited very different trends, caused by the interplay between packaging and environmental factors. Indeed, testbed nodes are enclosed in a plastic box attached to the ceiling. This "boxed" configuration protects the electronics from temperature variations induced by air movement (e.g., caused by passers-by) that, albeit minimal in absolute, do induce nanosecond-scale timing variations. In contrast, the nodes we used for our ad hoc experiments were "naked" (i.e., without a protective case) and therefore prone to such environment-induced timing errors. We verified this phenomenon by placing each type of node in exactly the same position in our indoor office environment, observing dramatically different dynamics of clock drift (Figure 5.3). The clocks of boxed nodes are slow-changing and variations are within 0.1 ppm; instead, the clocks of naked nodes vary abruptly and in a range of almost 1 ppm.

We argue that this finding is of practical interest to developers and users of TDoA systems, as it is common to have boxed nodes in the final deployment but to use naked nodes during preliminary tests. In any case, our simulation framework can accommodate and faithfully reproduce both, as discussed next.

Modeling and reproducing clock drift. To analyze and model the clock drift, we acquire hour-long traces from several pairs of UWB devices. In each pair, one device transmits packets at a 10 Hz frequency, and the other logs the corresponding RX timestamps reported by the radio. We then compute the clock drift between every two consecutive timestamps (§5.1), obtaining a *reference* clock drift signal capturing the real-world timing behavior of our UWB devices.

To generate a *synthetic* drift signal, we create a time signal with the same power spectral density (PSD) as the reference. To this end, our simulator performs the following steps:

1. obtain the frequency domain version of the reference drift signal by applying the Fast Fourier Transform (FFT) and compute the amplitude $A(f)$ for each

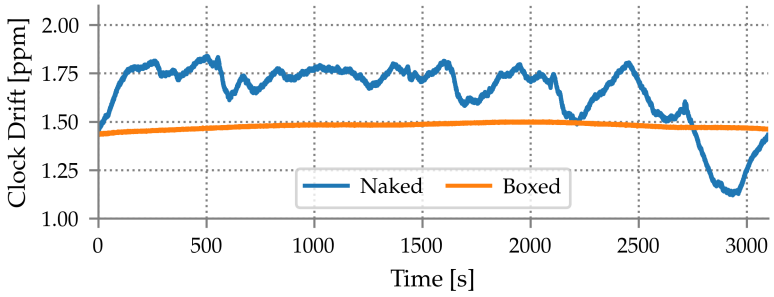
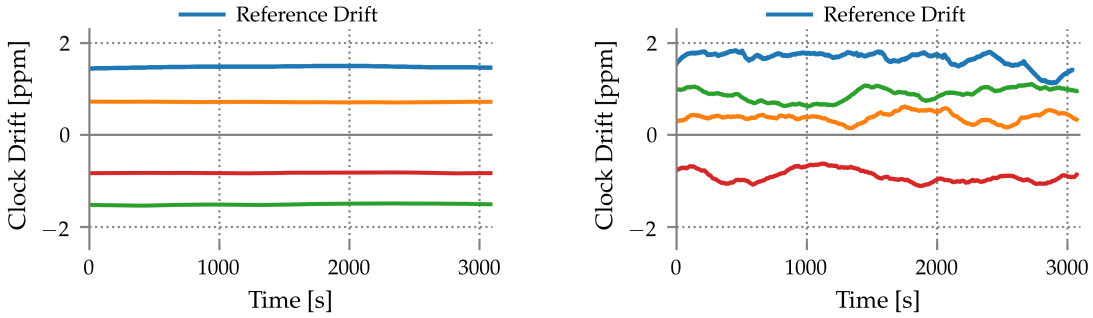


Figure 5.3: Impact of packaging on clock drift.



(a) Boxed nodes.

(b) Naked nodes.

Figure 5.4: Measured (reference, in blue) and simulated clock drift curves.

frequency;

2. assign a random phase $\phi(f) \in [0, 2\pi)$ from a uniform distribution to each frequency;
3. build a new frequency signal as $A(f)e^{j\phi(f)}$ and apply the inverse FFT to obtain the time domain version;
4. filter the generated synthetic signal with a 1 Hz digital low-pass Butterworth filter to eliminate the random RX timestamping error introduced by the radio (see below);
5. upsample the generated signal to match the timeslot frequency of the simulated system.

To generate an arbitrary number of signals, we repeat the process assigning different random phases $\phi(f)$ to each signal.

Modeling and reproducing RX timestamping errors. Another source of timing error is induced by inaccuracies in determining the time of arrival of the received radio signal. The procedure we illustrated above intentionally filters out these inaccuracies to obtain undisturbed clock drift curves. Therefore, the simulator reintroduces the timestamping error for received packets that follows a zero-mean random normal distribution $\mathcal{N}(0, \sigma^2)$. The corresponding parameters, as with clock drift, are determined based on real-world experiments. We estimate the standard deviation σ of the error with several short tests using a higher sampling frequency of 100 Hz, logging the RX timestamps. We then isolate the RX timestamping error by using the same 1 Hz low-pass filter of step *4)*, this time subtracting the trend imposed by the clock

drift. The result of this analysis showed that the measured σ lies in the $[0.14, 0.18]$ ns range for our setup; hereafter, we use $\sigma = 0.18$ ns, as we verified experimentally that this yields the best match between simulation and reality, discussed next.

Simulation vs. real-world. The techniques we described are at the core of the simulator we use in this chapter. The simulator generates synthetic packet traces whose timing error is *different* from the real ones, but whose dynamics are *very similar.* This can be visually ascertained in Figure 5.4, showing that the simulated curves faithfully reproduce the trends of the measured ones for both boxed and naked nodes. The next section provides further evidence for this claim, as we show that indeed the localization accuracy obtained by our simulator is very close to the one of our real-world prototype, confirming that our model precisely reproduces timing errors.

As a final note, topology does not bear an impact on these timing errors. Therefore, the same reference curve can be used for simulated nodes in arbitrary positions; when generating the RX timestamp traces, the simulator accounts for the propagation time between the anchors, whose position is known (Figure 5.2). Of course, other deployment effects (e.g., non-line-of-sight propagation) may affect system performance; however these are outside the scope of this study.

5.5 Evaluation

We assess the ability of Talla to provide continuous and accurate localization over large-scale areas, using simulation and testbed experiments as outlined in §5.3.

5.5.1 Experimental Setup

Hardware, firmware, software. We employ the DecaWave EVB1000 platform [71], featuring an STM32F105 MCU and the DW1000 UWB transceiver with a PCB antenna. The firmware is implemented atop Contiki OS, using the port for the EVB1000 described in [83]. Finally, our TDoA solver and our simulator are implemented in Python, based on the techniques outlined in §5.1 and §5.4, respectively.

RX/TX timestamp calibration. Our TDoA solver uses both the TX timestamps of the reference anchor and the RX timestamps of others. We noticed that, without calibration, location estimates show a decimeter-level bias towards the reference anchor. We attribute this to a discrepancy between TX and RX timestamps, probably caused by a difference between the TX/RX antenna delays. To compensate, we apply a constant correction of -53 ticks (-0.83 ns) to the TX timestamps.

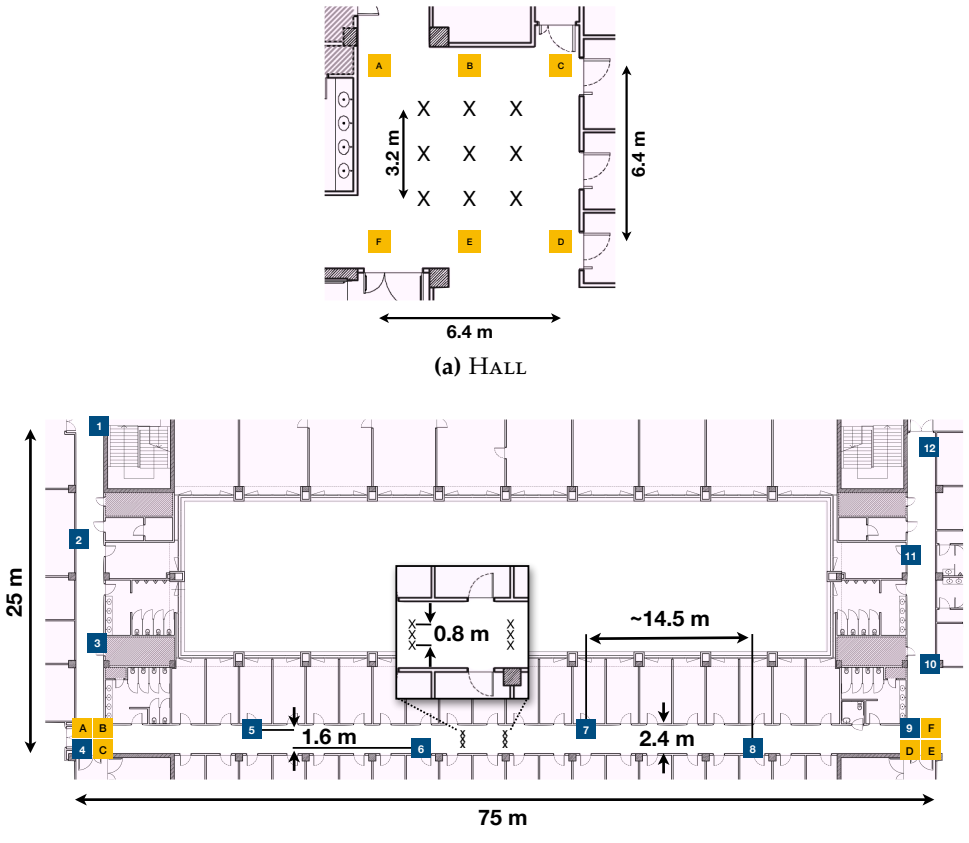
Experimental facilities. Our testbed consists of 12 fixed nodes attached to the ceiling of our office building. The experiments presented in this chapter have been performed before those of Chapter 4, when the testbed was not complete yet. However, for this dissertation, we have extended the evaluation to an additional 18-node area of the new testbed. Each UWB node is connected to a JTAG programmer and a Raspberry Pi. The ceiling nodes use a wired Ethernet infrastructure for automated experiment control and collection of the logs. This setup allows us to easily schedule and run many experiments without handling the nodes individually. Moreover,

we used 7 additional portable nodes that could be placed anywhere in the building and/or used as the mobile tags being positioned.

Anchor placement. We define two setups (Figure 5.5): *i)* HALL, a regular deployment of 6 portable anchors placed 70 cm above floor on the perimeter of a 6.4× m^2 square, and *ii)* CORRIDOR, a significantly sparser deployment along a U-shaped corridor whose ceiling hosts our fixed testbed nodes.

These two setups have very different characteristics. In HALL, all nodes are within communication range of each other. Further, the square deployment provides a good (low) dilution of precision (DOP) [109] along both axes. HALL is representative of typical scenarios for indoor localization (e.g., a room) and therefore we take it as our baseline for evaluating our system. Moreover, we also base our large-scale simulation in §5.5.3 on HALL, by replicating its topological characteristics over a much bigger area with over a hundred anchors, while faithfully reproducing the corresponding timing errors (§5.4).

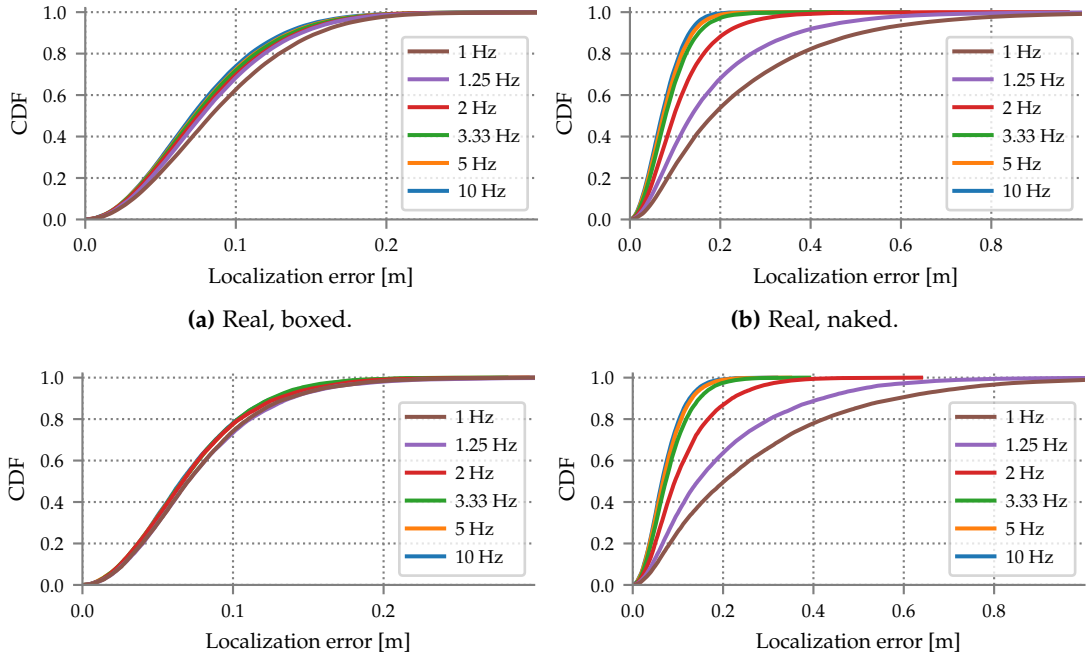
In contrast, the topology of CORRIDOR is definitely sub-optimal. The width of the corridor area is only 2.4 m, generating significant multipath effects on the radio signal and yielding a high DOP (i.e., high uncertainty), albeit partially ameliorated by the



(a) HALL

(b) CORRIDOR

Figure 5.5: Anchor deployments. A dark blue square denotes a stationary anchor attached to the ceiling, an orange square stands for a portable anchor, and an X represents a ground-truth landmark.



(a) Real, boxed.

(b) Real, naked.

(c) Simulated, boxed.

(d) Simulated, naked.

Figure 5.6: Localization error vs. synchronization rate in HALL (6 anchors).

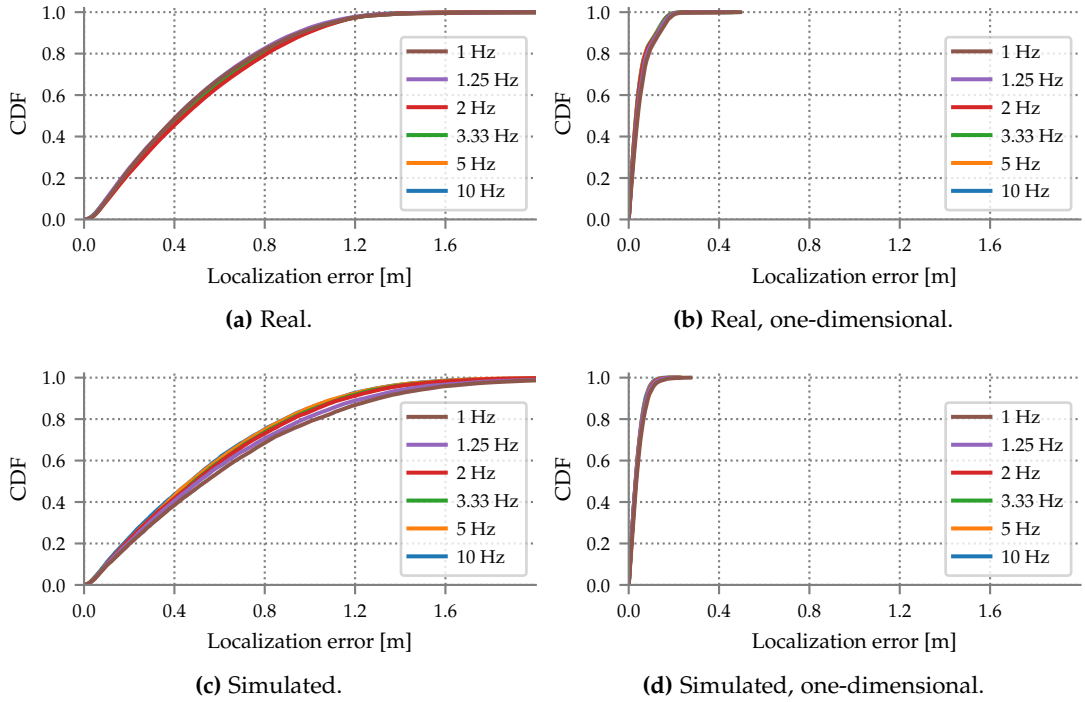
zig-zag placement of anchors across the ceiling. On the other hand, CORRIDOR is the only "large-scale" testbed we have access to, with a diameter of 3 hops, essentially corresponding to each of the corridor segments (Figure 5.5b). Therefore, its value lies in the ability to run real-world experiments and *directly* validate our claims, although the localization accuracy we obtain is not indicative of the typical one that we can expect, instead represented by HALL. For these reasons, we evaluate TALLA in both setups.

5.5.2 Small-Scale, Single-Hop Experiments

We establish a baseline to compare our large-scale scenario against by evaluating the localization accuracy in small-scale experiments representative of common TDoA deployments with all anchors in range. This serves also as a validation of our simulator, showing its ability to faithfully reproduce timing errors and therefore localization accuracy.

We consider HALL (Figure 5.5a) and the portion of CORRIDOR with nodes 4–9 (Figure 5.5b). We manually measure with a laser pointer ground-truth coordinates for 9 and 6 landmarks, respectively, and acquire location estimates at each of them at the rate of 90 samples/s for 5 minutes, yielding ≈27000 samples per landmark. In each setup, one anchor serves as time reference, periodically broadcasting synchronization beacons.

Synchronization rate vs. localization accuracy. TDoA is very sensitive to clock drift; the rate at which time synchronization beacons are sent affects the system performance. Hereafter, we experiment with rates between 1 Hz and 10 Hz.



(a) Real.

(b) Real, one-dimensional.

(c) Simulated.

(d) Simulated, one-dimensional.

Figure 5.7: Localization error vs. synchronization rate in Corridor (6 anchors, boxed only). Note the different x-axis scale w.r.t. Figure 5.6.

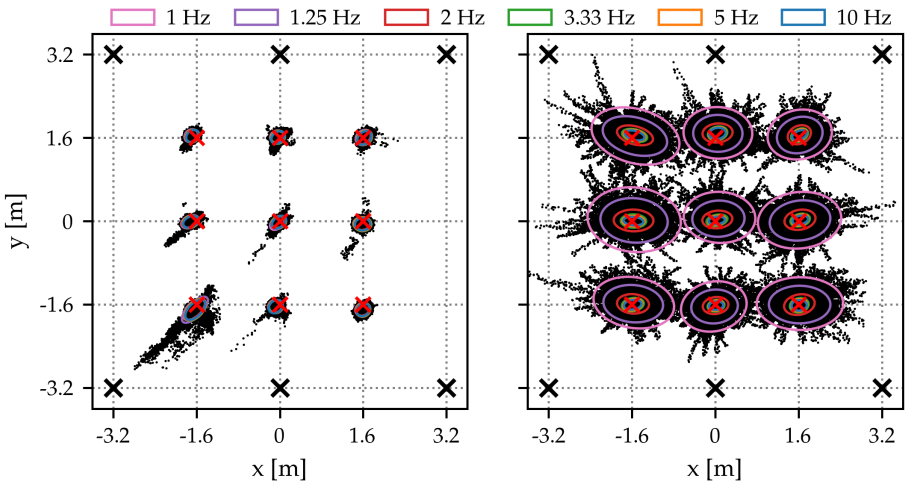


Figure 5.8: Localization error vs. synchronization rate in real Hall experiments with boxed (left) and naked (right) nodes. The external black crosses are anchors, the internal red ones are landmarks. Error ellipses denote the 3×standard deviation for a given rate. Black dots are individual samples.

Figure 5.6 shows the corresponding CDFs of localization error in Hall, for both boxed and naked nodes. The setup with boxed nodes show excellent localization accuracy; 99% of the samples fall within 20 cm of the ground-truth landmarks, regardless of the synchronization rate (Figure 5.6a). As mentioned in §5.4, the encasing protects nodes from temperature variations and yields very slow changes in the clock

drift, which can be compensated precisely (§5.1). The same holds in the case of naked nodes, but only when the synchronization rate is $\geq$3.3 Hz; otherwise, the variations induced by the environment cannot be successfully compensated (Figure 5.6b). Figure 5.8 offers an alternate view for real experiments, showing the actual spread of sample points with error ellipses denoting the 3$\times$ standard deviation of the error, further reasserting the impact of packaging on clock drift and localization accuracy.

Figure 5.7 shows analogous results for the portion of Corridor considered. In this case, we show only the case with boxed anchors, as they reflect the actual deployment. As in Hall, the synchronization rate does not bear a significant impact in the value range considered. However, as expected, the localization accuracy (Figure 5.7a) is significantly worse than in Hall, due to the very narrow geometry of the setup—a rectangle 2.4 m wide and 75 m long. Given this extreme shape, the localization accuracy is actually still reasonable, but not very indicative. Moreover, we observe that often what matters in a corridor is to localize where the target is along its length rather than a fine-grained localization including its width. Therefore, we also report the results for this one-dimensional case, which show high accuracy (Figure 5.7b).

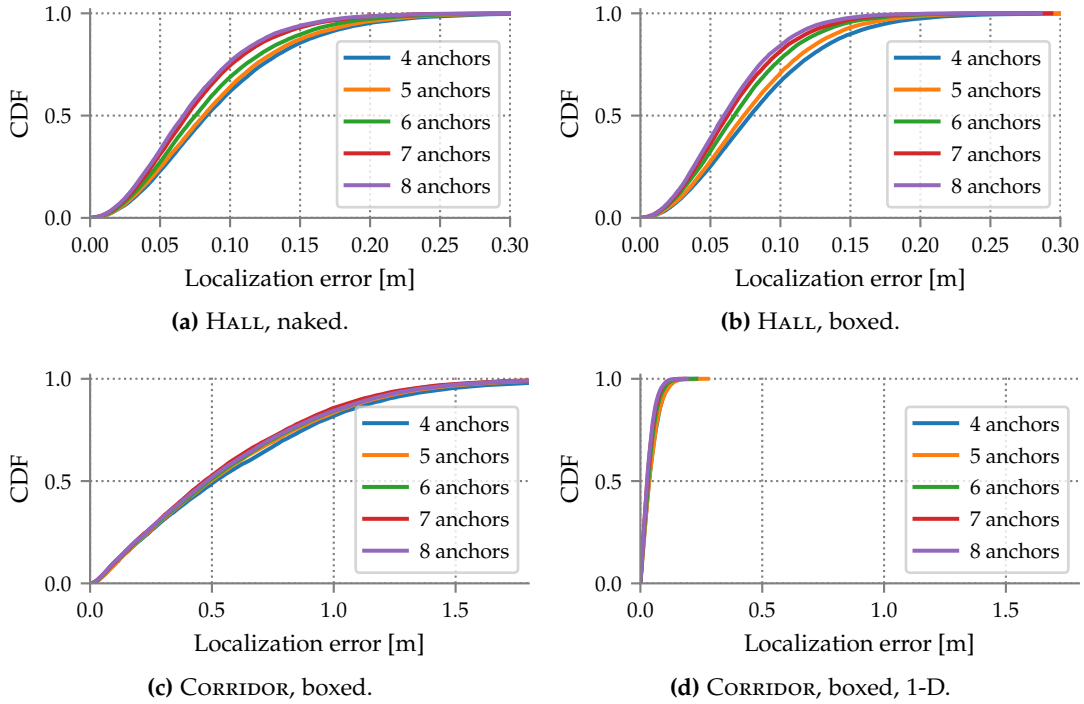
Simulation vs. reality. The bottom rows of charts in Figure 5.6–5.7 show the results output by our simulator when configured to replicate the real setup. Comparison between real and simulated results confirm that the simulator faithfully reproduces the real-world behavior of our TDoA localization. The small difference between real and simulated results can also be appreciated in Table 5.1, for various error percentiles. The largest difference is found in the Corridor with two-dimensional localization, a scenario that is severely challenged by geometry and multipath effects. In all other cases, simulated and real results are in very good agreement. This confirms that we can safely use the simulator to analyze the performance of TDoA systems in general, including large-scale deployments.

Number of anchors vs. localization accuracy. Empowered by our simulator, we study an aspect crucial to deployments: the effect of the number of anchors on localization accuracy.

We explore a number of anchors ranging from 4 (i.e., the minimum required by TDoA) and 8. In Hall, we obtain the minimum by removing anchors B and E; dually, we obtain the maximum by inserting one anchor in the middle of A–F and C–D. In Corridor we start from anchors 5–8 and mirror them by adding an anchor on the other side of the corridor, most outer ones first. Figure 5.9 shows the CDFs in various configurations, and Table 5.2 shows the corresponding values. Interestingly, both Hall and Corridor appear to be only marginally affected by the number of anchors, even in the case of naked boxes. We verified that this is the case also in the corresponding real-world experiments.

5.5.3 Large-scale, Multi-hop Experiments

We evaluate our system in three large-scale, multi-hop scenarios. The first one, Grid, is a synthetically generated one, taking advantage of our simulator. Grid can be seen as a tiling of the Hall scenario across a much larger area. The next scenario is Corridor, enabled by our real-world testbed.



(a) Hall, naked. **(b)** Hall, boxed.

(c) Corridor, boxed. **(d)** Corridor, boxed, 1-D.

Figure 5.9: Effect of the number of anchors on the error distribution with a 3.3 Hz synchronization rate in simulation.

Table 5.1: Localization error for Figure 5.6–5.7 with a 3.3 Hz synchronization rate.

			Percentile [cm]			
			75^{th}	90^{th}	95^{th}	99^{th}
Real	Hall	boxed	11	13	15	21
		naked	10	14	16	21
	Corridor	2D	69	89	101	377
		1D	6	9	10	14
Simulated	Hall	boxed	9	12	14	18
		naked	9	11	14	18
	Corridor	2D	81	112	130	531
		1D	5	7	9	12

With respect to the work presented in [4] that provides the basis for this chapter, we include an additional scenario, Entrance. As discussed in this evaluation, Corridor is sufficient to test Talla in a multi-hop network, but the placement of anchors is unsuited for accurate localization. Instead, Entrance uses a separate part of the testbed, installed more recently, with better anchor geometry and therefore better accuracy.

Simulation experiments: Grid. We model two large rectangular fields connected by a "passageway", and fill the area with a regular grid of anchors with a step of L. This configuration combines both wide, open areas and a narrow one, and is

Table 5.2: Impact of the number of anchors on localization error (cm).

Percentile		4 Anchors				8 Anchors			
		75^{th}	90^{th}	95^{th}	99^{th}	75^{th}	90^{th}	95^{th}	99^{th}
Hall	boxed	11	14	17	22	8	11	12	16
	naked	11	15	17	23	9	11	14	18
Corridor	boxed 2D	86	118	143	600	81	113	132	538
	boxed 1D	6	9	10	14	4	6	8	10

representative of situations found in several indoor and outdoor large areas, e.g., halls of large buildings, parking lots, etc. The rectangles fit 9×9 anchors each, while the passageway contains 9×2 anchors, totaling 180 nodes (Figure 5.10). To ensure uniform coverage through this anchor placement, we set the communication range to $3 \times L$; geometrical considerations guarantee that a tag is always in range of 4 to 9 anchors. This results in a network diameter of 26 hops.

In Talla, all N anchors broadcast synchronization beacons; however, the number n of their time slots in an epoch can be defined to be $n \ll N$ (§5.2). Indeed, anchors 3 hops away from each other do not interfere and can safely reuse the same slot. Therefore, $n = 9$ slots can accommodate synchronization beacons from all $N = 180$ anchors in Grid.

Figure 5.10 shows an example of tracking a mobile tag in Grid; the positions estimated by Talla are compared against a ground truth trajectory. As the tag moves, the set of anchors used for positioning changes, and so does the number of anchors (Figure 5.11a). Nevertheless, position estimates do not exhibit gaps, witnessing the ability of Talla to support *continuous* localization across large areas. Moreover, Figure 5.11b also demonstrates that this result is achieved *without sacrificing localization accuracy*. Indeed, the median error is 6 cm and the maximum is below < 30 cm across all 3000 points visited by the tag; these results are comparable to those we observed for Hall in both simulation and reality (§5.5.2).

Testbed experiments: Corridor. We demonstrate the effectiveness of Talla holds also in reality, although in a more limited setup w.r.t. the synthetic one in Grid. In these experiments, a person carrying the tag walks at a speed of ≈ 0.5 m/s in the center of the Corridor U-shaped path, from one extreme to the other. As the person passes the corners, the system dynamically switches the sets of anchors it relies on.

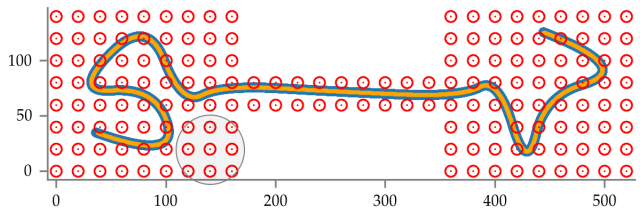
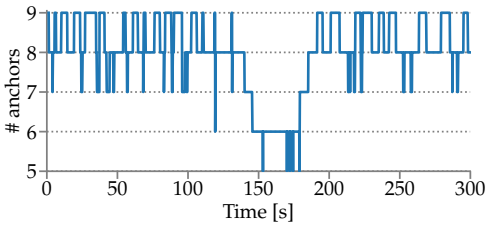
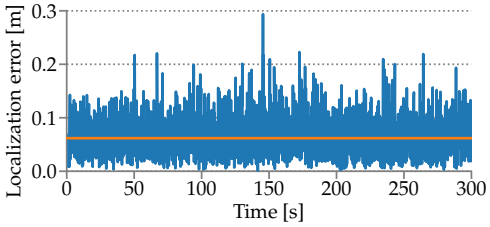


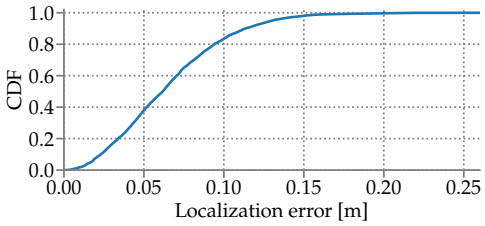
Figure 5.10: Grid: ground truth (blue) vs. estimated position (yellow).



(a) Number of anchors used over time.



(b) Localization accuracy over time.



(c) Overall localization accuracy.

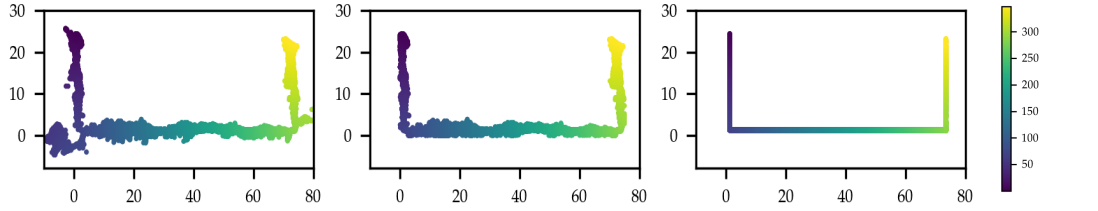
Figure 5.11: Grid: Number of anchors and localization accuracy.

The anchor placement in Corridor is slightly different w.r.t. the one in §5.5.2. Our ceiling testbed was originally motivated by the need to run experiments about communication quality. However, while experimenting with Talla we discovered that the testbed node layout is problematic for localization; a few nodes (3, 5, 10) near the corners are in communication range with the tag but not in line of sight, causing a large positioning error. Therefore, we removed these from the analysis and added 6 portable anchors (A–F), 3 in each corner, to ensure line of sight (Figure 5.5b).

Figure 5.12 shows the results obtained from our experiments, plotting the tag trajectory in the Corridor area. Our testbed setup is not equipped to gather precise ground-truth information about the location of the tag. Therefore, we encode the time information associated with the estimated trajectory as a color gradient. This enables us to visually ascertain the correct and continuous operation of Talla localization.

As for localization accuracy, we analyze different configurations of our TDoA solver. When configured without any knowledge of the environment, the solver outputs the results in Figure 5.12a, showing a high variance along the axis perpendicular to the corridor and suboptimal performance around corners due to residual non-line-of-sight and DOP in those areas. Nevertheless, the one-dimensional positioning—arguably the relevant one in a corridor area like ours—remains very accurate, as confirmed by the smooth color gradient.

However, in a practical deployment targeting a geometrically-challenged area like Corridor, it is common (and easy) to configure the TDoA solver with knowledge of the area. Figure 5.12b shows that, by simply setting the outer boundaries of the corridor at the corners, the solver not only automatically drops unreasonable estimates but also improves them. This simple modification, whose nature has to do with



(a) Unbounded **(b)** Bounded **(c)** One-dimensional **(d)** Time, *s*.

Figure 5.12: Estimated trajectories in the Corridor. The color gradient represents time, axis values are in meters.

the "local" TDoA localization and not with our multi-hop scheme, enables Talla to successfully track the tag without gaps across the 3 hops in Corridor, with a localization accuracy similar to what we observed in §5.5.2. Finally, Figure 5.12c shows the results when one-dimensional localization along the corridor path is performed. As observed in §5.5.2, the accuracy is even higher, and the correct and continuous operation of Talla even more evident.

Testbed experiments: Entrance. We present another test to confirm Talla can support continuous operation in a different part of the testbed that was not available during the initial campaign of experiments [4]. We refer to this additional scenario as Entrance, since anchors are in fact installed in the foyer of one of the university buildings. Two large rooms are divided by sliding glass doors, that, when open, let anchors from the two areas synchronize yielding a unified localization network. The infrastructure is made of 18 anchors, positioned along the edges of the two rooms and well distanced along the two localization axes to ensure low DOP and therefore high accuracy in 2D.

Similar to Corridor experiments, a person carries the tag and moves at a speed of $\approx$ 0.5 m/s, back and forth between the two rooms making an L-shaped path. A trajectory captured by Talla in Entrance is shown in Figure 5.13, again with the color gradient representing time. The total duration of the trajectory is 338 s. The trajectory is continuous and steady as the tag crosses from one area to another, which is further proof of Talla capabilities.

However, if the Corridor scenario highlighted the issues created by non-line-of-sight (NLOS) between anchors and the tag, Entrance brings out evidence for errors induced by NLOS between two anchors. Since NLOS leads to timestamp errors, TDoA synchronization is also affected, ultimately causing localization errors. While the problem is not to be ascribed to the logic of Talla, we comment briefly on how we solved it in this experiment to obtain the steady trajectory shown in the figure. In our preliminary tests, we observed a significant increase in positioning error while the tag was crossing the sliding doors, that would be avoided by disabling anchor 51. The outliers were due to consistent bad synchronization between anchor 51 and those in the upper part of the deployment (specifically 70, 75, 76). We verified that those links (51-70, 51-75, 51-76) are affected by NLOS by performing two-way ranging for each anchor pair. The test yields meter-level ranging error, confirming the hypothesis.

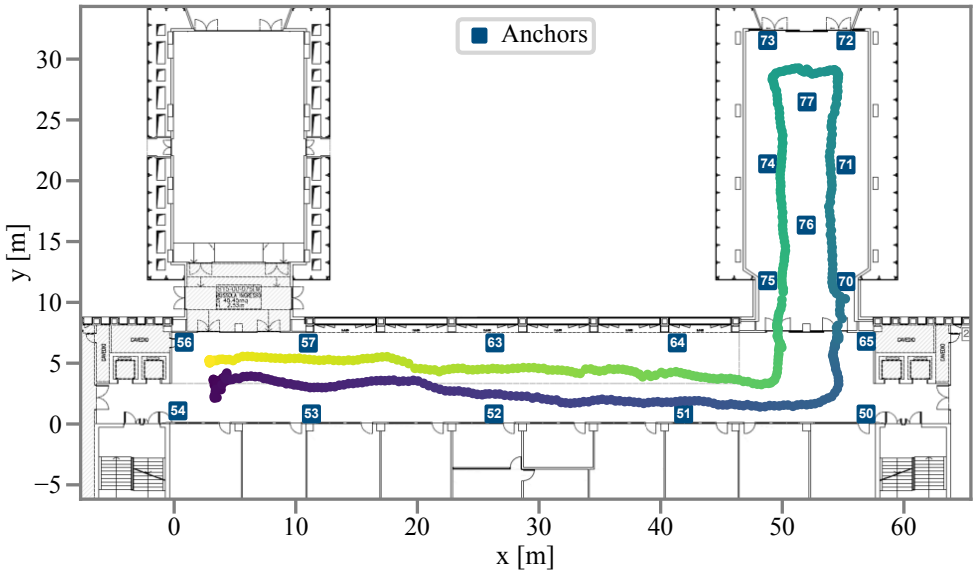


Figure 5.13: Estimated trajectory in the ENTRANCE.

Since anchor 51 is still useful for localization in its primary area, we do not disable it completely. Instead, we modified TALLA to ignore the reception of anchor beacons associated to the aforementioned links affected by NLOS. Since the TDoA solver does not receive the required synchronization information, anchor 51 can never be used in the same anchor subset of 70, 75 or 76 for localization. This simple change was sufficient to avoid the outliers we initially observed.

Since in TALLA anchor positions are known, this procedure to exclude NLOS links can be easily replicated in the setup phase, together with the neighbor assessment for the schedule definition discussed in §5.2. As NLOS detection techniques improve [110, 111, 112, 113], TALLA may include them in the anchor firmware to react to link changes immediately, rather then requiring a dedicated assessment phase.

5.6 Related Work

Recent research on UWB TDoA localization consists of *i)* conventional schemes, with tightly-synchronized anchors listening to mobile tag transmissions, and *ii)* reversed schemes, where tags are passive listeners of anchor transmissions.

Conventional TDoA. TALLA builds on the mechanisms in [104] and the ATLAS localization system [105, 106]. ATLAS is based on *i)* a 1-hop star topology, resulting in limited spatial scalability, and *ii)* random access to the wireless medium, leading to performance degradation as the tag number or positioning rate increase [103]. To avoid collisions between tags and anchors, ATLAS assigns a different UWB preamble code for synchronization, possibly leading to timestamping errors (see §2.4.3). TALLA's TDMA schedule tackling large areas implicitly solves this issue, avoiding collisions altogether. Another significant difference, besides implementation details, is the ATLAS dependency on a dedicated external node for synchronization; this is

not required in Talla, which simplifies deployment by relying solely on the required anchors.

A recent extension, ATLAS-FaST [114], targets high-rate positioning also via a TDMA scheme. The authors mention that this could be exploited to scale to large areas, but do not provide experimental evidence for this claim. In contrast, we evaluate Talla with both real-world and simulated experiments. Further, the methodology and techniques in the latter (§5.3–§5.4) are a contribution per se, enabling further developments in the general field of TDoA localization.

Reversed TDoA. In Chorus [99] and SnapLoc [100] anchors transmit beacons concurrently, and tags self-position by estimating TDoA from anchor peaks in the channel impulse response (CIR). Loco [115] instead employs staggered beacons, transmitted sequentially. These systems support high update rates for unlimited tags, as these are passive listeners. Positioning information remains local to the tags, which is an asset for some applications and a drawback in others (e.g., logistics). None of these solutions addresses large-scale operation.

5.7 Conclusions and Future Work

We presented Talla, a novel TDoA scheme enabling continuous and accurate localization across large-scale operational areas. We evaluated the TDMA design of Talla both in a real-world and simulated setups; in the latter, we designed a novel technique enabling us to faithfully reproduce the timing inaccuracies of UWB devices causing TDoA positioning errors. Results show that Talla achieves state-of-the-art accuracy in small-scale single-hop settings but also when targets move and change their set of anchors in range, confirming that Talla enables uninterrupted, reliable positioning across large areas.

Future work on the topic revolves around real-time NLOS detection and correction, both for links between anchors and for links between the tag and the anchors. By assessing link conditions and exploiting this information in the TDoA solver, we aim to improve the anchor selection process and to reduce the overall localization error. This will enable real-world experiments in more complex setups, for example with moving obstacles causing unpredictable changes in the link conditions.

Acknowledgements. The authors are grateful to Francesco Giopp, whose M.Sc. thesis [116] explored the core ideas described here, and Davide Molteni, for his help in performing testbed experiments.

6

Fine-grained Stop-Move Detection in UWB-based Trajectories

The increasing availability of *spatial trajectories* [117, 118] from Global Navigation Satellite Systems (GNSS) and efficient techniques to process them [119, 120] enable the extraction of *mobility patterns*, application-specific abstractions of the movement of individuals [121, 122]. Existing approaches focus on the large-scale, outdoor settings germane to GNSS, yielding coarse spatio-temporal resolution. However, a recent technological wave targets sub-meter position accuracy even in indoor spaces, a powerful enabler in several applications [39].

Leading this wave, ultra-wideband (UWB) radios enable communication *and* accurate localization. WiFi and Bluetooth also offer both, but with a positioning error of meters [39]. UWB yields decimeter-level accuracy by relying on narrow pulses ($\leq$2 ns) improving time-of-arrival estimation and separation from multipath components (§2.1). The increasing role of UWB in location-based applications is witnessed by the many Real-Time Location Systems (RTLS) based on it, and its recent inclusion in smartphones, albeit still awaiting public APIs.

We exploit Talla, introduced in the previous chapter, to acquire human movement trajectories, and evaluate whether UWB can support the analysis of fine-grained mobility patterns.

Research questions. Existing mobility analysis techniques, limited by GNSS, target applications over large areas (m to km) and temporal intervals (hours to months). UWB trajectories, accrued with much higher spatial resolution and temporal frequency, should intuitively unlock finer-grained analyses, e.g., to decimeters and sec-

This chapter revises our publication [5]: F. Hachem, D. Vecchia, M. L. Damiani, and G. P. Picco. "Fine-grained Stop-Move Detection in UWB-based Trajectories". In *Proceedings of the* 20^{th} *IEEE International Conference on Pervasive Computing and Communications (PerCom)*. 2022.

onds. However, the increased spatio-temporal density of UWB positions, along with errors induced by indoor environments, may conflict with the operation of GNSS-based techniques or limit the resolution attainable. Unfortunately, UWB-based mobility analyses are largely unexplored, leaving crucial questions unanswered: 1. *Are existing techniques applicable to UWB trajectories, and* 2. *to what* quantitative *extent do they improve spatio-temporal accuracy?*

Focus: Stop-move detection. We seek answers to these questions by focusing on the well-known stop-move [123] mobility pattern (§6.1), key to many applications, for which we illustrate representative techniques (§6.2). Broadly, a *stop* is an abstraction capturing a visit to a place; a *move* is a transition between stops. In a trajectory, stops are spatio-temporally disjoint: 1. an individual visits one place at a time, and 2. visits to the same place are distinct stops. Stops may represent the home range of migrating animals, lasting months over a large area, or people visits to a point of interest, e.g., the workplace, lasting hours. This application-dependent spatio-temporal granularity differentiates our work from the literature, as we consider stops only few decimeters apart and lasting only few tens of seconds.

Real-world requirements and setup: Science museum. These challenging requirements stem from a motivating real-world use case also offering the concrete experimental setup where to distill findings. We are collaborating with the curators of the MUSE science museum (Trento, Italy), interested in the fruition of exhibits by visitors. This is hardly a novel topic; however, existing works [124, 125, 126, 127] rely on Bluetooth and are limited to coarse spatial resolution, e.g., room-level. In contrast, our target museum area contains exhibits within few decimeters of each other (Figure 6.1). Reliably discerning stops near them is basically impossible via Bluetooth. Further, the curators are interested in stop durations as short as 10 s, yielding precious insights on the visitors' behavior.

Contributions and methodology. Real-world validation entails comparing stop estimates vs. ground truth. This is often done qualitatively, or by using metrics (e.g., focusing on individual points [128]) that cannot faithfully capture the fine-grained spatio-temporal features we target (§6.6). Therefore, we contribute a **novel metric** (§6.3) that 1. associates estimated stops to true ones on a per-stop level, and 2. quantifies their temporal overlapping with a novel indicator, S-score, along with the classic F-score summarizing precision and recall.

We base our results on experiments in the museum (§6.4). We track users wearing an UWB tag via Talla, i.e., time-difference-of-arrival (TDoA) localization, while recording their ground-truth movement via a user-operated smartphone application and cameras we deployed. The 9 trajectories we gather consist of 70000+ position samples and 209 stops over 100 minutes.

We exploit this dataset for a **quantitative analysis along several dimensions** of UWB-based stop-move detection (§6.5). We first confirm the expressiveness of our novel metric, then use it to compare techniques after selecting their best configuration in our context. We consider both the raw trajectories output by UWB and those "smoothed" via Kalman filters. Finally, we quantify spatio-temporal errors by sharply separating the contributions of segmentation and positioning.

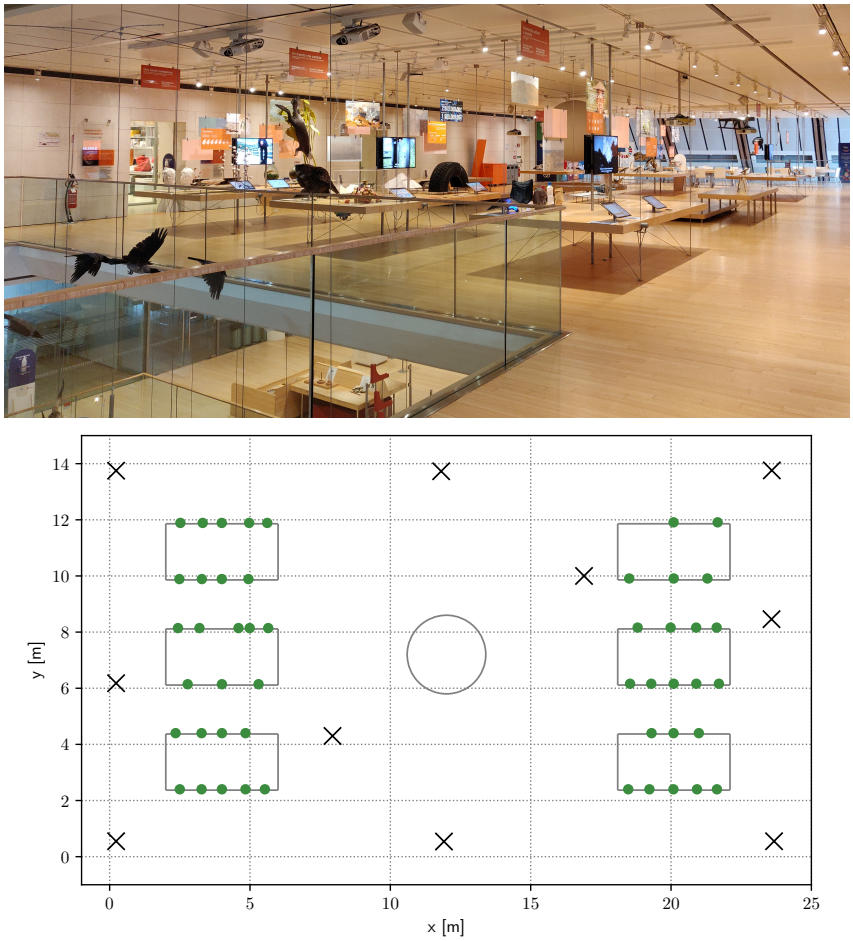


Figure 6.1: Museum target area (top) and map (bottom). Dots are points of interest (POI) in front of exhibits; crosses are UWB anchors on the ceiling.

Despite our challenging setup, the best technique correctly detects 186 stops out of 199, estimates their duration with an average error of 3.4 s and their position with an extra 3.1 cm error w.r.t. the larger ones from UWB localization, which we identify as the main source of spatial error. In absence of ground truth, the estimated stop can be correctly associated to the closest exhibit in 88.9% of the cases. Overall, these findings provide positive, quantitative answers to our research questions, pushing the applicability of mobility analysis to unprecedented fine-grained spatio-temporal resolution.

Our survey of related efforts (§6.6) shows that this work is the first studying stop-move detection on UWB-based trajectories and, importantly, to offer an evaluation against systematically-acquired ground truth in a real-world environment. We end the chapter with concluding remarks (§6.7).

6.1 Problem Formulation and Definitions

A *trajectory* $T=\{(p_1,t_1)\dots(p_n,t_n)\}$ is a sequence of positions p_k and associated timestamps t_k, sampling the movement in space of an entity, a user in our case. We call (p_k,t_k) a trajectory *unit*. A *stop* or *segment* is a sub-sequence $s=\{(p_i,t_i)\dots(p_j,t_j)\}\subseteq T$ capturing the user permanence in the area represented by the centroid of positions $\{p_i\dots p_j\}$ during the time interval $[t_i,t_j]$. A *segmentation* of T is a sequence of temporally disjoint stops. Abstractly, the problem is to extract from a trajectory T the segmentation $S=\{s_1\dots s_m\}$ of *estimated* stops via a detection operation $m(T,\rho,\Pi)$. Stops shorter than the application-dependent temporal threshold ρ are irrelevant and neglected; the set Π of configuration parameters depends on the technique at hand, as described next.

6.2 Segmentation Techniques

We summarize the stop-move detection techniques we compare (§6.5) over our dataset, representative of state-of-the-art approaches (§6.6) with different complexity and tradeoffs.

Using spatial distance between units. A stop s can be seen as the sequence of time-consecutive units $\{(p_i,t_i)\dots(p_j,t_j)\}$ whose spatial distance from the start of the segment is smaller than an application-dependent threshold $\delta\in\Pi$, i.e., $|p_i-p_j|<\delta$. Segments with duration $[t_i,t_j]<\rho$ are ignored.

This approach was proposed in [129] as Stay Point Detection (SPD) to identify visits to point of interests in GPS trajectories, and is commonly used due to its simplicity. However, it is not well-suited when stops have different spatial size (δ) or are affected by outliers, e.g., due to positioning noise.

Using unit density. These limitations can be tackled by via density-based clustering. In DBSCAN [130], a *core point* in some abstract space has at least N neighbors within distance ϵ; a cluster contains these points and, transitively, those of neighboring core points. Unfortunately, when used for segmentation, these methods cannot guarantee the temporal separation of clusters except when spatially far from each other; recurrent stops, e.g., at the same exhibit, become indistinguishable.

SeqScan [131] overcomes this limitation by defining a stop as a cluster of units $\{(p_i,t_i)\dots(p_j,t_j)\}$ where $\{p_i\dots p_j\}$ is a DBSCAN cluster. Unlike SPD, clusters can thus have arbitrary spatial shape; unlike DBSCAN, SeqScan clusters have (disjoint) time intervals $[t_i,t_j]$; those $<\rho$ are ignored.

Using user velocity (from Kalman filters). Another way to look at stops is when user velocity is (nearly) zero. For generality, we do not determine it with sensors, rather compute it directly from UWB trajectories, whose noisy raw positions however induce unacceptable velocity jitter. Nevertheless, these trajectories are typically improved via Kalman filters (§6.4.1) whose operation already entails hidden state variables representing the velocity associated with units (p_i,t_i). Segmentation then simply consists of identifying consecutive units whose velocity is greater than a threshold $\theta\in\Pi$, ignoring segments $[t_i,t_j]<\rho$. This Kalman-based velocity technique

(hereafter, KBV) is, to our knowledge, novel in stop-move detection; it is interesting as a computationally cheap approach reusing the filtering often already applied to trajectories.

6.3 A Novel Metric for Stop-Move Detection

Crucial to the practical application of segmentation techniques is a clear understanding of the quality they offer. This is important not only to compare alternatives but also their different parameter configurations. To provide reliable results, the output by a technique with a given configuration must be evaluated against a ground-truth segmentation; the question is how to measure the quality of the former vs. the latter.

Baseline: Unit-centric metric. A recent approach [128] is to quantify this quality at the *unit* level. A unit (§6.1) belonging to an estimated stop is a true positive (TP) if also part of a ground-truth stop (hereafter, true stop); a false positive (FP) if it is not. Dually, a unit belonging to no estimated stop is a false negative (FN) if part of some ground-truth stop; otherwise, it is a true negative (TN), hereafter disregarded. The values of TP, FP, FN can be used to compute popular metrics like precision $P = \frac{TP}{TP+FP}$, recall $R = \frac{TP}{TP+FN}$, and the *F-score* $= 2\frac{P \times R}{P+R}$ offering a single-value, concise quality indicator.

Unfortunately, this unit-centric approach is oblivious to the higher-level *structure* of the segmentation. By focusing on which *units* fall into the time interval of true stops, it does not capture properties of the estimated *stops* these units belong to. Even something as simple as the number of correctly identified stops is lost in the aggregated, unit-centric view.

Contribution: A stop-centric metric. We propose an alternative metric that aims at directly *matching* each true stop with an estimated one. We use the same popular metrics above (TP, FP, FN, precision, recall, F-score), but defined at the level of stops rather than units. For instance, precision is directly the fraction of true stops over all estimated ones, rather than the fraction of units belonging to true stops over all units belonging to estimated stops. We argue, and confirm quantitatively (§6.5), that this change in the "lens" used to analyze segmentations increases expressiveness and practical relevance. However, this approach also raises new problems, discussed next.

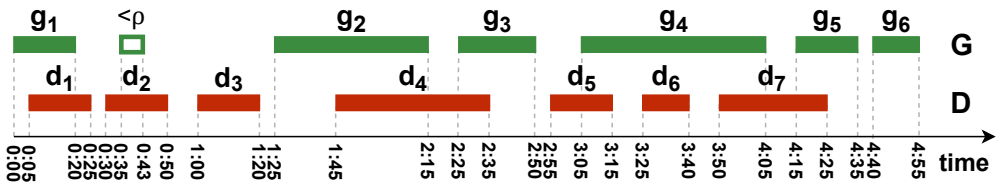


Figure 6.2: An (artificial) example of segmentation. G is the set of time intervals of true stops, D the one for estimated stops.

Matching estimated and true stops (F-score). Consider Figure 6.2, an artificial ex-

ample to illustrate all relevant cases, where G and D are the sets of time intervals associated to true stops and estimated stops, respectively. Our goal is to establish a one-to-one relationship between them, based on the intuition that if $g \in G$ and $d \in D$ represent the same stop, their time intervals must overlap (ideally, coincide). Still, an estimated stop can overlap multiple true stops and vice versa. How can the matching be performed?

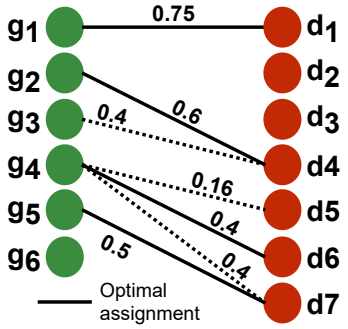


Figure 6.3: Finding the matching set.

We cast the problem as an unbalanced assignment optimization over the bipartite graph (G, D, E, W), where E is the set of edges (g_i, d_j) linking overlapping stops, and W the set of associated weights $w_{ij} = \frac{g_i \cap d_j}{g_i}$ capturing how much of the true stop g_i overlaps with the estimated one d_j. The optimal assignment yields the *matching set* M maximizing the sum of weights. Applying these concepts to Figure 6.2 yields the graph in Figure 6.3. Solving the assignment problem yields

$$M{=}\{(g_1, d_1), (g_2, d_4), (g_4, d_6), (g_5, d_7)\}$$

from which

$$TP{=}\{d_1, d_4, d_6, d_7\}, FP{=}\{d_2, d_3, d_5\}, FN{=}\{g_3, g_6\}$$

can be derived, along with precision, recall, and F-score.

Determining the nature of false detections. This can be inferred for a segmentation technique based on the relationship between an estimated segment d and a true one g.

A false positive d is an actual *fake* if it does not overlap with any true stop g (e.g., d_3 in Figure 6.2). Instead, when $d \cap g \neq 0$ the FP is a *split* stop, i.e., a true stop is estimated as two or more separate ones (e.g., g_4 matched by d_6 and overlapping with d_5). Indeed, as d is a FP it does not match any g but some other $d_i \neq d$ must, otherwise the assignment would not be optimal. A special case is a *short* stop, when the duration of g is $< \rho$ (i.e., irrelevant, §6.1) but the one of d is not (d_2).

Similarly, a false negative g is an actual *missing* stop (g_6) when $g \cap d = 0$, otherwise it is a *merged* stop, i.e., two or more true stops estimated as a single one (e.g., g_3 lumped into the same estimate d_4 matched to g_2). Indeed, g is not matched by d although they overlap; thus, d must match another $g_i \neq g$ otherwise, again, the assignment would

not be optimal. As in the FP case, a FN short stop could capture a true g incorrectly estimated by d with duration $< \rho$. However, in practice, these are automatically filtered by segmentation techniques.

Quantifying the similarity of matched stops (S-score). In our example, M contains both (g_1, d_1) and (g_2, d_4), whose temporal overlapping between estimated and true stops is remarkably different (Figure 6.2). Still, another segmentation yielding the same M but with (g_1, d_1) and (g_2, d_4) perfectly temporally aligned would yield the same F-score. This indicator is therefore an expressive measure of *correctness*, but does not capture how *similar* matched stops are. Yet, accurate detection of temporal intervals is key to many analyses and also impacts the accuracy of the corresponding stop positions, as discussed later (§6.5). Therefore, we complement the F-score with the

$$\textit{S-score} = \frac{1}{|M|} \sum_{(g,d) \in M} \frac{g \cap d}{g \cup d}$$

The summation argument is the Jaccard index over the intervals g and d associated to true and estimated stops, respectively, a common way to capture their similarity. Its average over the matching set yields the S-score, whose value is in $(0, 1]$; it cannot be 0 because stops belong to the matching set and is 1 only when they are perfectly aligned.

6.4 Dataset Acquisition and Characteristics

Before applying the proposed metric to configure and compare segmentation techniques on our dataset, we describe the methodology used for collecting it and characterize its content.

6.4.1 Collecting the Dataset

UWB localization system. We use Talla [4], the time-difference-of-arrival (TDoA) localization system introduced in Chapter 5. TDoA can support many users as a *single* UWB packet broadcast by the mobile tag enables position computation by the fixed localization anchors in the target area, based on their different packet time of arrival induced by signal propagation along different distances. However, positioning accuracy depends on the tight time synchronization of anchors, typically achieved via dedicated and costly wired infrastructure. Talla can localize many mobile tags over large-scale areas spanning several anchors and with wireless-only time synchronization, all highly desirable features for adoption by the museum.

All nodes follow a common TDMA schedule. Anchors exchange beacon packets whose period (*epoch*) is key to time synchronization and thus positioning accuracy; beacons are also received by nearby tags, ensuring they are time-aligned. Each tag owns at least one slot where it broadcasts a location packet, whose time of arrival is timestamped at all anchors. These timestamps, along with the reference ones acquired during synchronization, are used by a localization server to estimate the tag position via a TDoA least-squares solver.

Improving position estimates via Kalman filters. UWB measurements are affected by errors, yielding noisy trajectories commonly "smoothed" via Kalman filters [39].

Noise exists both when the tag moves and when stationary; Kalman filters are usually optimized for either case. This clashes with stop-move detection, which requires efficient noise reduction in both cases *and* a fast switch between them to accurately determine stop start/end times. Therefore, we combine two Unscented Kalman filters (UKF) representing the tag mode (stopped or moving) under the framework of Interacting Multiple Models (IMM) [132]. The output position is a linear combination of both filter estimates, weighted by the probability of each filter to match the current tag behavior.

Hereafter, we refer to the trajectories output by Talla as *raw* and to those post-processed via IMM-UKF as *filtered*. Figure 6.4 exemplifies their difference, whose impact on stop positions (§6.4.2) and segmentations (§6.5) we analyze later.

System configuration and target area. User tags are battery-powered and anchors mains-powered; both are DWM1001 nodes by DecaWave (now Qorvo) equipped with the popular DW1000 UWB radio [70]. Each anchor is connected via USB to a Raspberry Pi, relaying TDoA data to the localization server. The UWB anchors are deployed on the ceiling of a 25×15 m^2 area (Figure 6.1). Those on the perimeter ensure low geometric dilution of precision (GDOP) and thus high positioning accuracy. The two near the center improve position diversity, mitigating the impact of radio signal occlusions.

We configured Talla with 250 ms epochs (4 Hz time synchronization) and tags broadcasting 3 times/epoch (12 Hz position update rate), and the UWB radio with the recommended channel 5, a 64 MHz pulse repetition frequency (PRF) and a 128 µs preamble.

Data collection methodology and ground truth. The target area contains a large

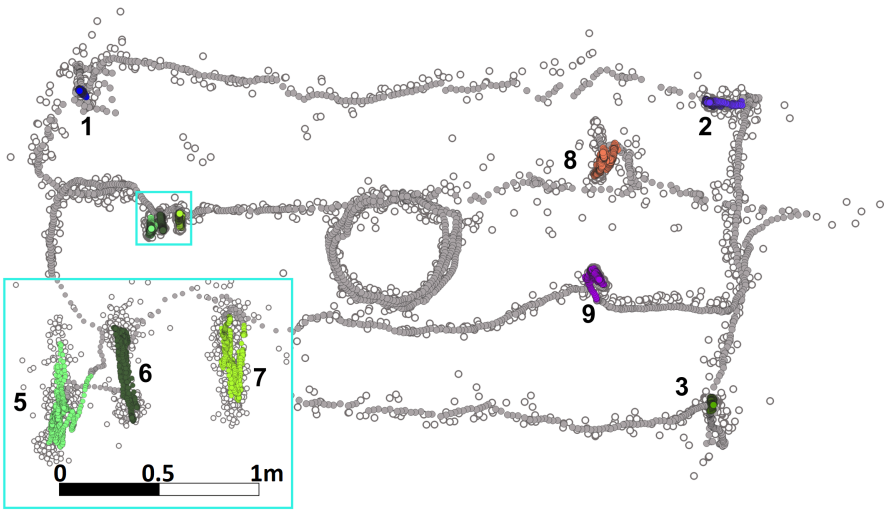


Figure 6.4: Raw (white) vs. filtered (gray) trajectory. The colored points in the latter fall in ground-truth stop time intervals.

globe surrounded by 6 tables hosting many small-size exhibits, of which 44 were visited. Members of the research team emulate the behavior of visitors by repeatedly moving in the area then stopping in front of some exhibit. Each user wears a necklace with an UWB tag on the chest, a natural option for a real use. The position of the tag is the one actually recorded by the UWB localization system.

Collecting reliable ground truth is challenged by mobility. We obtained accurate *spatial* data by placing floor stickers at all point of interests marking user stops (Figure 6.1) and acquiring with a laser meter their position (hereafter, POI). As for *temporal* ground truth, a smartphone application enables each user to record arrival/departure times for each POI by toggling a button. We synchronized the smartphone and Talla time, obtaining a common time reference for ground truth and UWB trajectories. Moreover, we placed 2 tripod-mounted cameras with 180° angle on opposite sides, covering the entire area, whose videos enabled validation of the smartphone data.

Dataset content. An UWB trajectory, the input for segmentation, contains units (§6.1) in the form $((x, y), t)$. For each timestamp t we collect both raw and filtered (x, y) positions (Figure 6.4). The ground-truth segmentation is represented by a sequence of stops (ID, t_{arr}, t_{dep}), containing the arrival/departure times to/from the POI with identifier *ID*, separately associated to its ground-truth position. This unambiguously identifies the stop location and accounts for recurrent visits. We collected 9 trajectories of similar duration (∼11 mins) for a total of 70090 units over 100.03 mins. The number of stops differs across trajectories, from 11 to 29, for a total of 209 stops.

6.4.2 Characterizing the Dataset

Spatio-temporal characteristics. Figure 6.6a shows *temporal* features via the cumulative distribution function (CDF) of stop durations in the ground truth. The chart substantiates the claim that our dataset contains very short stops, with a median of 12.4 s. The red line denotes the threshold below which durations become irrelevant for the application (§6.1), set to ρ=10 s based on requirements by the museum curators. The 10 stops (<5%) below it should not be detected, leaving 199 true stops as the expected ideal segmentation output (§6.2).

As for *spatial* characteristics, exhibits (POIs, §6.4.1) are physically very close (Figure 6.1). Figure 6.6b shows that 80.6% of adjacent POIs are within 1 m, with a maximum of 1.59 m. Again, this challenging dataset demands *high spatial resolution* in discerning stops when segmenting trajectories. Still, the distance between *consecutive stops within a trajectory* varies significantly (POI-POI line, Figure 6.6c) as exhibits are not necessarily visited in order. Figure 6.5 shows the filtered trajectory in Figure 6.4 with an extra time dimension. The user mimicking visitors mixes short strides to adjacent exhibits with longer ones (e.g., to join friends or avoid crowds), including one around the central globe (e.g., to observe it from all angles).

Ground-truth stops vs. estimated UWB centroids. Figure 6.5 illustrates another key point: each UWB trajectory contains *several* positions (in color) for a stop, i.e., falling inside the interval $[t_{arr}, t_{dep}]$ whose ground-truth value is reliably determined

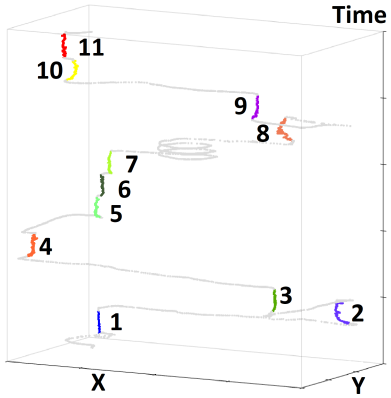
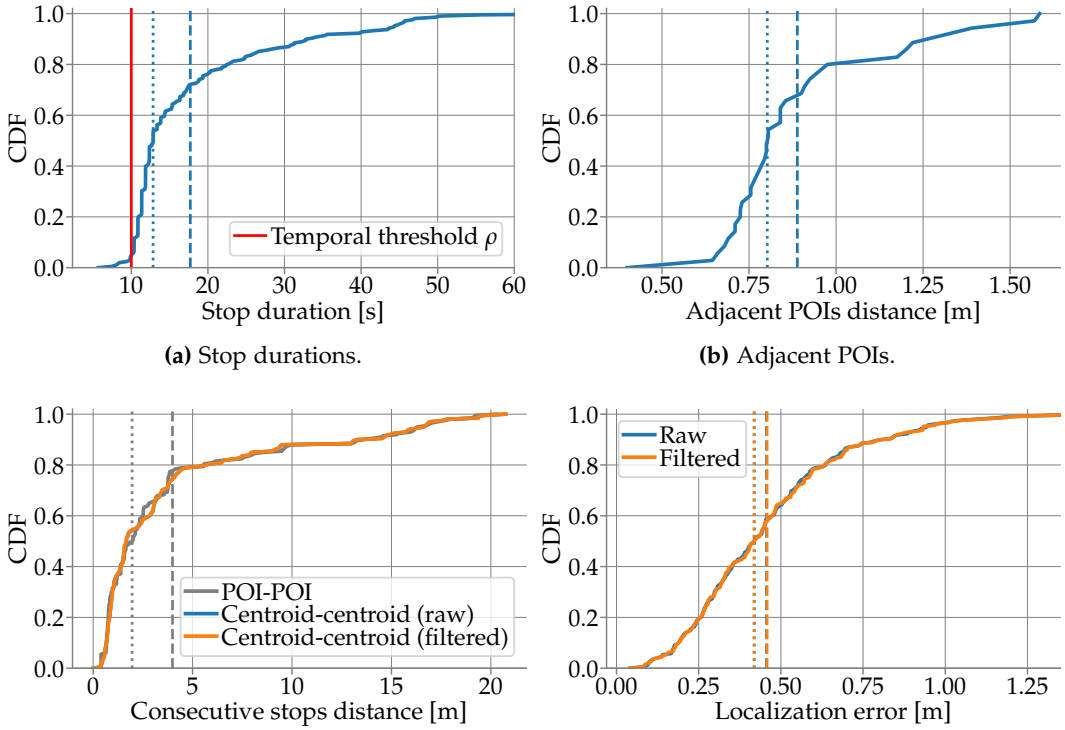


Figure 6.5: Spatio-temporal view of Figure 6.4 (filtered).



(a) Stop durations.

(b) Adjacent POIs.

(c) Consecutive stops.

(d) Stops: Estimated vs. POI.

Figure 6.6: Spatio-temporal characteristics of the dataset. Dashed lines represent the mean and dotted lines the median.

via smartphone and cameras. Ideally, the *centroid* of these UWB positions for a POI matches exactly the ground-truth one; in practice, this is not the case. The POI is very accurately measured with a laser meter in a fixed position; the positions yielding the centroid are measured for a moving tag and with larger UWB errors. Their main source is the user body, creating non-line-of-sight (NLOS) between the tag on the chest and the anchors behind the back. This is crucial when one of them is the main time reference; manually changing the latter when in NLOS reduces the mean positioning error by 25%. NLOS mitigation techniques, an active topic of

research [110, 111, 112, 113], could be incorporated in Talla and yield improvements; however, it is beyond the scope of this chapter.

Nevertheless, the error between the POI ground-truth and centroid positions (Figure 6.6d) remains sub-meter in 96.2% of the cases, i.e., significantly better than techniques based on WiFi and BLE, plagued by errors of several meters [39]. For both raw and filtered trajectories, the median and mean error are 42 cm and 46 cm, respectively; the commonly-used 75^{th} percentile is 57 cm. Interestingly, these metrics are within few percents, i.e., *the smoothing induced by Kalman filters does not affect the position of the UWB stop centroid*.

Trajectory structure: A key observation. Figure 6.6c compares the distance between consecutive stops computed using ground-truth POI vs. UWB centroids; their difference is negligible, despite the errors affecting the latter and the close placement of exhibits (Figure 6.6b). Put differently, *the UWB positioning error does not modify the structure of trajectories*; the sequence of stops contained in each trajectory is essentially the same regardless of whether we express it via ground-truth positions or estimated UWB centroids. This fact has important implications in deriving our findings, as discussed next.

6.5 Findings

We study the quality of reference segmentation techniques (§6.2) against the yardstick of our novel metric (§6.3) using our dataset (§6.4) as the means to distill quantitative findings.

What to compare? The ground truth in our dataset consists of stops (ID, t_{arr}, t_{dep}) where the times of arrival/departure from a known POI are accurately determined. However, segmentation techniques operate on UWB trajectories whose positions are less accurate than ground-truth POIs, with an error depending on the specific UWB localization system used.

Figure 6.7 illustrates the methodological problem. The time interval $[t_{arr}, t_{dep}]$ spent at a POI is known precisely from ground truth. A segmentation estimating it perfectly

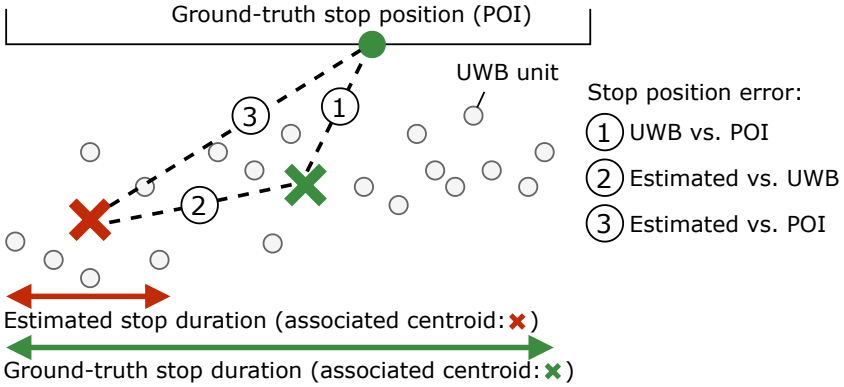


Figure 6.7: UWB trajectories, ground-truth vs. estimated stops.

is still affected by spatial error: the distance (#1) between the centroid of units in $[t_{arr}, t_{dep}]$ and the POI. However, real segmentations introduce errors w.r.t. $[t_{arr}, t_{dep}]$ and thus the centroid computed over it, causing additional spatial error (#2).

Our metric focuses on the *temporal* dimension by matching estimated and ground-truth intervals and *directly* associating them to the *ID* of true POI positions. This may seem to neglect the spatial dimension of segmentation quality. However, we observed (§6.4.2) that UWB trajectories and ground-truth ones have the same spatial structure; the temporal structure is therefore key. Moreover, our choice sharply decouples the quality of segmentation from the one of localization; separating the errors they induce would be impossible otherwise.

Consequently, we first use our metric to compare the temporal errors of segmentations. Then, leveraging the above decoupling, we study their impact on both the position of the estimated centroid vs. the UWB one in the dataset *and* vs. the POI position. This accounts for both time and space, ultimately providing precious information for practical use.

Novel metric: Is it worth? We confirm the higher expressiveness of our stop-centric metric by observing that *segmentations with the same unit-centric quality can have different stop-centric quality, and vice versa.* For illustration only, we focus on SeqScan segmentations of sample trajectories.

Figure 6.8 shows two segmentations of the same trajectory. The unit-centric metric assigns them the same F-score $= 0.90$; yet, the spatio-temporal maps show that they are very different. Segmentation $\langle 15, 24 \rangle$ correctly detects all 28 true stops. Instead, $\langle 15, 12 \rangle$ lumps 8 distinct true stops into 2 large estimated ones, representing incorrectly the user behavior. The unit-centric metric is oblivious to *structure*, as it considers only whether individual units belong to *any* stop. Conversely, our stop-centric metric accounts for the 6 false negatives in the second segmentation with a lower F-score than the first.

Figure 6.9 illustrates the dual situation on another trajectory. The unit-based metric assigns a higher F-score to $\langle 20, 36 \rangle$; one could infer it detects *more* stops than $\langle 10, 36 \rangle$. Instead, both detect 21 true stops with 1 FN; $\langle 10, 36 \rangle$ actually yields more accurate duration estimates, as shown for one sample stop in the spatio-temporal maps. These aspects are faithfully captured by our metric via the F-score and S-score, respectively.

This higher expressiveness has practical implications. For instance, in our museum context, some analyses may focus on *how many* exhibits are visited, others on *how long* each visit is. Our metric clearly distinguishes the two aspects, guiding the choice of the most appropriate technique and/or configuration.

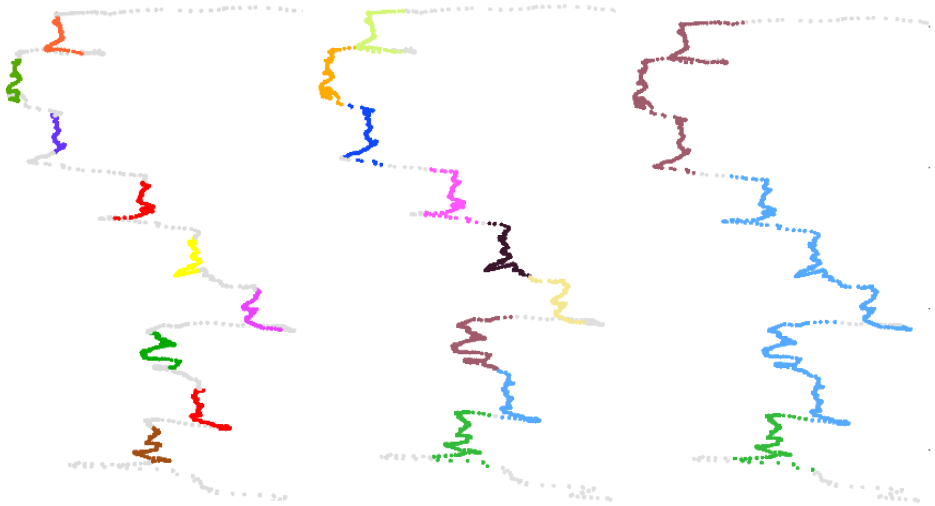
Based on all these considerations, hereafter we report only the results obtained with our proposed stop-centric metric.

Segmentation: Which parameters for what quality? We ascertained the impact on quality of several configurations for each technique. Table 6.1 shows results over the entire dataset for filtered trajectories. The highlighted best configurations are those with highest F-score and, when equal, highest S-score, e.g., as in $\langle 10, 12 \rangle$ and $\langle 15, 24 \rangle$

for SeqScan. Again, alternative criteria striking different quality tradeoffs are possible.

All methods yield good quality. KBV has the lowest one and is the most sensitive to its θ parameter; yet, it is the cheapest computationally (§6.2). At the other extreme, SeqScan yields the highest quality and its two-parameter configuration increases flex-

	Stop-centric					Unit-centric			
ϵ, N	TP	FP	FN	F-score	S-score	TP	FP	FN	F-score
15, 24	28	0	0	1.00	0.79	4953	932	80	0.90
15, 12	22	0	6	0.88	0.74	5013	1087	20	0.90



(a) Ground truth. **(b)** $\langle 15, 24 \rangle$. **(c)** $\langle 15, 12 \rangle$.

Figure 6.8: *Different* stop properties, *same* unit quality.

	Stop-centric					Unit-centric			
ϵ, N	TP	FP	FN	F-score	S-score	TP	FP	FN	F-score
10, 36	21	0	1	0.98	0.84	4348	483	1228	0.84
20, 36	21	0	1	0.98	0.83	5375	718	201	0.92

(a) Ground truth. **(b)** $\langle 10, 36 \rangle$. **(c)** $\langle 20, 36 \rangle$.

Figure 6.9: *Similar* stop properties, *different* unit quality.

ibility. Table 6.1 also reports the unit-centric metric, confirming its lower expressiveness. This is evident for KBV, whose highest F-score is obtained with θ=100 that 1. detects only 123 out of 199 true stops, yet 2. has nearly the same F-score of the best SPD configuration, detecting 189.

Raw vs. filtered trajectories: Does it matter? Filtered trajectories reduce spatial jitter vs. raw ones (Figure 6.4) but also induce the same stop-move structure (Figure 6.6); it is unclear whether and how they affect segmentation quality. Therefore, we performed for raw trajectories the same parameter exploration of Table 6.1, except for KBV. The best configurations for SPD (δ=80) and SeqScan (ϵ=15, N=24) respectively detect 8 and 10 fewer TP with an increase in FN, despite exploiting higher values of both distance (δ, ϵ) and number of points (N) to account for the higher position dispersion. This confirms that filtered trajectories yield higher quality, although the difference is not dramatic, as shown also by the other metrics (Tables 6.2 and 6.3).

What is the nature of false positives/negatives? The F-score of our stop-centric metric offers further insights on quality by expressing the *type* of false detections (§6.3). The analysis in Table 6.1 shows that SPD detects nearly as many true stops as SeqScan (more, on raw trajectories) but with more false positives, lowering precision and F-score. Table 6.2 now clearly shows that the culprit are split stops, a known weakness of the method. Moreover, all techniques are equally sensitive to stops with duration shorter than ρ=10 s; interestingly, this is the main source of mis-detection in SeqScan. Finally, fake stops are surprisingly rare, even absent with KBV.

Dually, missing stops are the main source of FN for all techniques. SeqScan and SPD achieve similar results; the latter is more sensitive to spatial resolution. A smaller δ does not affect split and merge stops but increases missing ones; with δ=20 (not shown), they become the only FN source. In contrast, the two-parameter structure of SeqScan achieves high quality with similar (or even lower) spatial resolution ϵ, slightly increasing merged stops in other less-performant configurations (not shown). Finally, the many missing stops in KBV are due to its reliance on velocity rather than distance, frequently changing around the threshold θ. This parameter crucially affects the nature of FN, dominant in KBV (Table 6.1); a value of 100 cm/s yields a majority (40) of merged stops, while 10 cm/s yields all missing stops.

How temporal errors affect spatial ones? We defined the S-score as a concise indicator of the temporal overlapping between true stops and estimated ones (TP). However, it does not account for the *absolute* error in the temporal alignment between the estimated and true stops. Here, we analyze this aspect along with the impact it bears on spatial error (Figure 6.7).

Given an estimated segment $[t_1, t_2]$ and a ground-truth one $[t_a, t_b]$ of duration t_d=t_2−t_1 and t_g=t_b−t_a, we consider the errors in duration Δt=t_d−t_g, start t_{start}=t_1−t_a, and end t_{end}=t_2−t_b. Moreover, we consider the corresponding spatial error Δs=$|p_d - p_g|$, i.e., the (absolute) distance (#2 in Figure 6.7) between the centroids p_d and p_g of UWB positions falling in t_d and t_g. Table 6.3 reports their mean μ and standard deviation σ in the best configurations; Figure 6.10 shows the CDFs of Δt and Δs of filtered trajectories only, due to space limitations.

Table 6.1: Exploring parameters for filtered trajectories; δ and ϵ are in cm, θ in cm/s.

	Stop-centric							Unit-centric
	TP	**FP**	**FN**	**Precision**	**Recall**	**F-score**	**S-score**	**F-score**
δ	**SPD**							
20	116	17	83	0.872	0.583	0.699	0.714	0.630
30	161	16	38	0.910	0.809	0.856	0.772	0.777
40	175	19	24	0.902	0.879	0.891	0.790	0.822
50	181	19	18	0.905	0.910	0.907	0.797	0.835
60	189	18	10	0.913	0.950	0.931	0.778	0.854
70	187	21	12	0.899	0.940	0.919	0.757	0.850
80	187	18	12	0.912	0.940	0.926	0.749	0.850
90	183	17	16	0.915	0.920	0.917	0.730	0.849
ϵ, N	**SeqScan**							
10, 12	186	8	13	0.959	0.935	0.947	0.810	0.871
15, 12	176	11	23	0.941	0.884	0.912	0.774	0.872
20, 12	165	13	34	0.927	0.829	0.875	0.741	0.869
10, 24	176	8	23	0.957	0.884	0.919	0.815	0.842
15, 24	187	9	12	0.954	0.940	0.947	0.796	0.870
20, 24	177	11	22	0.941	0.889	0.915	0.770	0.871
10, 36	166	6	33	0.965	0.834	0.895	0.813	0.824
15, 36	183	8	16	0.958	0.920	0.938	0.803	0.859
20, 36	178	10	21	0.947	0.894	0.920	0.777	0.869
θ	**KBV**							
10	31	4	168	0.886	0.156	0.265	0.594	0.42
20	110	6	89	0.948	0.553	0.698	0.729	0.658
30	154	4	45	0.975	0.774	0.863	0.766	0.719
40	168	11	31	0.939	0.844	0.889	0.781	0.801
50	175	10	24	0.946	0.879	0.911	0.781	0.831
60	168	10	31	0.944	0.844	0.891	0.751	0.844
70	162	9	37	0.947	0.814	0.876	0.724	0.846
80	142	7	57	0.953	0.714	0.816	0.684	0.847
90	129	9	70	0.935	0.648	0.766	0.650	0.851
100	123	11	76	0.918	0.618	0.739	0.629	0.855

All techniques perform well, with errors of few seconds and centimeters. KBV is the most accurate temporally, with a mean error μ=1.72 s. Yet, its mean spatial error Δs is the highest among filtered trajectories; it is very near to SPD, whose median is however significantly worse. At the other extreme, SeqScan yields the worst duration estimates; μ is nearly twice w.r.t. KBV, although the absolute difference is $<$2 s. Nevertheless, it is the most accurate spatially—a counterintuitive result explained by observing that 1. SeqScan is robust to outliers by design, intrinsically reducing spatial noise 2. temporal precision (σ) is the highest 3. t_{start} is *under*estimated and t_{end} *over*estimated, both in median (not shown) and mean by nearly the same amount, which tends to center the true stop inside the estimated one, reducing the distance

Table 6.2: Segmentation techniques at a glance: Stop-centric metric and additional insights on the nature of false detection.

Technique	Dataset	Main metric		TP	Nature of false detections				
		F-score	S-score		FP			FN	
					split	short	fake	merged	missing
SPD	filtered	0.931	0.778	189	10	7	1	3	7
	raw	0.903	0.764	181	13	7	1	5	13
SeqScan	filtered	0.947	0.810	186	1	6	1	4	9
	raw	0.921	0.779	176	1	5	1	8	15
KBV	filtered	0.911	0.781	175	5	5	0	7	17

Table 6.3: Comparison of spatio-temporal errors for the various segmentation techniques.

Technique	Dataset	Spatio-temporal errors (in TP)							
		t_{start} (s)		t_{end} (s)		Δt (s)		Δs (cm)	
		μ	σ	μ	σ	μ	σ	μ	σ
SPD	filtered	-0.83	4.60	1.16	3.61	2.00	6.65	4.11	6.12
	raw	-1.01	3.82	1.33	5.04	2.35	7.25	4.55	6.71
SeqScan	filtered	-1.64	2.53	1.78	4.05	3.43	4.93	3.12	4.45
	raw	-3.03	3.40	1.98	4.82	5.02	6.41	6.24	33.1
KBV	filtered	0.26	5.21	1.99	4.96	1.72	8.01	4.26	8.42

Table 6.4: Stop-POI association for true positives, and overall performance for all true stops.

Technique	Dataset	Correct POI (%)	
		TP only	overall
SPD	filtered	93.1	88.4
	raw	92.8	84.4
SeqScan	filtered	95.2	88.9
	raw	92.6	81.9
KBV	filtered	92.0	80.9

between the corresponding centroids (Figure 6.7).

Figure 6.10a also shows that SPD often severely underestimates stop duration, likely the culprit for the many stop splits (Table 6.2). Nevertheless, its performance in terms of Δt and Δs does not change significantly when moving from filtered to raw trajectories. This is not the case for SeqScan, whose metrics for the latter (Table 6.3)

are nonetheless heavily affected by a *single* outlier, caused by the merging of distant stops, whose removal yields μ=3.82 cm and σ=7.74 cm for Δs. Anyway, this is in line with false detections (Table 6.2); while SPD is prone to stop splitting, SeqScan is to merging.

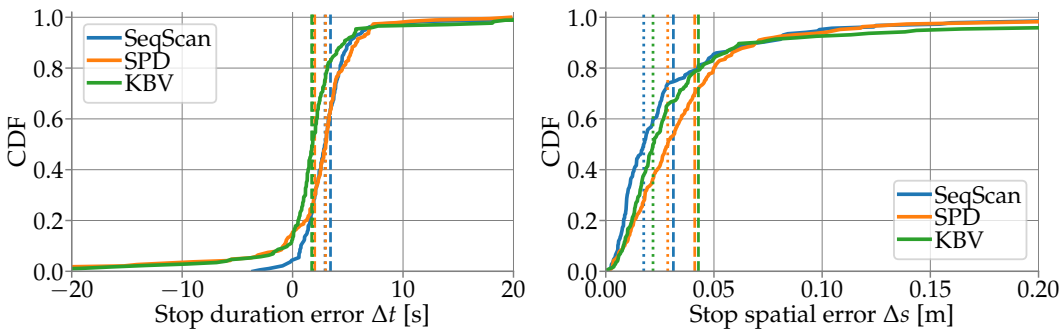
Can estimated stops be correctly associated to POIs? Among the distances in Figure 6.7, we analyzed #1 in our dataset (Figure 6.6d) and #2 by reporting Δs (Table 6.3); we now investigate their combination, #3. Its value is not very informative, given that Δs is very small and so is the difference between #1 and #3; for SPD, the worst-case (filtered), is 46 cm median and 52 cm mean, only few centimeters more than in Fig 6.6d.

Instead, the crucial question is: *Can we correctly identify the POI visited by the user via the estimated centroid, i.e., without ground truth?* This of practical relevance, as it is how a real system would work. Intuitively, the association can consist simply of determining the POI *closest* to the estimated stop.

This is challenging in our setup with 1. POIs close to each other (Figure 6.6b) and 2. non-negligible UWB positioning error (Figure 6.6d). Still, Table 6.4 shows that stops can be accurately associated to POIs. Considering only TP, the POI closest to the estimated stop is correct in >92% of the cases, with a maximum of 95.2% for SeqScan. However, techniques differ in their ability to identify TP, reducing the fraction of overall correct associations, nonetheless always >80%. KBV is the worst, due to its lowest TP; for the same reason, SeqScan is the best at 88.9%, although its accuracy on raw trajectories degrades below SPD, reasserting the importance of filtering.

6.6 Related Work

Stop-move detection. Trajectory segmentation has been applied to GPS trajectories for a long time [133]. We considered techniques (§6.2) representative of two main classes: *criteria-based* define stops based on, e.g., distance, time duration, velocity, as in SPD and KBV, but also [134, 133]; *cluster-based* include SeqScan and, e.g., [135, 136, 137]. Others are based on statistical models [128] or do not even rely on segmen-



(a) Stop duration error.

(b) Stop spatial error.

Figure 6.10: Duration and spatial error for filtered trajectories.

tation [138]. Our focus is not on exhaustive comparison, rather on *quantifying* the quality attainable w.r.t. ground truth when applying stop-move techniques to UWB, and contributing a novel metric better eliciting tradeoffs and challenges.

Ground truth and metrics. Evaluation against ground truth is hindered by the complexity induced by mobile targets (e.g., a migrating animal). Validation is often only qualitative, e.g., through visual inspection of trajectories, or via simple metrics like the number of stops [135, 139]. This may suffice over large-scale areas with well-separated stops, but is difficult in our setup where stops are close both in space and time. The recent unit-centric approach [128] offers a quantitative metric, but without a real-world validation.

Our stop-centric metric (§6.3) is significantly more expressive, as confirmed by the analysis of our dataset gathered in a real museum and with accurately acquired ground truth. The use of F-score on stops instead of units was also proposed in [137, 136]. However, these works do not state how a true stop is measured, likely demanding it to qualitative considerations. Instead, we provide definitions and methods enabling both automated quantitative analysis and interpretation of false detections, along with a measure of stop similarity, reuniting all relevant dimensions in a single methodological framework.

Stop-move and museums. Museums are a natural application for stop-move detection, crucial to understanding visitor behavior. However, reported experiences are based on Bluetooth, whose poor positioning accuracy [39] forces coarser trajectory models whose units are entry/exit or proximity events instead of positions. In [124], nodes in key areas of the Louvre enabled analysis of the overall stay in the museum (hours) and frequency of visits to areas. Similar experiences determined the stay in a room [127] or "hotspots" and other macro-level indicators [125]. These approaches extract spatio-temporal features from the Bluetooth signal, increasing complexity and reducing accuracy. In contrast, we showed quantitatively how the higher accuracy of UWB directly translates in a much greater spatio-temporal resolution in discriminating stops.

6.7 Conclusions

The high spatio-temporal accuracy of UWB localization intuitively enables fine-grained detection of stop-move patterns, key to many applications. Yet, this opportunity has not been studied, let apart quantitatively and experimentally. This is our goal, exploiting a museum deployment with accurate ground truth, also rare in the literature. The findings, albeit not directly generalizable, concretely inform about the quality attainable in practice in a real-world setting.

We define a novel, expressive metric relating estimated and true stops, enabling the configuration and comparison of segmentation techniques originally targeting coarser-grained scenarios. We show that, once applied to UWB trajectories, they induce only small spatio-temporal errors of few centimeters and seconds. Therefore, we identify the UWB localization system, not segmentation, as the main source of spatial error, likely mitigated by continuous progress in UWB research. Neverthe-

less, estimated stops are correctly associated to true ones in the vast majority of cases, enabling fine-grained UWB-based stop-move detection in this and other practical contexts.

Acknowledgements. We are deeply grateful to Michele Lanzingher and Vittorio Cozzio for making this study possible at MUSE. We thank A. Bacchiega, M. Fenu, A. Giovannone, T. Istomin, and D. Molteni for their help on various technical and experimental issues.

7

Self-Organizing Neighbor Discovery and Ranging for Battery-powered UWB Anchors

GNSS services have enabled autonomous vehicle navigation in large areas, providing near-ubiquitous localization without the need for a dedicated infrastructure. Unfortunately, satellite signals cannot be detected in indoor environments, and signal quality degrades significantly when obstacles, such as buildings or thick foliage, interrupt the line of sight with the user to be localized. Because of this, navigation in indoor areas and agricultural fields must rely on different localization solutions. Among the approaches based on radio frequency, UWB has steadily gained attention due to its decimeter-level accuracy in distance estimation. However, UWB systems are hindered by the high deployment costs that come with the infrastructure of anchors. While it has been shown that wired synchronization is not necessary in UWB systems [106, 4], cabling operations are still required to connect anchors to the power grid, due to the high energy consumption of UWB chips.

To reduce energy consumption, and pave the way for battery-powered infrastructures, the uptime of anchors must necessarily be kept to a minimum. To this end, we observe that anchors could enter low-power mode ("deep sleep") when no user to be localized is around. This is crucial when covering large operational areas, since anchors may be isolated for most of the time. On the other hand, this approach requires an energy-efficient mechanism for anchors and users to discover the presence of each other and resume localization. One possible solution is to maintain network-wide synchronization to facilitate communication between users and anchors. However, this translates to a constant energy drain even when users are far from anchors, in addition to the energy cost for neighbor discovery.

In this chapter we present Sonar (Self-Organizing Neighbor discovery And Rang-

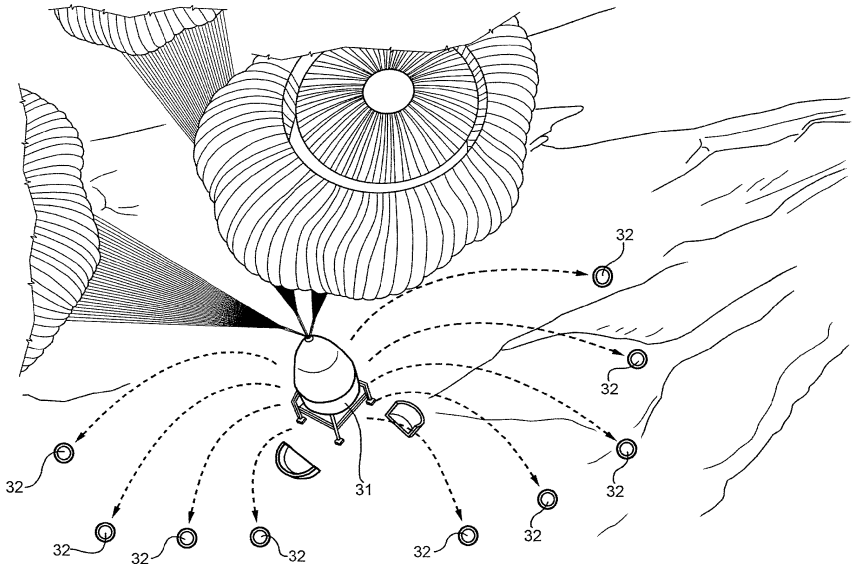


Figure 7.1: A lander ejects the UWB anchors that will make up the localization infrastructure. From [140].

ing), an energy-efficient UWB protocol for ranging-based localization, extending the lifetime of battery-powered infrastructures. Sonar finds application in all scenarios that require a permanent deployment where mains-powered anchors cannot be installed. For example, the proposed system can support autonomous navigation in agricultural fields. Roaming vehicles can perform ranging with Sonar anchors to estimate their own position and follow a predetermined path. Navigation is the use case that first motivated this research, albeit not in the context of agriculture. We designed Sonar in collaboration with Thales Alenia Space Italia, as part of the effort to enable localization in unknown or difficult environments, and specifically for unexplored planets, like Mars. The overall approach has been described in a patent filed by the company [140], and implemented in a prototype replicating the various actors of the system [141]. In the envisioned approach, the lander ejects UWB anchors before reaching the planet surface (as in Figure 7.1). After an initialization phase, in which the anchors perform ranging among themselves, results are reported to the lander. The patent describes the methodology to construct the coordinate system from the estimated distances between anchors. The lander is assigned the $(0, 0)$ coordinates, then a nearby anchor at distance d from the lander is chosen to fix the x-axis, with coordinates $(d, 0)$. The coordinates of the remaining anchors are then computed one by one based on their distance from neighbors whose coordinates have already been fixed. Finally, the lander disseminates all the computed anchor positions. At that point, anchors can start to interact with the rover(s) used to explore the planet, enabling localization. The question is then how to minimize anchor consumption in this localization phase, and Sonar is our proposed solution.

The main contribution of this chapter is the design of a protocol that extends the lifetime of battery-powered anchors by relying on user orchestration and synchronizing connected users and anchors only when needed. This is done with no communica-

tion overhead, as a byproduct of the main operations of the system, i.e., neighbor discovery and ranging. In contrast with existing solutions, Sonar builds on the assumption that users (e.g., the roaming vehicles) have a large battery capacity. Since the energy consumption of the radio is negligible compared to that of the motor, the user radio in Sonar is always active, listening to incoming transmissions from anchors advertising their presence. As the name of the protocol suggests, multiple users in Sonar can organize a shared schedule for transmissions, thereby avoiding collisions and needless competition for the same anchors, dynamically accommodating the needs of multiple users when they co-exist in the same localization area.

We evaluate Sonar in a multi-hop UWB testbed with multiple nodes acting as users meeting and departing to study the responsiveness of the protocol in harsh conditions. Our analysis confirms that Sonar allows users to quickly discover nearby anchors and establish a shared ranging schedule, while anchors remain in power-saving mode most of the time. The remainder of this chapter is structured as follows. In the next section, we provide an overview of state-of-the-art localization systems with a focus on the energy consumption of their anchors. We describe next, in Section §7.2, the core properties of the proposed protocol, and in §7.3 the design principles that allow Sonar to overcome the limitations of traditional localization systems with always-on anchors. After highlighting relevant implementation details in §7.4, we report in §7.5 the estimation of the energy consumption of anchors and their battery duration. In §7.6 we discuss the results of our evaluation, analyzing the performance of Sonar in terms of time to synchronize, ranging reliability and ranging accuracy. Finally, we share our conclusions and outlook in §7.7.

7.1 Related Work

UWB localization systems are typically designed to reduce the energy consumption of the user, i.e., the UWB tag attached to the target to localize, assuming anchors are part of a wired infrastructure and connected to the power grid. This is often the case for TDoA approaches, that exploit the single-message localization approach to maximize the number of supported users [106, 4]. Even though TDoA requires only one UWB transmission, it is difficult to use in energy-efficient schemes. First, the reception timestamps of each anchors must be then transmitted to a single node for computation. The node must then take care of solving the TDoA optimization problem, correct timestamps for sub-ns synchronization, and potentially apply filters on data. Therefore, a remote server is typically in charge of handling TDoA data. In our use cases, this is impractical as each localization attempt would come with additional energy expenditure for data collection, quickly depleting the power supply. Recent works have introduced battery-powered anchors using two-way ranging. For small areas of co-located users, a TDMA network-wide schedule allows nodes to contend for medium access and range with nearby targets [142]. However, this does not easily scale for large areas, requires continuous synchronization and frequent slot negotiation. In [143], authors suggest that tag-equipped users orchestrate nearby anchors using a dual-radio approach, deferring all operations other than ranging to a low-power sub-GHz module. Nonetheless, this requires frequent anchor listening

activity and has a significant impact on anchor lifetime, with a constant consumption of 3.4 mA even when no ranging is performed. Instead of relying on TDMA, [52] adopts a dedicated MAC layer based on low-power listening. This increases the energy consumption of the tag, but dramatically extends anchor lifetime, especially when no tag is around. However, since both tags and anchors are designed to save energy, a long phase before all ranging exchanges is needed for discovery, contention resolution, and scheduling. Because of this, the approach is only suited for very low ranging rate, and not for localization applications supporting multiple users, as each requires multiple ranging exchanges for one position estimation. A clear trend emerges from recent works: we can stretch anchor lifetime by offloading operations to the tag. Unlike the previous protocols, in our target applications we can exploit always-on tags to minimize anchor operations and achieve unprecedented anchor efficiency.

7.2 Goals and Assumptions

Battery-powered anchors provide great flexibility when deploying the infrastructure. This flexibility does not come for free, but with its own set of challenges for protocol designers to tackle in the quest to preserve energy.

In 2D ranging-based localization, users typically perform distance estimation with three or more anchors with known coordinates to compute their own position. This is easy to do when anchors are always active, listening on the radio channel waiting for user requests. However, when the energy supply is limited, anchors must stay in low-power mode for most of the time, becoming unavailable for users. Therefore, a dedicated protocol must be designed for anchors to keep their radio on only when there are users nearby that want to perform ranging.

Users perform ranging exchanges with the anchors in range, e.g., to perform proximity detection or full-blown localization. However, if the area is large, nodes (users and anchors) are *not* necessarily in range of each other.

We introduce a system for the orchestration of neighbor discovery and ranging across multiple users interacting with battery-powered anchors. The system is designed to ensure a long deployment lifespan, stretching to years of operation. The protocol should be easily configurable to balance ranging rate, responsiveness of neighbor discovery, and energy consumption, while adapting dynamically based on the user needs.

Sonar achieves the aforementioned goals by implementing the following properties:

- **Aggressive duty cycling of anchors**. We first observe that anchors should limit energy consumption, entering low-power mode whenever possible. Two opportunities arise. On one hand, depending on the size of the operational area and the number of roaming users, it is likely that several anchors remain isolated (i.e., not in range of any user) for the vast majority of time; their operation can therefore be aggressively duty-cycled. On the other hand, even when one or more users are around, the anchor radio is put to sleep whenever the user is not interacting with it, e.g., while engaged with other anchors.

- **Adaptable neighbor discovery**. As users are moving, it is important that they rapidly discover the set of anchors that can be exploited for ranging. The period of network discovery depends on application and system requirements. Aiming for the lowest-latency discovery possible is unneeded in situations where users are slowly-moving, and therefore neighborhoods are mostly static.
- **Dynamic time allocation for multiple users**. When different users are operating in far-away, non-overlapping areas, each user can enjoy the entire communication bandwidth available for continuous, maximum-rate ranging. However, this is no longer possible when other users share the same medium. The system adapts to the contingent situation, e.g., dynamically throttling down the ranging rate of a user to make space for others or, on the contrary, throttling it up when they disappear and the user is alone again.
- **User-side orchestration**. One of the key aspects of Sonar is that the users, not the anchors, are responsible for ranging scheduling and inter-user arrangement. To make this possible, we assume the radio of users is always on. Indeed, the energy drain of the radio is orders of magnitude smaller than the one of, e.g., the engines required to enable the user movement.
- **No assumption on anchor connectivity**. In the scenario of planetary exploration, the lander ejecting the anchors has virtually no control over their final placement, which means that the network topology cannot be known in advance. In agricultural fields and industrial settings, obstacles are likely to temporarily impair or block communication. Considering the challenging environments we target, we need to avoid assumptions on the quality of radio links between anchors. Therefore, by relying on user-side orchestration, Sonar does not require the anchors to ever communicate with each other.

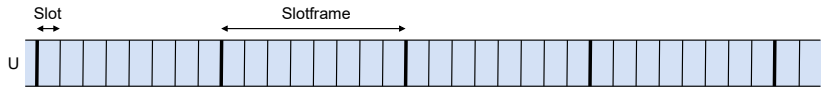
7.3 Sonar Design

We present the Sonar logic by first providing a general overview of the protocol, followed by the description of a single-user system, and finally introducing the changes required for multi-user mode.

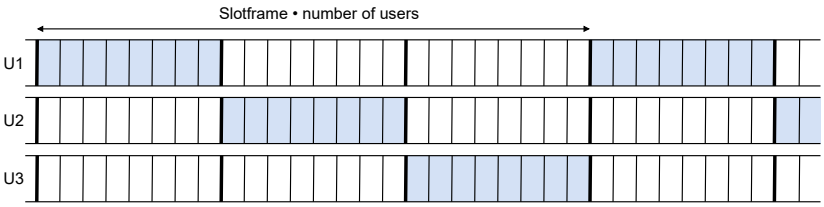
7.3.1 High-level Protocol Overview

The protocol is based on a TDMA structure: the user coordinates the interactions with its neighboring anchors in a time-slotted fashion. However, the protocol does not rely on a global time reference or a global slot schedule; slot scheduling and synchronization are performed *locally*, in the neighborhood of the user(s).

The protocol comprises two main techniques for low-power operation. First, the anchors that are far away from any user keep their radio off most of the time, probing the medium periodically with a beacon to discover users. Second, the anchors that are serving one or more users follow the TDMA schedule which defines active and passive slots for each anchor, thus letting them keep the radio off during time slots in which they are not involved. Users, instead, keep their radios always on to



(a) Single/isolated user.



(b) Multiple users.

Figure 7.2: Slotframe ownership for an isolated user and for 3 co-located users.

detect neighbor discovery beacons and to quickly coordinate the schedules among themselves once they overhear the activity of each other.

Users are able to self-organize without relying on any additional radio operation involving the infrastructure, adjusting the ownership of time slots to avoid interference during ranging. If a single user is operating in an area, it is allowed to utilize all available time slots, grouped into *slotframes* that repeat over time (Figure 7.2a). At every iteration, the user can select its preferred group of anchors to range with. When different users approach, they switch to multi-user mode, where slotframes controlled by each of them alternate without overlapping (Figure 7.2b).

7.3.2 Single-user Mode

Time is split into *slots* of the same duration T_{slot}, long enough to fit the ranging handshake used in the system, i.e., single-sided two-way ranging (SSTWR), that requires two transmissions. Similarly, bidirectional neighbor discovery exploits a single slot hosting the anchor beacon and the user reply. Several consecutive time slots are grouped in a slotframe of a fixed number of slots, which is a system-wide constant S, giving a slotframe duration $T_{sf} = S \cdot T_{slot}$. A configurable number of slots D at the end of the slotframe is reserved for neighbor discovery, leaving the first $S - D$ slots of a slotframe for ranging. Figure 7.3 shows an example of a slotframe.

Anchor-initiated neighbor discovery. Initially, anchors and the roaming user are not aware of each other. Anchors that are out of reach of any users (hereafter, *isolated anchors*) keep their radio off most of the time. To advertise their presence to the user, they periodically transmit a discovery beacon, called `ND-INIT`, in a randomly selected slot not used for ranging. The interval between neighbor discovery attempts, T_{ND}, is an important parameter of Sonar anchors, balancing energy consumption and discovery responsiveness. After transmitting `ND-INIT`, the anchor waits for the short time needed for the user to reply. The user replies with a `ND-RESP` message, which includes the Sonar header with synchronization information, and discovery is then confirmed to the user with the `ND-FINAL` message (Figure 7.3).

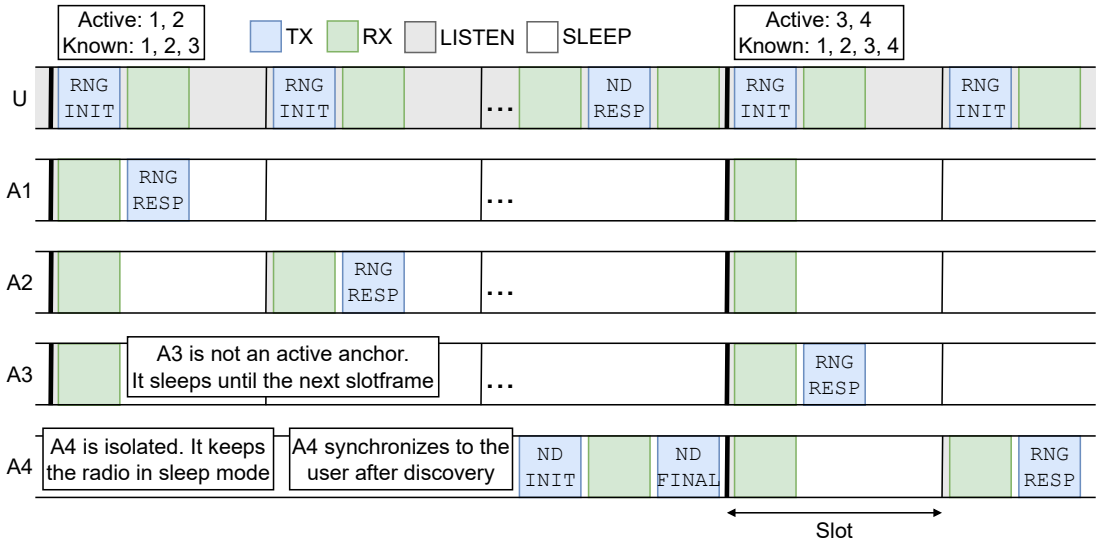


Figure 7.3: A slotframe schedule example, with one user, three known anchors and one to be discovered.

User-anchor synchronization. Anchors synchronize their slotframes to the one of the discovered user, adjusting their TDMA structure based on the content of `ND-RESP`. Synchronization information is expressed by the combination of slot number $s \in \{0, ..., S-1\}$ and time offset t_{offset} from slot start. Whenever the user sends a message, it embeds t_{offset} as the difference between transmission time and slot start time. When the anchor receives `ND-RESP` or `RNG-INIT` from the user, it timestamps the arrival of the message at t_{RX}, exploiting the high-frequency clock of the UWB radio chip. This timestamp is embedded in `RNG-RESP` for ranging, but is also used by the anchor for synchronization. Synchronization can simply be performed as $t_{slot} = t_{RX} - t_{offset}$, which gives the slot start in UWB anchor time. Exploiting the current slot number s, the anchor can also compute the start of the next slotframe as $t_{sf} = t_{slot} + (S - s) \cdot T_{slot}$.

Synchronization information also includes the current slotframe number f, necessary in the multi-user scenario §7.3.3 to assign slotframe ownership to the right user.

Ranging with discovered anchors. While neighbor discovery is initiated by anchors, ranging handshakes are always initiated by the user. If the user is aware of at least one nearby anchor, it sends ranging requests starting in the first slot of the slotframe. Onto the ranging requests, called `RNG-INIT`, users piggyback the same Sonar header as in `ND-RESP` messages, together with the scheduling information for the current slotframe. To receive the schedule, all time-synchronized anchors always wake up in the first slot of the slotframe and switch on the radio. Having received a `RNG-INIT` message, surrounding anchors learn the following:

1. the synchronization information in the header, to (re-)align to the advertised slotframe;
2. the list of *known anchors* that the user discovered;
3. the list of *active anchors* to be used for ranging in the current slotframe;

4. the slot assignment for the current slotframe, or *ranging schedule*: which anchor is expected to reply in which slot to perform the ranging exchange.

The first active anchor the user wants to range with replies directly to the message carrying the schedule in the first slot to complete the ranging exchange. Otherwise, if it is in the active list, it goes to sleep to save energy and sets a wake up in the appropriate slot, which depends on its position in the list. If an anchor is not active but is known, it is a *passive anchor* in the current slotframe. Passive anchors sleep until the beginning of the next slotframe to receive the new schedule. In the example in Figure 7.3, the user initiates a single-sided two-way ranging handshake with anchor 1 and simultaneously announces that 2 is going to be requested next in the current slotframe while 3 is known to the user but will not be requested. In this case, anchors 1 and 2 are active, while 3 is passive.

Interestingly, the described scheduling approach allows for an arbitrary interleaving of the interactions with anchors. For instance, in Figure 7.3 the user could decide to perform, in the same slotframe, several consecutive ranges with anchor 1 (e.g., to accumulate enough samples back-to-back) followed by multiple consecutive ranges with 2, or instead alternate between 1 and 2. Notably, nothing prevents users from selecting other users as ranging targets, instead of anchors.

7.3.3 Multi-user Mode

The previous overview is sufficient for the operations of a single Sonar user with any number of anchors. To support multiple users, the protocol must be extended to include inter-user discovery and synchronization.

A look at slotframes in multi-user mode. Whenever a user discovers the presence of others, it shifts from single-user to multi-user mode. Instead of using all available slotframes, each user can range in a subset of them, interleaving operations with the other peers (Figure 7.2b). Slotframe assignment can be performed exploiting the user ID and the current slotframe number f. For example, if four users are involved with IDs from 0 to 3, and the current slotframe number is 17, the owner is given by $(17 \bmod 4)$, i.e., user 1.

For slotframe division to be effective, of course, users must all agree on the current slotframe number and start time, but because there is no global synchronization in the system, this requires dedicated mechanisms. To synchronize and avoid collisions, Sonar users must first be able to detect each other.

Multi-user neighbor discovery. The first step is to extend neighbor discovery, which was previously only performed between a single user and the anchors. To ensure that an anchor discovers all surrounding users, the `ND-INIT` beacon it broadcasts includes which users are already known to the anchor. This keeps known users from replying, allowing the missing ones to be discovered. Indeed, all users not yet discovered immediately send back a `ND-RESP` message. While multiple overlapping replies could in principle cause a collision, one of them is likely to be decoded correctly by the anchor due to the properties of UWB concurrent transmissions (§2.4.4). Therefore, the anchor eventually discovers all users in range.

Now that anchors can discover multiple users, we must also enable users to discover their peers. In this case, the relevant neighborhood of users is not limited to other users in direct communication range with them. Rather, to avoid interference and needless competition, they must synchronize with the set of all users interacting with the same anchors. This can be accomplished by exploiting just ND and RNG messages. Relying on their always-on radios, users can learn about the presence of others either by:

- overhearing other user messages (RNG-INIT, ND-RESP),
- overhearing ranging replies from anchors to other users (RNG-RESP), or
- receiving a neighbor discovery message from an anchor (ND-INIT).

Once users discovered their peers, they must synchronize before they can arrange a shared schedule.

Inter-user synchronization. Like user discovery, synchronization is performed by exploiting the same messages used for ND and RNG and does not require additional transmissions. In multi-user all messages, not only those from the user (RNG-INIT and ND-RESP), carry the SONAR header with the synchronization information introduced in the single-user case. Indeed, while previously the only possible synchronization source was the one user, in multi-user mode anchors become synchronization forwarders themselves, bridging the gap between nearby users. As multiple synchronization sources are available, a mechanism is needed to disambiguate them and dictate which user should prevail, becoming the primary source the others align to. To this end, synchronization information is extended with the unique user ID indicating which user the sender of the message has already aligned its slotframes to. A synchronization structure is created, starting from the primary source, based on the natural order in IDs. Users and anchors exploit this new information to align to the same reference as the rest of their neighborhood, either accepting synchronization if the source ID is lower than their own current source, or rejecting it otherwise.

To illustrate why the reference node ID needs to be propagated together with the time reference, we give an intuition in Figure 7.4. When different users contend for the same anchors, they eventually learn about each other. Among these users, the one with the lowest ID will become the reference that others must align to, while distant users remain unaffected. The user group containing users 2 and 3 with the time reference bound to user 2 approaches and merges with another group, with users 1 and 4. After the merge, the whole group binds to user 1 as the new time reference, and operations can continue without disruption.

Finally, the time reference must also be attached to a freshness timestamp, revealing how long ago the reference user was heard. This is needed to discard references older than the one known by the receiving node and avoid loops in time propagation. Through these mechanisms, synchronization can be performed only when needed and quickly propagated to the set of co-located users.

Fast discovery and synchronization recovery. With multiple users meeting and departing, we see opportunities to speed up the process of user-anchor discovery, using

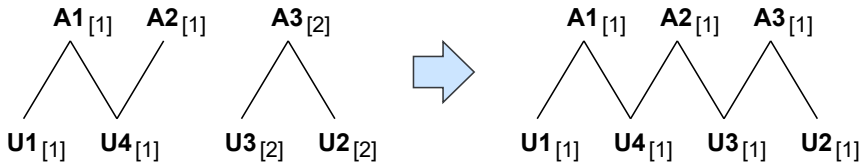
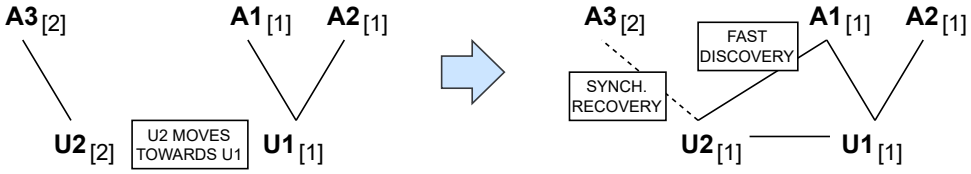
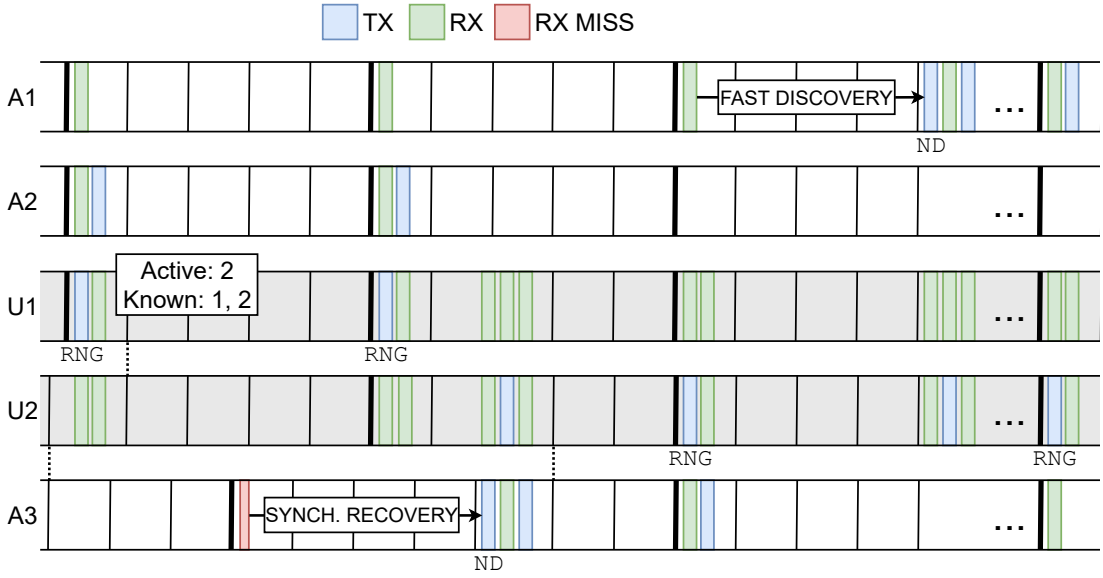


Figure 7.4: Time propagation when two user groups merge. Value in brackets is the ID of the time reference user. Synchronization across groups is performed through an anchor: after anchor 4 transmits a message overheard by user 3, the two groups quickly align to reference 1.



(a) Merging of two groups of nodes. User 2 enters in the range of user 1 and anchor 1. User 2 synchronizes to user 1, causing anchor 3 to become out-of-synch.



(b) Slotframe operations when users meet.

Figure 7.5: Fast discovery and synchronization recovery. Anchor 1 receives the schedule from user 2, but is not in the list of known anchors. Anchor 3 detects synchronization loss when no schedule is received at the beginning of the slotframe.

fast discovery. We observe that anchors may be synchronized to the user with lowest ID before being discovered by all the others. The anchor can detect this circumstance by inspecting the list of known anchors when they receive the schedule embedded in `RNG-INIT`. If the anchor cannot find itself in the list, it issues a `ND-INIT` in an appropriate slot of the current slotframe even if T_{ND} has not expired yet. The fast discovery mechanisms is particularly useful when a new user approaches anchors that are al-

ready synchronized to a user with lower ID, which the new user also aligns to.

On the other hand, ranging schedules may experience a transient inconsistency due to re-synchronization upon the arrival of a new user with a lower ID. All anchors discovering the newcomer synchronize to it, becoming unable to receive ranging requests from any other user. Or, equivalently, other users synchronize to the newcomer and become out-of-synch w.r.t. anchors. Instead of waiting for re-discovery, we tackle this issue with the *synchronization recovery* technique. When an anchor misses the `RNG-INIT` at the start of the slotframes, it knows which user the slotframe belongs to, based on the current slotframe number f. Therefore, the anchor can recover synchronization by issuing a `ND-INIT` message, similarly to fast discovery. However, in this case it first removes the lost user from the list of known users embedded in the message, to trigger the reply. Exploiting the synchronization information in `ND-RESP`, the anchor resumes normal operations.

Discovery and synchronization logic as a whole can be appreciated with an example involving the known messages of the system. In Figure 7.5, user 2 comes close to user 1 and synchronizes to it. This causes anchor 3 to become out-of-synch and miss the expected ranging schedule at the beginning of the next slotframe. Anchor 3 then removes user 2 from the list of known users and performs `ND` in the same slotframe, re-aligning with user 2 (synchronization recovery). After recovery all nodes are time-aligned. However, user 2 has yet to discover anchor 1. The anchor exploits the list of known anchors in the schedule from user 2. Since it is not in the list, it exploits the fast discovery mechanism and prepares the `ND` slot to advertise itself to user 2.

7.3.4 On the Responsiveness of Neighbor Discovery

There is clearly a tradeoff between discovery responsiveness and energy consumption, governed by the *frequency* with which the anchor transmits `ND-INIT` messages. The proper configuration of this parameter is application dependent; we provide some intuition about it based on the expected number of new anchors encountered by a user. How fast a new anchor should be discovered is a function of the velocity of the user and the density of anchors. We present here a simple model that computes the number of neighbors changed (appearing/disappearing) per second based on the simplifying assumptions of:

1. uniform distribution of anchors
2. constant velocity of the user
3. perfectly isotropic communication (unit disk graph model, UDG).

Figure 7.6 illustrates the main concept. Assuming a minimal meter distance l between anchors, the uniform density of anchors (anchors/m^2) is $m = 1/l^2$. The number of anchors in range of a given user is $n = mU$, where $U = \pi R^2$ is the space area around the user, assuming a communication range R. The area in which the number of nodes does *not* change is the intersection of the two circles in Figure 7.6, which can be computed as

$$S = 2R^2 \cos^{-1}\left(\frac{d}{2R}\right) - \frac{1}{2} d \sqrt{4R^2 - d^2}$$

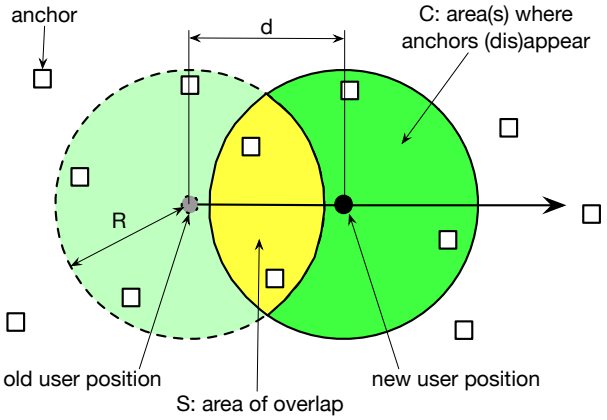


Figure 7.6: Model for neighbor churn.

Velocity (m/s)	c (#nbr changed)	$1/c$ (seconds between changes)
0.05	0.01	90
1.5	0.33	3.0
5	1.11	0.9
10	2.22	0.45
15	3.33	0.30

Table 7.1: Sample values for the rate of neighbor change, to inform the periodicity of neighbor discovery.

where d is the distance traveled by the user in 1 s, as in the figure.

Therefore, the two areas in which nodes change are each $C = U - S$, and therefore there are $c = mC$ nodes appearing (and c disappearing) or, dually, there is a neighbor change every $\frac{1}{c}$ seconds.

To put these numbers in context, with $R = 100$ m (typical for UWB outdoors), $l = 30$ m, and a velocity between 0.05 m/s and 15 m/s, we obtain the values in Table 7.1. 1.5 m/s, 5 m/s are representative of human movement for walking and running, respectively. The highest value is similar to the maximum speed (54 km/h) of large vehicles used in agriculture, like tractors. The lowest value is included to represent the moving speed of a rover like Perseverance [144]. In the planetary exploration case, neighbor discovery can be performed on a time scale of minutes, which translates to almost no power consumption while anchors are isolated.

7.4 Implementation

The following is a detailed explanation of some prominent aspects of our prototype, that are however not part of the core concepts that constitute Sonar. This includes the implementation of active and passive anchor lists in the Sonar header, neighbor table management, and alternative ways to assign slotframes to users. We also provide insights on synchronization operations to motivate our choice for RX guard

times, which affect energy consumption. SONAR was built atop Contiki [72] for the DecaWave EVB1000 platform [83] that hosts the DW1000 radio [40].

Active/known anchor bitmaps. When users broadcast the schedule they need to include a list with the active anchors for ranging within the slotframe and another list with the known anchors. To reduce packet length and transmission time, we have implemented these lists as bitmaps based on the node ID of each anchor. Hence, in a 32-anchor deployment, only 4 bytes are needed per list. Using bitmaps limits flexibility in anchor selection; in our prototype, anchors are used for ranging once per slotframe, and the order depends on the position of the bit associated to the anchor, i.e., the anchor associated to the n-th set bit in the active bitmap will be used in slot n. Still, considering that short IEEE 802.15.4 addresses are 2 bytes, using bit arrays can reduce significantly the packet length. In deployments with hundreds of anchors, or when the application requires full control of ranging requests, transmitting a list with short addresses or node IDs may be preferred.

Slotframe duration and anchor ranging rate. Due to bitmap-based schedule, an anchor will be used for ranging up to once per slotframe. Therefore, the slotframe duration T_{sf} also defines the maximum ranging rate of the anchor. For example, with $T_{sf} = 50$ ms, as in our evaluation, the maximum ranging rate is 20 Hz if the anchor is always active. This limit is not to be confused with the ranging and localization rate offered to the user. In our evaluation §7.6, we use $T_{sf} = 50$ ms, and 7 slots per slotframe dedicated to ranging, giving a maximum ranging rate of 140 Hz.

Strategies for slotframe division. There are two main options to assign slotframe ownership to users. In the presented prototype, we opt for fixed assignment based on the total number of users in the system. Another possibility is to assign slotframes based on the number of users in the same area, which would enhance the ranging rate of co-located users. However, this option increases contention during re-synchronization, since the slotframe assignment would change upon the arrival or departure of any user.

Neighbor table. All nodes, users and anchors, host a neighbor table. Users store information for both discovered users and anchors, while anchors store only the information related to users. Upon receiving a packet from a given node, its corresponding neighbor entry is created or refreshed. Each entry in the table includes the last-heard timestamp for neighbor expiration. Periodically, each neighbor in the table is checked to see if its entry has expired and is therefore no longer valid, in which case it is removed. Expiration time should be carefully chosen to avoid the early removal of a valid neighbor. On the other hand, anchor entries in the user table must be kept fresh for the user to only range with those that are within range, while out-of-range anchors need to be removed. Similarly, anchors need to remove far users to return to isolated mode and save energy. Expiration time is application-dependent and is related to the chosen anchor selection policy. In our prototype, we set the equivalent of $3 \cdot T_{ND}$ for both user and anchor entries, to give nodes multiple chances to be heard before removal.

Low-power synchronization. If anchors could keep the UWB radio always on, no other mechanism except the one presented in §7.3.2 would be needed for synchro-

nization. Unfortunately, precise synchronization is at odds with the need for SONAR to be as energy efficient as possible. To avoid the constant consumption of the DW1000 in idle mode, which can be as high as 18 mA, we make the radio enter deep sleep mode ($<$ 100 nA) at the end of the slot. This means that we cannot exploit the 125 MHz digital PLL clock of the DW1000 (8 ns precision) to maintain the time reference. To enforce the time-slotted operation, we resort to the 32 kHz MCU clock present in the EVB1000 which provides a much lower precision, around 30 µs. Upon synchronization, after receiving a packet, we acquire the current PLL and MCU time to convert the RX timestamp to MCU time. The MCU clock can then be used to schedule the beginning and the end of each slot, firing an interrupt at a given future timestamp.

RX guard and duration. Because of the precision of the MCU clock, we make anchors waiting for `RNG-INIT` start listening $\sim$32 µs in advance. The DW1000 is then configured to search the radio medium for a preamble in the following 64 µs (preamble hunting phase). Due to the guard, this ensures at least 32 µs are allocated to detect the preamble. If a preamble is detected in this very short period, the receiver remains active until the SFD sequence is found, after at most 129 µs. If no preamble is detected, or if no SFD is found, the radio returns to deep sleep. Note that the guard time applies only to the reception of `RNG-INIT`. When the anchor is waiting for `ND-RESP` after transmitting `ND-INIT`, the DW1000 has not yet entered deep sleep and can exploit precise PLL time. Therefore no guard time is needed and preamble hunting is performed for 32 µs. These values for RX duration play an important role in the estimation of the energy consumption, in the next section.

7.5 Energy Consumption

SONAR is designed to reduce the energy consumption of anchors, while interacting with users (active and passive anchors) and especially when isolated. In this section, we break down the energy costs in these various modes of operation to provide an estimate of the lifetime that anchors can achieve. Since SONAR follows a simple slotframe structure, it is easy to compute the expected consumption by combining the cost of neighbor discovery, schedule reception and ranging.

Discharge for neighbor discovery. The lost charge for one (successful) neighbor discovery slot is obtained as:

$$Q_{ND} = Q_{ND}^{TX} + Q_{ND}^{idle} + Q_{ND}^{listen} + Q_{ND}^{RX} \tag{7.1}$$

where the lost charge for each contribution is obtained by multiplying its duration by the associated current draw found in the DW1000 datasheet [70]. Energy expenditure is then obtained by multiplying the lost charge by the supply voltage of the radio, 3.3 V. Table 7.2 shows the duration of packets used in our evaluation and the average current draw per packet. Q_{ND}^{TX} represents the cost for the transmission of the `ND-INIT`, including the writing of the frame into the radio buffer. Similarly, Q_{ND}^{RX} combines the cost of reception of the reply and the read operation from the radio buffer. Q_{ND}^{listen} is the discharge for preamble hunting, when the anchor is trying to detect an ongoing

Table 7.2: Current draw and duration for each frame type. Write/read refer to the required radio operations in addition to TX/RX. Listen refers to the preamble hunting phase, and is only relevant for anchors, that receive `ND-RESP` and `RNG-INIT`. Idle duration is the portion of the response delay (before `ND-RESP` is sent, or waiting to transmit `RNG-RESP`) in which the anchor is not listening or receiving.

	`ND-INIT`	`ND-RESP`	`RNG-INIT`	`RNG-RESP`
Frame duration (µs)	196.86	190.71	199.94	205.06
TX average current (mA)	82.99	83.58	82.71	82.26
RX average current (mA)	131.80	131.68	131.86	131.96
Write duration (µs)	88	82	93	96
Read duration (µs)	75	69	81	84
Write current (mA)	15			
Read current (mA)	12			
Listen/miss duration (µs)	1 / 32		32 / 64	
Listen/miss current (mA)	118			
Idle duration (µs)	661			512
Idle current (mA)	18			
Wake up duration (µs)	5507			
Wake up current (mA)	3.01			
Deep sleep current	< 100 nA			

transmission; its value is close to zero, since we configure the radio to start listening exactly when `ND-RESP` is transmitted (§7.4). Q_{ND}^{idle} is the discharge when the radio is in idle mode, and depends on the user delay between the reception of `ND-INIT` and the transmission of `ND-RESP`. In our prototype, this delay is set to 0.66 ms, which is sufficient to read the content of `ND-INIT`, update the neighbor table, synchronize, and prepare the response.

The calculation above gives the lost charge when a user replies with the `ND-RESP` packet. To estimate the consumption of isolated anchors (no user in the vicinity) we compute the cost of an *unsuccessful* discovery attempt as:

$$\hat{Q}_{ND} = Q_{ND}^{TX} + Q_{ND}^{idle} + \hat{Q}_{ND}^{listen} \tag{7.2}$$

where $\hat{Q}_{ND}^{listen}$ is a listening time that replaces $Q_{ND}^{listen} + Q_{ND}^{RX}$, due to the fixed duration of preamble hunting (32 µs) when reception does not occur.

Discharge for schedule reception. Schedule reception is the only additional operation for passive anchors, which are not used for ranging. The schedule is effectively transmitted as part of the first ranging exchange of the slotframe. Therefore, the discharge is the same as when receiving `RNG-INIT`, and we reuse the same values here and in upcoming calculation for ranging. The lost charge to receive the schedule in one slotframe is:

$$Q_{SR} = Q_{RNG}^{listen} + Q_{RNG}^{RX} \tag{7.3}$$

where Q_{RNG}^{listen} is the consumption during the 32 µs guard time.

Discharge for ranging. Finally, we estimate the cost of a ranging exchange, which is added to the previous costs for active anchors:

$$Q_{RNG} = Q_{SR} + Q_{RNG}^{idle} + Q_{RNG}^{TX} \tag{7.4}$$

Q_{RNG}^{idle} is the energy cost associated to the idle time before the `RNG` response, set to 0.51 ms.

Combining anchor phases. We can combine the energy costs above to obtain the overall energy expenditure of an anchor. We assume that *(i)* an active anchor is such for all nearby users, in all slotframes, *(ii)* in each slotframe the anchor is used for ranging only once (as in our implementation), and *(iii)* it is never the first anchor in the schedule. If the anchor is not always active, energy consumption decreases, approaching the estimate for the passive case. Similarly, consumption is reduced if the anchor is the first in the schedule and can immediately complete ranging in the first slot. While pessimistic, these assumptions simplify the overall estimation: energy consumption does not depend on the number of users since the anchor is ranging once in all slotframes in any case.

We need to define two parameters: neighbor discovery interval T_{ND}, which depends on the wanted responsiveness of neighbor discovery, and slotframe duration T_{sf}, which directly governs the ranging rate of the anchor under our assumptions. We also note that most of neighbor discovery attempt from the anchor will receive no response once all users have been discovered. Therefore, $\hat{Q}_{ND}$ is considered instead of Q_{ND}, and the one-time additional cost for successful discovery is considered to be negligible. Finally, wake-up cost cannot be neglected. We add the cost to wake up the radio from deep sleep, Q_{wu}, at the beginning of the slot to all the previous components.

The resulting consumption over a duration D for each state can be computed as follows:

$$Q_{isolated} = \frac{D}{T_{ND}}(\hat{Q}_{ND} + Q_{wu}) \tag{7.5}$$

$$Q_{passive} = \frac{D}{T_{ND}}(\hat{Q}_{ND} + Q_{wu}) + \frac{D}{T_{sf}}(Q_{SR} + Q_{wu}) \tag{7.6}$$

$$\begin{aligned} Q_{active} = {} & \frac{D}{T_{ND}}(\hat{Q}_{ND} + Q_{wu}) \\ & + \frac{D}{T_{sf}}(Q_{SR} + Q_{RNG} + 2 \cdot Q_{wu}) \end{aligned} \tag{7.7}$$

Anchor lifetime estimation. We quantify the expected lifetime of a Sonar anchor based on the discharge formulas for each phase. The energy cost for individual operations has been computed, resulting in 0.165 mJ for $\hat{Q}_{ND}$, 0.157 mJ for Q_{SR}, and 0.248 mJ for Q_{RNG}. We report in Table 7.3 the average energy cost per slotframe,

Table 7.3: Energy consumption for an anchor in isolated, passive or active mode. Estimated duration of a 10 Ah, 3.7 V battery in each mode and for mixed anchor activity patterns.

Slotframe duration T_{sf} (s)	0.05			0.5		
Neighbor discovery interval T_{ND} (s)	0.1	0.3	1	0.1	0.3	1
Energy consumption						
Isolated cost per slotframe (mJ)	0.084	0.030	0.010	0.845	0.296	0.104
Passive cost per slotframe (mJ)	0.242	0.187	0.168	1.002	0.453	0.261
Active cost per slotframe (mJ)	0.490	0.435	0.416	1.251	0.702	0.509
Lifetime						
100% isolated battery duration	2.4y	6.9y	19.7y	2.4y	6.9y	19.7y
100% passive battery duration	308d	1.1y	1.2y	2.0y	4.5y	7.8y
100% active battery duration	152d	171d	179d	1.6y	2.9y	4.0y
5% passive 5% active	1.8y	3.5y	5.3y	2.3y	6.3y	15.5y
20% passive 20% active	1.0y	1.4y	1.7y	2.1y	5.0y	9.4y
50% passive 50% active	204d	240d	255d	1.8y	3.5y	5.3y

computed from equations (7.5) to (7.7), and the expected lifetime when using a large battery providing 10.4 Ah at 3.7 V. We do not account for the loss of battery capacity due to aging or environmental factors, such as temperature. In our analysis, we also consider the constant current consumption of the board that would host the DW1000. However, the EVB1000 platform used in our setup is an evaluation board whose consumption is not representative of the achievable performance. We use a 13μA current consumption and a 93% power efficiency, the same considered by the manufacturer in a similar evaluation [145]. We considered two values for T_{sf} and three for T_{ND}, plus several activity patterns, from isolated to always active anchors. The choice of T_{sf} depends on the desired ranging rate. T_{ND}, instead, controls the responsiveness of neighbor discovery (§7.3.4). With $T_{sf} = 50$ ms, $T_{ND} = 0.3$ s and a low-activity pattern that sees the anchor passive for 5% of the time and active for an additional 5% of the time, the battery is expected to last more than 3 years and a half. This corresponds to an average ranging rate of 1 Hz. Note that, while active, the anchor performs ranging at 20 Hz, which is sufficient to support high-rate localization for the vehicles in our target scenarios. Even in a high-activity scenario, where the anchor is passive half of the time and active the other half, with an average ranging rate of 10 Hz, the battery is expected to last 240 days. For shorter `ND` intervals and longer slotframes, the anchors would last many years. For example, with $T_{ND} =$ 1 s and $T_{sf} = 500$ ms (in low-activity scenario, i.e., average ranging rate 0.1 Hz), the battery would last almost 16 years. Finally, we look back at our motivating scenario, planetary exploration. For low-activity applications with slow users (i.e., T_{ND} in the order of minutes, few ranging exchanges per minute), the average consumption becomes almost negligible. Therefore, given the small impact of Sonar on the battery, the UWB infrastructure could easily support additional services, like communication between the rover and the lander.

7.6 Evaluation

We evaluate SONAR in an UWB testbed at our premises, assessing the capability of the system to quickly synchronize and dynamically define the ranging schedule. We first show an execution example, with a person carrying an UWB tag representing the user. Then, we observe that ranging outliers may appear for neighboring nodes that have not discovered each other. We analyze the relationship between communication range and interference range to prevent them. We then investigate the robustness of neighbor discovery and slotframe alignment in benchmark experiments, with a selection of testbed nodes acting as anchors or users. Testbed nodes configured as users stop and resume operations in controlled time intervals, simulating arrival and departure. In multi-user benchmarks, we show that user synchronization is achieved quickly even in complex scenarios and that the ranging success rate in SONAR is comparable to the success rate of the baseline ranging application. These experiments explore the most adverse conditions that can affect SONAR, including the *simultaneous* arrival of up to five users in the same area.

7.6.1 Discovery and Ranging in Single-user and Multi-user Mode

We configure SONAR as follows. The RF settings for the DW1000 radio are 6.8 Mbps data rate, 64 MHz PRF, 128 symbols preamble length, channel 2. We set $S = 10$ slots per slotframe, leaving always the last 3 for neighbor discovery, and $T_{slot} = 5$ ms, which gives $T_{sf} = 50$ ms and a maximum ranging rate of 140 Hz.

A person carries an UWB tag across a testbed, the same infrastructure used in Chapter 4. This mobile user discovers various anchors and performs ranging, as clearly shown by the V-shape of the estimated distance created by the user approaching the anchor and then leaving (Figure 7.7). The hand-held tag is one of two users in the experiments. The other user is one of the infrastructure nodes. We use only two users to improve visualization and to clearly distinguish the zones where the users are interacting with different anchors, in single-user mode, from those where the two users meet, switching to multi-user mode. The user can select up to 7 nearby anchors for ranging in the same slotframe, but each slotframe in multi-user mode is entirely assigned to a specific user. The transition between modes is more evident in Figure 7.8, a zoom-in of the previous one, where the gaps in ranging on the right are due to slotframe division.

For the remainder of the evaluation, we keep the same SONAR configuration but use channel 5. We note that channel 5 has a significantly lower communication range w.r.t. channels with lower central frequency, also available with DW1000. However, short communication range is beneficial for evaluation, allowing us to explore different topologies even in indoor settings.

7.6.2 Interference vs. Neighbor Discovery Range

In experiments with mobile users, similar to the previous one, we observed the appearance of outliers when users were departing. Because SONAR users and anchors are synchronized, the problem of CIR interference might emerge if users that were

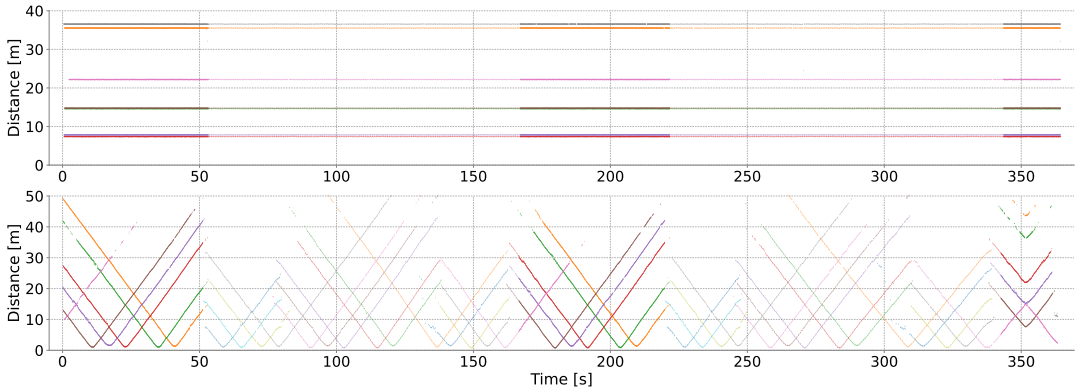


Figure 7.7: A stationary user (above) and a mobile user (below) range with the neighboring anchors. When the mobile user discovers the same anchors used by the stationary one, they switch to multi-user mode.

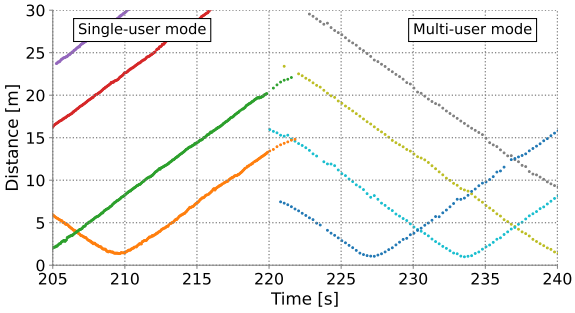


Figure 7.8: Zoom-in of the mobile user, switching from single-user (ranging in all slotframes) to multi-user mode (ranging in a subset of the slotframes).

previously co-located (and time-aligned) return to single-user mode, even if they do not interact with the same anchors anymore. Indeed, interference range is larger than communication range, which means that nodes can interfere with each other may not be able to discover each other. When this happens, nodes in interference range transmit in the same slots, causing their signals to overlap. While overlapping UWB transmissions rarely cause collisions at the receiver, the energy in the interfering preamble is still accumulated in the estimation of the channel impulse response (CIR). This can cause an extraneous peak to appear in the CIR, compromising the first path detection algorithm, and finally resulting in large ranging errors (see also §2.4.4).

Several solutions can be considered. If users are always expected to be co-located, or increasing the ranging rate is not necessary, they could simply stay in multi-user mode and never switch back to single-user. It is also possible to assign specific complex channels (§2.4.2, §2.4.3) to each user, reducing interference. Finally, the system could perform `ND` with greater transmission power (and therefore longer communication range) w.r.t. `RNG`. If the communication range of `ND` messages from anchors extends beyond `RNG` interference range, then no user is affected by overlapping trans-

missions from distant anchors. Instead, any interfering anchor would be quickly discovered, which in turn would trigger multi-user mode again.

But what is the TX gain required by ND messages to achieve a communication range beyond the CIR-level interference range of RNG messages? We experimentally obtain an approximation of such gain, replicating the interference scenario with testbed nodes. The ranging initiator synchronizes two other nodes. The recipient closer to the initiator is the ranging responder, while the other is the interference source. The responder is the farthest testbed node within RNG communication range w.r.t. the initiator. We observe a 13% success rate for ranging. For the interference source, we pick the farthest testbed node that negatively affects ranging results, as clearly noticeable from the standard deviation of measurements ($>$ 1 m). We perform 8000 ranging rounds and find that, with the responder at $\sim$29 m from the initiator, an interference source at $\sim$57 m or closer can negatively affect ranging. In Sonar, the interference source could be an anchor. Therefore, its ND packets should use a sufficient TX gain to reach the initiator of this experiment (i.e., a user). We found that a TX gain of 7 dB gives a reliable link towards the initiator (with 81% PDR), and we use this value in our evaluation.

7.6.3 Analysis in Multi-user Mode

Whenever a new user engages with the surrounding anchors it just discovered, other users in the vicinity may suffer a temporary disruption due to re-synchronization. Therefore, it is critical to evaluate the performance of Sonar during these transitions.

The example presented in §7.6.1 is not suited to extract quantitative results. To this end, we need a more controlled setup, with predetermined neighbor sets and known distances between nodes. We employ fixed nodes of the testbed to represent both users and anchors.

We test the protocol by running many experiments in which several users appear almost simultaneously, contending for channel access in the attempt to range with the same group of anchors. The random arrival of users is "staged" in software. The experiment is divided into 200 rounds. At the beginning of a round, anchors and users apply a random delay between 0 and the duration of a slotframe (T_{sf}) to enforce a complete lack of synchronization. As users re-join the system in each round, they have to organize the schedule quickly to prevent collisions and spurious ranging results due to unwanted overlapping signals. Between rounds, users stop interacting with the surrounding nodes, making all of them "forget" their neighbors, thus artificially emulating users that leave the area to return later.

Our experiments produce the following indicators: 1. *synchronization time*, i.e., the delay between the appearance of a new user and its synchronization to the user with lowest ID, 2. *ranging success rate*, and 3. *distance estimation error*. For ranging metrics, i.e., success rate and error, we compare against an application performing single-sided two-way ranging. This baseline is acquired by scheduling each ranging exchange individually so that no other node is interfering.

Intuitively, synchronization time is affected by the ranging rate, as users can overhear

the ranging exchanges of other nodes. But it is especially related to the neighbor discovery interval T_{ND}, even more so when users are not directly in range with each other. We identify three relevant scenarios:

1. CLIQUE. The set of users forms a clique; as soon as a user begins ranging, all users can discover it by directly overhearing its ranging poll (`RNG-INIT`) or its neighbor discovery response (`ND-RESP`), in addition to the packets from anchors.

2. CONNECTED. The set of users is connected, but a given user can overhear only some of the others; while discovery through ranging exchanges is possible, the propagation of synchronization is not instantaneous. It potentially takes multiple handshakes spreading over multiple hops.

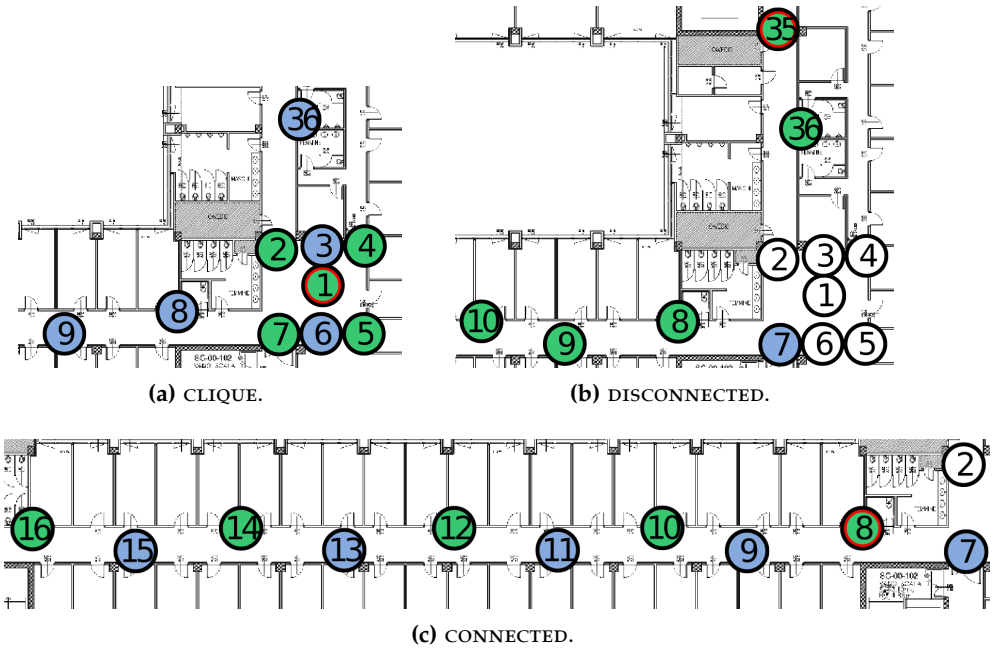
3. DISCONNECTED. The set of users is not connected, but at least one anchor exists that is shared among the separated groups of users; synchronization can only happen by means of neighbor discovery beacons (`ND-INIT`) or ranging responses (`RNG-RESP`) from the shared anchor.

By carefully selecting different nodes of the testbed, we can replicate the three scenarios above, and show that synchronization is achieved quickly regardless of the type of connectivity between users.

Node setup. Figure 7.8 shows which nodes act as users and anchors in each scenario. For CLIQUE, all users are in a hall with no obstacles between them. Node 1 is the reference, i.e., the user with lowest ID that others must synchronize to. In CONNECTED, the limited transmission range creates a multi-hop synchronization structure. The `RNG` messages of node 8, the reference user with ID 1 in this scenario, can reach 10, but not 12, 14 and 16. The DISCONNECTED scenario shows two separated groups of users, and only one node (7) that acts as an anchor. Node 35 is the reference, in communication range with user 36. However, there is no link between them and the second group of users, making anchor 7 the only source of synchronization for nodes 8, 9, 10. Nodes 10 and 35 also have weak `RNG` links to the anchor.

Synchronization time. We show CDFs for the synchronization time, i.e., the delay before users align their slotframes to the reference. First, we experiment with two users in range with each other and a single anchor. In this case, as expected, synchronization time is uniformly distributed between $\sim$0 ms and T_{ND}. Users do not transmit any message until they discover at least one anchor, and the only way for anchors to advertise their presence in "isolated" mode is through neighbor discovery beacons. Once they do, the reference user discovers the first anchor and begins ranging. Its ranging polls are received by the other user(s), quickly reaching synchronization. Beyond this very basic case, we compute the synchronization time for different setups (CLIQUE, CONNECTED, DISCONNECTED) involving more anchors and users, with varying T_{ND} (1 s, 0.3 s, 0.1 s).

The synchronization time across all users in CLIQUE and CONNECTED is shown in (Figure 7.9a) and (Figure 7.9b), respectively. In both cases, the CDF grows until the delay that is slightly larger than the neighbor discovery rate. In the case of CLIQUE, this can be ascribed to occasional packet losses. Instead, CONNECTED shows longer



(a) CLIQUE. **(b)** DISCONNECTED.

(c) CONNECTED.

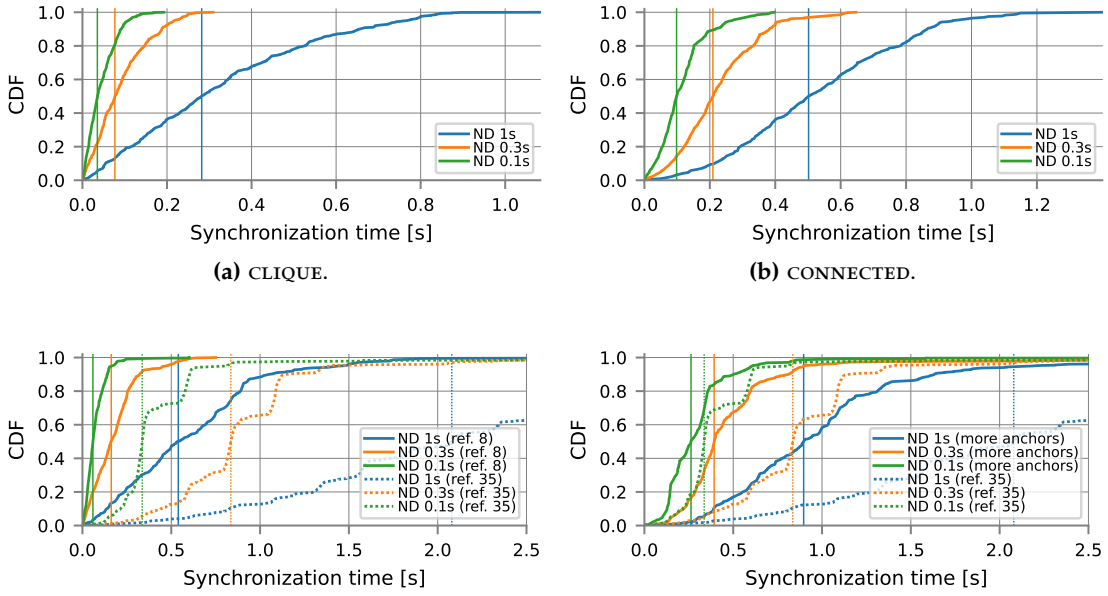
Figure 7.8: Users (green) and anchors (blue) in the different connectivity scenarios. The reference user is highlighted in red.

tails of the distribution, due to the multi-hop topology which can occasionally delay the propagation of the reference when user 8 is not the first discovered.

Next we analyze the results for the DISCONNECTED scenario, the most challenging of the three. Since the only way to propagate the synchronization information is through a single anchor, the time can increase for users that are not neighbors of the reference. If the reference is far from the anchor, its `ND-RESP` will often be superseded by one of the other concurrent replies whose signal is stronger due to proximity. This is the case when selecting node 35 as the reference user (Figure 7.9c). Since multiple `ND` rounds are often required, synchronization is delayed up to 4.85 s with $T_{ND} = 1$ s. However, the figure also shows that if the reference is 8, the user closest to the anchor, synchronization is significantly faster.

It is worth noting that the described worst-case scenario (DISCONNECTED with distant reference) is unlikely to appear in real deployments. The increased synchronization time depends on multiple factors affecting the system simultaneously: 1. multiple users entering anchor range at the same time, 2. different sets of users that cannot communicate directly, 3. a single anchor available for the communication between the sets of users, and finally 4. the reference user being farther from the anchor w.r.t. the others. Even under these extremely unfavorable conditions, SONAR reliably achieves synchronization at the cost of a small increase in latency. If a long `ND` interval is required for energy efficiency, but multiple users are expected to appear simultaneously (as in a swarm), SONAR can be configured to transmit another `ND-INIT` following a successful discovery to speed up the process at a small energy cost.

Ranging success rate and error. After assessing synchronization delay, we evaluate



(a) CLIQUE.

(b) CONNECTED.

(c) DISCONNECTED (comparing ref. 8 and 35).

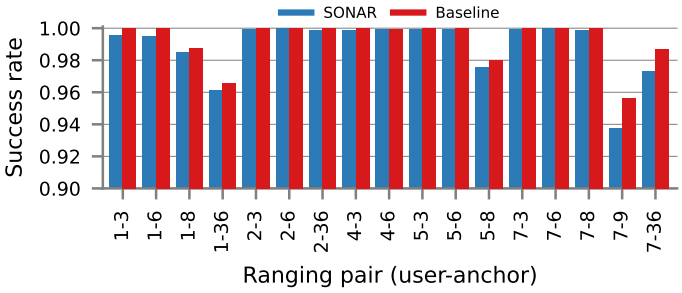
(d) DISCONNECTED (single anchor vs. multiple anchors between users).

Figure 7.9: Synchronization time in seconds for all users across different scenarios and varying ND interval. Vertical lines represent the median values. Figure 7.9d shows the improvements when multiple anchors are available to discover the reference and propagate synchronization.

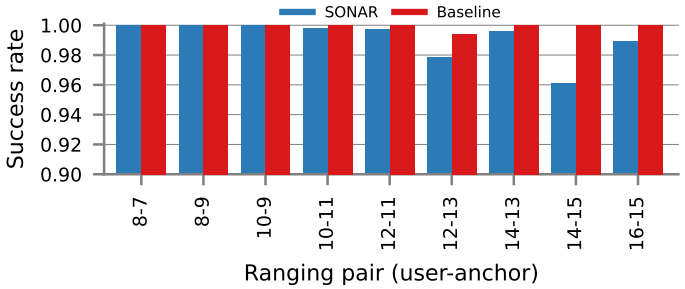
the reliability of ranging and the error in distance estimation. Because multiple users attempt to range with the same anchors, their packets could collide, or create interference in the CIR estimation and therefore increase the ranging error. Frequent ranging failures or a difference error distribution would then indicate that user transmissions often collide and/or interfere with each other. We compare against a baseline which schedules ranging exchanges in a round-robin fashion among nodes, with no chance of interference. For the baseline, the radio of the nodes is always on, but we do not observe significant difference in performance when following the same deep sleep schedule as SONAR.

For each SONAR user, the ranging success rate is computed after two conditions are met: 1. the user has latched onto the anchor, i.e., after bidirectional discovery has been confirmed through ND-FINAL, and 2. the user has synchronized to the reference. Indeed, the notion of failed ranging is fuzzy before synchronization, when the shared schedule is not yet defined. Results for $T_{ND} = 1$ s are reported in Figure 7.10. Similar success rates are obtained for the other ND intervals. Overall, SONAR achieves a ranging success rate comparable to that of the baseline application.

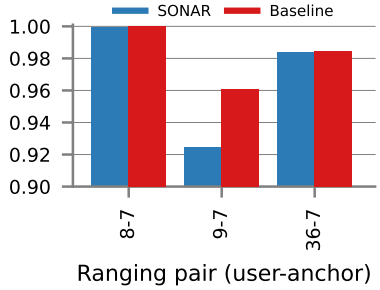
Figure 7.11 shows the aggregated distance estimates for SONAR and the baseline in each connectivity scenario. SONAR estimates are similar to those of the baseline for all ranging pairs, with the mean distance within 2 cm from the baseline result. This confirms that the established ranging schedule is free from interference.



(a) CLIQUE.

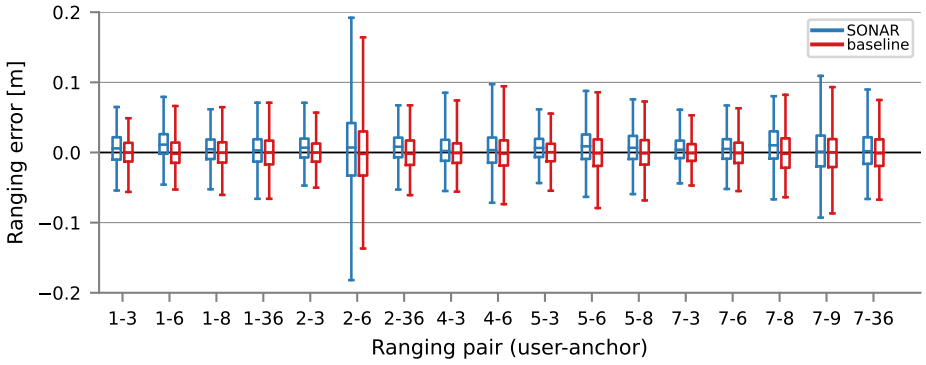


(b) CONNECTED.

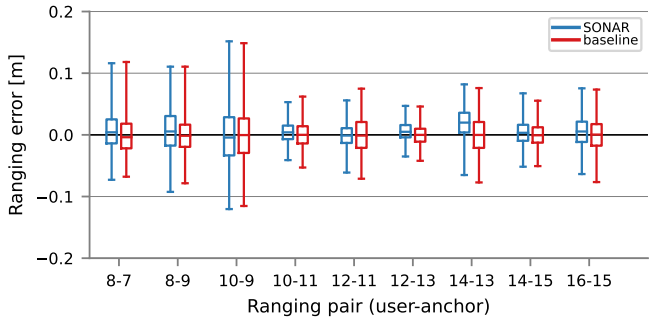


(c) DISCONNECTED.

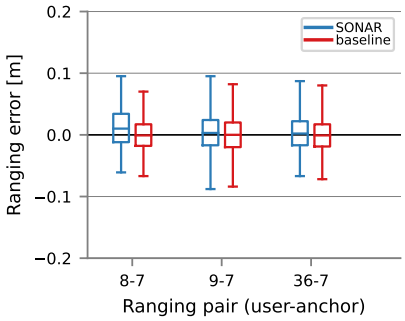
Figure 7.10: Comparison of the ranging success rate of SONAR and the baseline ($T_{ND} = 1$ s).



(a) CLIQUE.



(b) CONNECTED.



(c) DISCONNECTED.

Figure 7.11: Comparison of the distance estimation of SONAR and the baseline ($T_{ND} = 1$ s). Boxes represent the 25-75% percentile. Bars represent 0.1-99.9% of the data.

7.7 Conclusions

Since UWB radios are generally considered too energy-hungry, anchor infrastructures for ranging and/or localization are typically mains-powered. This allows anchors to keep their radio always on, and removes any concern in terms of energy consumption. On the other hand, this approach increases installation costs, limits deployment flexibility, and even precludes UWB infrastructures entirely from environments with no access to the power grid.

In this chapter we introduced Sonar, a novel protocol to support multiple users ranging with battery-powered anchors. Our design jointly provides efficient neighbor discovery, quick synchronization, and dynamic arrangement for the ranging schedule. By offloading radio operations to users, all these features come with minimal energy cost on the anchor side. Sonar is easily configurable in terms of discovery responsiveness and ranging rate, to adapt the protocol to different application needs. Indeed, we show that anchor lifetime can stretch to decades, making Sonar a practical solution for UWB ranging and localization infrastructures in any area.

8

Conclusions

In this thesis, we exploit UWB for both communication and localization, tackling the open issues that come with energy constraints and large-scale deployments. Most of UWB research revolves around the achievable ranging and localization accuracy, rather than the network coordination aspects required for practical deployments. Because of this, the technology has often been used in small setups, that are inapplicable in large areas. Moreover, because UWB radios have relatively high energy consumption compared to narrowband ones, their communication features have been barely explored by researchers, let alone applied in large multi-hop networks.

Instead, we demonstrate that UWB can be applied in large areas, with solutions for *i)* reliable and energy-efficient communication, through data dissemination and data collection protocols, and *ii)* two flavors of localization, TDoA for battery-powered tags and ranging-based for battery-powered anchors.

We accomplish this through the following main contributions:

1. **Characterization of UWB concurrent transmissions.** The proposed communication protocols are based on UWB concurrent transmissions (i.e., transmissions in the same frequency channel that overlap in time at receivers), whose applicability we analyzed in our first contribution. We distilled the guidelines to use concurrent transmissions effectively, paving the way not only for UWB protocols akin to those seen in narrowband, but also providing researchers with key insights for the design of entirely novel UWB systems.

2. **Multi-hop dissemination with Glossy and convergecast with Weaver.** We developed and evaluated the UWB version of the popular Glossy protocol. We have experimentally shown that UWB concurrent transmissions achieve very high dissemination reliability and order-of-magnitude improvements in terms of time synchronization error, with energy consumption comparable to narrowband radios. For convergecast, a pivotal traffic pattern in sensor networks, we departed from the common approach that uses Glossy floods as the funda-

mental communication primitive. Instead, Weaver logic combines individual concurrent transmissions to merge multiple data flows together. The approach, that we evaluated in very different network topologies and with mobile devices, has proven to be more energy efficient and more reliable than Crystal, a Glossy-based state-of-the-art protocol, while also unlocking ultra-low latency.

The two traffic patterns we targeted in this thesis provide the substrate for wireless control systems based on UWB, hitherto uncharted territory in the low-power networking community. To encourage further development, we have publicly released the full stack of Glossy and Weaver implementations.

3. **Large-scale localization and analytics with Talla.** We introduced a TDoA system based on wireless synchronization that can operate in multi-hop, without compromising accuracy when tags move across different groups of anchors. As part of the contribution, we analyze the clock behavior of the radios, generate faithful artificial clock traces, bring out the timing-induced localization error, and use both simulations and real-world experiments to provide the guiding principles when choosing the synchronization frequency and the number of anchors. In addition, we give technical recommendations for the definition of an efficient TDMA schedule, and for the process of TDoA synchronization in the case of non-line-of-sight (NLOS) links between anchors.

   Talla greatly simplifies continuous tracking since all anchors operate in the same time domain and any sub-group of anchors can be chosen in the localization process, which is crucial to avoid gaps in tracking. The coverage offered by Talla makes it an ideal candidate in industrial use cases, to find assets scattered in large warehouses, to support autonomous navigation in real time, and to extract movement traces for analytics.

   Talla applicability is not limited to industrial scenarios, as its localization accuracy makes it also very appealing for the analysis of human movement. With a Talla infrastructure in the MUSE science museum (Trento, Italy), we captured the visit of several persons, carrying a small tag on the chest with a necklace, as they moved between the many exhibits. After defining a novel metric to identify the best stop detection technique, we found that the estimated stop position could be reliably matched to the right exhibit, with small error in terms of estimated stop duration.

4. **Battery-powered localization infrastructures with Sonar.** In our last contribution we focus on a system for battery-powered infrastructures. We design Sonar for ranging-based localization, to avoid the infrastructure complexity that comes with TDoA operations, and consume less energy. To be as energy-efficient as possible, Sonar anchors remain in deep sleep mode most of the time, activating only if a tag is in range and only in the few slots necessary for neighbor discovery and ranging.

   Due to the low energy cost for anchors, Sonar can bring UWB in areas with no access to the power grid. The quintessential example is the scenario of planetary exploration, the purpose that first motivated this research in collaboration

with Thales Alenia Space. As in the best tradition of space research, its results also find application on Earth. For instance, Sonar can be used to enable accurate localization for smart agriculture. But it also greatly facilitates anchor deployment in any environment, including indoors. Since anchors require no power connection and no communication backbone for their operations, Sonar avoids the installation costs that come with cabling, especially relevant in large areas.

Outlook and future work. In this thesis we met our initial objectives towards large-scale UWB solutions, yet the many applications for this technology and the continuous evolution of radios and standards open up new scenarios in both the communication and the localization domains.

First, the achieved results could be exploited together, prompting us to make a case for UWB control in industrial settings, with localization for tracking and navigation, and communication for monitoring and actuation. The two services can be combined in a single, easy to deploy system, with no need for a wired connectivity backbone. Nevertheless, meeting the stringent guarantees required by industry is still a difficult task. While protocols like Weaver can achieve ultra-fast data collection, how to provide bounds on latency to consistently respect hard deadlines on the radio medium remains an open question.

Another major issue associated with UWB systems—and tackled in this thesis—is the lack of support for battery-powered nodes, justified by the high current drain from UWB chips. However, even the bad reputation of UWB in terms of energy consumption is being challenged, on two fronts. Newly released chips significantly reduce the consumption of the radio in reception mode, the culprit of the issue. On the other hand, our protocols show that careful scheduling and proper use of concurrent transmissions drastically abate radio uptime, making UWB battery-powered nodes feasible—already today.

From the point of view of energy consumption, our communication protocols do not make distinctions between nodes in the system, as they all run the same firmware and enter deep sleep under the same conditions. Instead, our localization approaches are asymmetric, with tags and anchors playing very different roles. Talla ensures low consumption for the tag, while Sonar is meant to support battery-powered anchors. As part of our future work we plan to design a protocol for neighbor discovery and ranging, in the same vein of Sonar, where all nodes operate under duty cycle to save energy. The symmetry of node roles would make this approach particularly flexible. It could be used to perform ranging between devices with no infrastructure, supporting proximity detection, as in social distancing applications, geo-fencing, or location-based services. By simply assigning known coordinates to some of the nodes (e.g., via GPS), it could also provide a full, battery-powered localization infrastructure.

Some technical challenges remain to be solved in the localization domain. As highlighted in our work on stop-move detection, high localization error is often due to NLOS conditions attenuating the signal. In our case, the main factor was the human body. The latter has a particularly harsh impact and requires a dedicated solution,

which is currently one of our main objectives. Nevertheless, NLOS due to other types of obstructions, such as concrete walls or metallic barriers, is also likely to affect localization performance. Therefore, it is essential to identify the presence of an obstruction between tags and anchors that is affecting the estimation of timestamps, to then disregard or correct the measurement. Some algorithms already exist that counteract NLOS, whose robustness in various environments is the subject of active research. A reliable mechanism to reduce NLOS error could allow UWB radios to achieve the theoretical centimeter-level error that, to this day, is only possible in ideal conditions. NLOS identification techniques would also benefit TDoA synchronization in Talla, as they would allow the solver to quickly react to changes in link conditions and exclude badly synchronized anchors.

For both communication and localization, the co-existence with other UWB networks will soon become an issue worth of consideration. The multiplication of IoT and personal devices embedding UWB radios is expected to significantly increase the level of interference and cause a number of inconveniences. For one thing, false preamble detection due to external transmitters can trigger receivers, leading to energy waste. Moreover, wrong receptions have the potential to halt the propagation of floods altogether, harm synchronization in protocols like Talla, or generate extraneous peaks in the estimation of the channel impulse response and therefore affect localization. Based on what we observed in our characterization of concurrent transmissions, the first step to decouple a system from co-located interference sources is to ensure tight synchronization between its transmitters and receivers. Borrowing from the experience accrued by narrowband researchers, we also foresee techniques like channel hopping becoming a necessity in UWB as well, not only for communication but also for localization.

Finally, after noting the difficulties emerging in a world where UWB is increasingly present, we look forward to the possibilities enabled by its widespread adoption. Indeed, UWB is becoming mainstream, soon to be on most smartphones and wearable devices. Researchers and developers are waiting for the release of APIs granting access to those UWB chips, which will undoubtedly mark the next major step towards the popularization of the technology. Scenarios like the museum analytics case presented in this dissertation—and indoor localization at large—will exploit personal devices, making UWB yet another technology to blend in our daily lives and that we take for granted. This thesis sets the stage for the forthcoming proliferation of UWB radios, by contributing novel, efficient communication and localization techniques and evaluating them in realistic conditions, ready for the multitude of applications and challenges ahead.

Bibliography

[1] D. Vecchia, P. Corbalán, T. Istomin, and G. P. Picco. "Playing with Fire: Exploring Concurrent Transmissions in Ultra-wideband Radios". In *Proceedings of the* 18^{th} *IEEE International Conference on Sensing, Communication and Networking (SECON)*. 2019. DOI: 10.1109/SAHCN.2019.8824929.

[2] D. Lobba, M. Trobinger, D. Vecchia, T. Istomin, and G. P. Picco. "Concurrent Transmissions for Multi-hop Communication on Ultra-wideband Radios". In *Proceedings of the* 17^{th} *International Conference on Embedded Wireless Systems and Networks (EWSN)*. 2020.

[3] M. Trobinger, D. Vecchia, D. Lobba, T. Istomin, and G. P. Picco. "One Flood to Route Them All: Ultra-fast Convergecast of Concurrent Flows over UWB". In *Proceedings of the* 18^{th} *ACM Conference on Embedded Networked Sensor Systems (SenSys)*. 2020.

[4] D. Vecchia, P. Corbalán, T. Istomin, and G. P. Picco. "TALLA: Large-scale TDoA Localization with Ultra-wideband Radios". In *Proceedings of the* 10^{th} *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. 2019. DOI: 10.1109/IPIN.2019.8911790.

[5] F. Hachem, D. Vecchia, M. L. Damiani, and G. P. Picco. "Fine-grained Stop-Move Detection in UWB-based Trajectories". In *Proceedings of the* 20^{th} *IEEE International Conference on Pervasive Computing and Communications (PerCom)*. 2022.

[6] A. Nikoukar, S. Raza, A. Poole, M. Güneş, and B. Dezfouli. "Low-Power Wireless for the Internet of Things: Standards and Applications". In *IEEE Access* 6 (2018), pp. 67893–67926.

[7] U. Raza, P. Kulkarni, and M. Sooriyabandara. "Low Power Wide Area Networks: An Overview". In *IEEE Communications Surveys & Tutorials* 19.2 (2017), pp. 855–873.

[8] A. Čolaković and M. Hadžialić. "Internet of Things (Iot): A Review of Enabling Technologies, Challenges, and Open Research Issues". In *Computer Networks* 144 (2018), pp. 17–39.

[9] J. V. Sobral, J. J. Rodrigues, R. A. Rabêlo, J. Al-Muhtadi, and V. Korotaev. "Routing Protocols for Low Power and Lossy Networks in Internet of Things Applications". In *Sensors* 19.9 (2019), p. 2144.

[10] A. Kumar, M. Zhao, K.-J. Wong, Y. L. Guan, and P. H. J. Chong. "A Comprehensive Study of IoT and WSN MAC Protocols: Research Issues, Challenges and Opportunities". In *IEEE Access* 6 (2018), pp. 76228–76262.

[11] R. C. Carrano, D. Passos, L. C. Magalhaes, and C. V. Albuquerque. "Survey and Taxonomy of Duty Cycling Mechanisms in Wireless Sensor Networks". In *IEEE Communications Surveys & Tutorials* 16.1 (2013), pp. 181–194.

[12] R. E. Mohamed, A. I. Saleh, M. Abdelrazzak, and A. S. Samra. "Survey on Wireless Sensor Network Applications and Energy Efficient Routing Protocols". In *Wireless Personal Communications* 101.2 (2018), pp. 1019–1055.

[13] S. A. Kumar and P. Ilango. "The Impact of Wireless Sensor Network in the Field of Precision Agriculture: A Review". In *Wireless Personal Communications* 98.1 (2018), pp. 685–698.

[14] C. Gomez, S. Chessa, A. Fleury, G. Roussos, and D. Preuveneers. "Internet of Things for Enabling Smart Environments: A Technology-centric Perspective". In *Journal of Ambient Intelligence and Smart Environments* 11.1 (2019), pp. 23–43.

[15] M. Alaa, A. A. Zaidan, B. B. Zaidan, M. Talal, and M. L. M. Kiah. "A Review of Smart Home Applications Based on Internet of Things". In *Journal of Network and Computer Applications* 97 (2017), pp. 48–65.

[16] C. Bachmann, M. Ashouei, V. Pop, M. Vidojkovic, H. De Groot, and B. Gyselinckx. "Low-Power Wireless Sensor Nodes for Ubiquitous Long-Term Biomedical Signal Monitoring". In *IEEE Communications Magazine* 50.1 (2012), pp. 20–27.

[17] Y. Mehmood, F. Ahmad, I. Yaqoob, A. Adnane, M. Imran, and S. Guizani. "Internet-Of-Things-Based Smart Cities: Recent Advances and Challenges". In *IEEE Communications Magazine* 55.9 (2017), pp. 16–24.

[18] M. Ceriotti, M. Corrà, L. D'Orazio, R. Doriguzzi, D. Facchin, G. P. Jesi, R. L. Cigno, L. Mottola, A. L. Murphy, M. Pescalli, et al. "Is There Light at the Ends of the Tunnel? Wireless Sensor Networks for Adaptive Lighting in Road Tunnels". In *Proceedings of the* 10^{th} *ACM/IEEE International Conference on Information Processing in Sensor Networks*. IEEE. 2011, pp. 187–198.

[19] P. Baronti, P. Barsocchi, S. Chessa, A. Crivello, M. Girolami, F. Mavilia, and F. Palumbo. "Remote Detection of Social Interactions in Indoor Environments through Bluetooth Low Energy Beacons". In *Journal of Ambient Intelligence and Smart Environments* 12.3 (2020), pp. 203–217.

[20] T. Istomin, E. Leoni, D. Molteni, A. L. Murphy, G. P. Picco, and M. Griva. "Janus: Dual-radio Accurate and Energy-Efficient Proximity Detection". In *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 5.4 (2021), pp. 1–33.

[21] T. Torfs, T. Sterken, S. Brebels, J. Santana, R. van den Hoven, V. Spiering, N. Bertsch, D. Trapani, and D. Zonta. "Low Power Wireless Sensor Network for Building Monitoring". In *IEEE Sensors Journal* 13.3 (2012), pp. 909–915.

[22] M. Ceriotti, L. Mottola, G. P. Picco, A. L. Murphy, S. Guna, M. Corra, M. Pozzi, D. Zonta, and P. Zanon. "Monitoring Heritage Buildings with Wireless Sensor Networks: The Torre Aquila Deployment". In *2009 International Conference on Information Processing in Sensor Networks*. IEEE. 2009, pp. 277–288.

[23] E. D. Ayele, K. Das, N. Meratnia, and P. J. Havinga. "Leveraging BLE and Lora in IoT Network for Wildlife Monitoring System (WMS)". In *2018 IEEE 4th World Forum on Internet of Things (Wf-IoT)*. IEEE. 2018, pp. 342–348.

[24] G. P. Picco, D. Molteni, A. L. Murphy, F. Ossi, F. Cagnacci, M. Corrà, and S. Nicoloso. "Geo-Referenced Proximity Detection of Wildlife with Wildscope: Design and Characterization". In *Proceedings of the* 14th *International Conference on Information Processing in Sensor Networks*. 2015, pp. 238–249.

[25] D. Baumann, F. Mager, U. Wetzker, L. Thiele, M. Zimmerling, and S. Trimpe. "Wireless Control for Smart Manufacturing: Recent Approaches and Open Challenges". In *Proceedings of the IEEE* 109.4 (2020), pp. 441–467.

[26] E. Sisinni, A. Saifullah, S. Han, U. Jennehag, and M. Gidlund. "Industrial Internet of Things: Challenges, Opportunities, and Directions". In *IEEE Transactions on Industrial Informatics* 14.11 (2018), pp. 4724–4734.

[27] S. Vitturi, C. Zunino, and T. Sauter. "Industrial Communication Systems and Their Future Challenges: Next-Generation Ethernet, IIoT, and 5G". In *Proceedings of the IEEE* 107.6 (2019), pp. 944–961.

[28] A. Rácz-Szabó, T. Ruppert, L. Bántay, A. Löcklin, L. Jakab, and J. Abonyi. "Real-Time Locating System in Production Management". In *Sensors* 20.23 (2020), p. 6766.

[29] T. Ruppert and J. Abonyi. "Integration of Real-Time Locating Systems into Digital Twins". In *Journal of Industrial Information Integration* 20 (2020), p. 100174.

[30] N. Soltanmohammadlou, S. Sadeghi, C. K. Hon, and F. Mokhtarpour-Khanghah. "Real-Time Locating Systems and Safety in Construction Sites: A Literature Review". In *Safety Science* 117 (2019), pp. 229–242.

[31] B. Berg, G. Longley, and J. Dunitz. "Improving Clinic Operational Efficiency and Utilization with RTLS". In *Journal of Medical Systems* 43.3 (2019), pp. 1–9.

[32] M. N. Kamel Boulos and G. Berry. "Real-Time Locating Systems (RTLS) in Healthcare: a Condensed Primer". In *International Journal of Health Geographics* 11.1 (2012), pp. 1–8.

[33] A. T. Lo'ai, A. Basalamah, R. Mehmood, and H. Tawalbeh. "Greener and Smarter Phones for Future Cities: Characterizing the Impact of GPS Signal Strength on Power Consumption". In *IEEE Access* 4 (2016), pp. 858–868.

[34] C. Mandrioli, A. Leva, B. Bernhardsson, and M. Maggio. "Modeling of Energy Consumption in GPS Receivers for Power Aware Localization Systems". In *Proceedings of the* 10th *ACM/IEEE International Conference on Cyber-Physical Systems*. 2019, pp. 217–226.

[35] K. Merry and P. Bettinger. "Smartphone GPS Accuracy Study in an Urban Environment". In *Plos One* 14.7 (2019), e0219890.

[36] O. K. Isik, J. Hong, I. Petrunin, and A. Tsourdos. "Integrity Analysis for GPS-Based Navigation of UAVs in Urban Environment". In *Robotics* 9.3 (2020), p. 66.

[37] E. Costa. “Simulation of the Effects of Different Urban Environments on GPS Performance Using Digital Elevation Models and Building Databases”. In *IEEE Transactions on Intelligent Transportation Systems* 12.3 (2011), pp. 819–829.

[38] J. Guo, X. Li, Z. Li, L. Hu, G. Yang, C. Zhao, D. Fairbairn, D. Watson, and M. Ge. “Multi-GNSS Precise Point Positioning for Precision Agriculture”. In *Precision Agriculture* 19.5 (2018), pp. 895–911.

[39] F. Zafari, A. Gkelias, and K. K. Leung. “A Survey of Indoor Localization Systems and Technologies”. In *IEEE Communications Surveys & Tutorials* 21.3 (2019), pp. 2568–2599.

[40] DecaWave Ltd. *Decawave Scensor DW1000 IC*. 2013.

[41] A. Alarifi, A. Al-Salman, M. Alsaleh, A. Alnafessah, S. Al-Hadhrami, M. A. Al-Ammar, and H. S. Al-Khalifa. “Ultra Wideband Indoor Positioning Technologies: Analysis and Recent Advances”. In *Sensors* 16.5 (2016), p. 707.

[42] G. Schroeer. “A Real-Time UWB Multi-Channel Indoor Positioning System for Industrial Scenarios”. In *2018 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. IEEE. 2018, pp. 1–5.

[43] J. Tiemann, A. Ramsey, and C. Wietfeld. “Enhanced UAV Indoor Navigation Through Slam-Augmented UWB Localization”. In *2018 IEEE International Conference on Communications Workshops (ICC Workshops)*. IEEE. 2018, pp. 1–6.

[44] *IEEE 802.15.4-2015, Standard for Low-Rate Wireless Networks*.

[45] A. Mahmood, N. Javaid, and S. Razzaq. “A Review of Wireless Communications for Smart Grid”. In *Renewable and Sustainable Energy Reviews* 41 (2015), pp. 248–260.

[46] M. Baert, J. Rossey, A. Shahid, and J. Hoebeke. “The Bluetooth Mesh Standard: An Overview and Experimental Evaluation”. In *Sensors* 18.8 (2018), p. 2409.

[47] J. F. Schmidt, D. Chernov, and C. Bettstetter. “Towards Industrial Ultra-wideband Networks: Experiments for Machine Vibration Monitoring”. In *IEEE Access* 8 (2020), pp. 42576–42583.

[48] J. F. Schmidt, D. Neuhold, J. Klaue, D. Schupke, and C. Bettstetter. “Experimental Study of UWB Connectivity in Industrial Environments”. In *European Wireless 2018; 24th European Wireless Conference*. VDE. 2018, pp. 1–4.

[49] *IEEE 802.11ax-2021, Standard for Information Technology - Telecommunications and Information Exchange between Systems Local and Metropolitan Area Networks-Specific Requirements*.

[50] H. Brunner, M. Stocker, M. Schuh, M. Schuß, C. A. Boano, and K. Römer. “Understanding and Mitigating the Impact of Wi-Fi 6E Interference on Ultra-Wideband Communications and Ranging”. In *Proceedings of the 21st ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. 2022.

[51] J. Xiao, Z. Zhou, Y. Yi, and L. M. Ni. “A Survey on Wireless Indoor Localization from the Device Perspective”. In *ACM Comput. Surv.* 49.2 (2016), 25:1–25:31.

[52] J. Bauwens, N. Macoir, S. Giannoulis, I. Moerman, and E. De Poorter. "UWB-MAC: MAC Protocol for UWB Localization Using Ultra-Low Power Anchor Nodes". In *Ad Hoc Networks* 123 (2021), p. 102637.

[53] C. Gezer, C. Buratti, and R. Verdone. "Capture Effect in IEEE 802.15.4 Networks: Modelling and Experimentation". In *IEEE* 5^{th} *International Symposium on Wireless Pervasive Computing 2010*. IEEE. 2010, pp. 204–209.

[54] D. Son, B. Krishnamachari, and J. Heidemann. "Experimental Study of Concurrent Transmission in Wireless Sensor Networks". In *Proceedings of the* 4^{th} *International Conference on Embedded Networked Sensor Systems*. 2006, pp. 237–250.

[55] M. Zimmerling, L. Mottola, and S. Santini. "Synchronous Transmissions in Low-Power Wireless: A Survey of Communication Protocols and Network Services". In *ACM Computing Surveys (CSUR)* 53.6 (2020), pp. 1–39.

[56] F. Ferrari, M. Zimmerling, L. Thiele, and O. Saukh. "Efficient Network Flooding and Time Synchronization with Glossy". In *Proceedings of the* 10^{th} *ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. 2011.

[57] P. Dutta, S. Dawson-Haggerty, Y. Chen, C. J. M. Liang, and A. Terzis. "Design and Evaluation of a Versatile and Efficient Receiver-initiated Link Layer for Low-power Wireless". In *Proceedings of the* 8^{th} *ACM Conference on Embedded Networked Sensor Systems (SenSys)*. 2010.

[58] J. Lu and K. Whitehouse. "Flash Flooding: Exploiting the Capture Effect for Rapid Flooding in Wireless Sensor Networks". In *Proceedings of INFOCOM*. 2009.

[59] F. Ferrari, M. Zimmerling, L. Mottola, and L. Thiele. "Low-Power Wireless Bus". In *Proceedings of the* 10^{th} *ACM Conference on Embedded Network Sensor Systems (SenSys)*. 2012. DOI: 10.1145/2426656.2426658.

[60] T. Istomin, A. L. Murphy, G. P. Picco, and U. Raza. "Data Prediction + Synchronous Transmissions". In *Proceedings of the* 14^{th} *ACM Conference on Embedded Network Sensor Systems (SenSys)*. 2016.

[61] O. Landsiedel, F. Ferrari, and M. Zimmerling. "Chaos: Versatile and Efficient All-to-all Data Sharing and In-network Processing at Scale". In *Proceedings of the* 11^{th} *ACM Conference on Embedded Networked Sensor Systems (SenSys)*. 2013.

[62] Texas Instruments. *"CC2420 Datasheet"*.

[63] DecaWave Ltd. *DW1000 User Manual, Version 2.15*. 2017.

[64] M. Z. Win and R. A. Scholtz. "Ultra-Wide Bandwidth Time-Hopping Spread-Spectrum Impulse Radio for Wireless Multiple-Access Communications". In *IEEE Transactions on Communications* 48.4 (2000), pp. 679–689. ISSN: 0090-6778. DOI: 10.1109/26.843135.

[65] F. Ramirez-Mireles and R. A. Scholtz. "Multiple-Access Performance Limits with Time Hopping and Pulse Position Modulation". In *Proceedings of the IEEE Military Communications Conference (MILCOM)*. 1998.

[66] M. Z. Win and R. A. Scholtz. "Impulse Radio: How It Works". In *IEEE Comm. Lett.* 2.2 (1998), pp. 36–38.

[67] *IEEE 802.15.4-2011, Low-Rate Wireless Personal Area Networks (LR-WPANs)*.

[68] DecaWave Ltd. *APH010 DW1000 Inter-Channel Interference: How Transmissions on One DW1000 Channel Can Affect Other Channels and How to Minimize That Effect*. Version 1.1. 2017.

[69] DecaWave Ltd. *APS011 Application Note: Sources of Error in DW1000 Based Two-Way Ranging (TWR) Schemes*. Version 1.1. 2014.

[70] DecaWave Ltd. *DW1000 Data Sheet*. Version 2.12. 2016.

[71] DecaWave Ltd. *Decawave Scensor EVB1000 Evaluation Board*. 2013.

[72] A. Dunkels, B. Grönvall, and T. Voigt. "Contiki - a Lightweight and Flexible Operating System for Tiny Networked Sensors". In *Proceedings of IEEE Workshop on Embedded Networked Sensors (Emnets-I)*. 2004.

[73] `https://github.com/d3s-trento/contiki-UWB`.

[74] B. Großwindhager, C. A. Boano, M. Rath, and K. Römer. "Enabling Runtime Adaptation of Physical Layer Settings for Dependable UWB Communications". In *Proceedings of the IEEE* 19th *International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (Wowmom)*. 2018. DOI: `10.1109/WoWMoM.2018.8449776`.

[75] P. Corbalán and G. P. Picco. "Concurrent Ranging in Ultra-wideband Radios: Experimental Evidence, Challenges, and Opportunities". In *Proceedings of the* 15th *International Conference on Embedded Wireless Systems and Networks (EWSN)*. 2018.

[76] B. Parr et al. "A Novel Ultra-Wideband Pulse Design Algorithm". In *IEEE Communications Letters* 7.5 (2003), pp. 219–221.

[77] M. Ghavami et al. "Novel UWB Pulse Shape Modulation System". In *Wireless Personal Communications* 23.1 (2002), pp. 105–120.

[78] B. Kempke, P. Pannuto, B. Campbell, and P. Dutta. "SurePoint: Exploiting Ultra Wideband Flooding and Diversity to Provide Robust, Scalable, High-Fidelity Indoor Localization". In *Proceedings of the* 14th *ACM Conference on Embedded Network Sensor Systems (SenSys)*. 2016.

[79] C. Stylianopoulos, M. Almgren, O. Landsiedel, and M. Papatriantafilou. "Continuous Monitoring Meets Synchronous Transmissions and In-network Aggregation". In *2019* 15th *International Conference on Distributed Computing in Sensor Systems (DCOSS)*. IEEE. 2019, pp. 157–166.

[80] M. Mohammad and M. C. Chan. "Codecast: Supporting Data Driven In-Network Processing for Low-Power Wireless Sensor Networks". In *Proceedings of the ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. 2018.

[81] R. Lim, R. Da Forno, F. Sutton, and L. Thiele. "Competition: Robust Flooding Using Back-to-Back Synchronous Transmissions with Channel-Hopping". In *Proceedings of the International Conference on Embedded Wireless Systems and Networks (EWSN)*. 2017.

[82] M. Baddeley, A. Stanoev, U. Raza, M. Sooriyabandara, and Y. Jin. "Competition: Adaptive Software Defined Scheduling of Low Power Wireless Networks". In *Proceedings of the International Conference on Embedded Wireless Systems and Networks (EWSN)*. 2019.

[83] P. Corbalán, T. Istomin, and G. P. Picco. "Poster: Enabling Contiki on Ultra-wideband Radios". In *Proceedings of the* 15^{th} *International Conference on Embedded Wireless Systems and Networks (EWSN)*. 2018.

[84] T. Istomin, M. Trobinger, A. L. Murphy, and G. P. Picco. "Interference-Resilient Ultra-Low Power Aperiodic Data Collection". In *Proceedings of the ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. 2018.

[85] B. A. Nahas, S. Duquennoy, and O. Landsiedel. "Network-wide Consensus Utilizing the Capture Effect in Low-power Wireless Networks". In *Proceedings of the ACM Conference on Embedded Networked Sensor Systems (SenSys)*. 2017.

[86] T. Chang, T. Watteyne, X. Vilajosana, and P. H. Gomes. "Constructive Interference in 802.15.4: A Tutorial". In *IEEE Communications Surveys Tutorials* 21.1 (2019), pp. 217–237. DOI: 10.1109/COMST.2018.2870643.

[87] B. A. Nahas, A. Escobar-Molero, J. Klaue, S. Duquennoy, and O. Landsiedel. "BlueFlood: Concurrent Transmissions for Multi-hop Bluetooth 5 - Modeling and Evaluation". In *ACM Transactions on Internet of Things* 2.4 (2021), pp. 1–30.

[88] C.-H. Liao, Y. Katsumata, M. Suzuki, and H. Morikawa. "Revisiting the So-called Constructive Interference in Concurrent Transmission". In *2016 IEEE* 41^{st} *Conference on Local Computer Networks (LCN)*. IEEE. 2016, pp. 280–288.

[89] B. A. Nahas, S. Duquennoy, and O. Landsiedel. "Concurrent Transmissions for Multi-Hop Bluetooth 5". In *Proceedings of the International Conference on Embedded Wireless Systems and Networks (EWSN)*. 2019.

[90] M. Zimmerling, L. Mottola, P. Kumar, F. Ferrari, and L. Thiele. "Adaptive Real-Time Communication for Wireless Cyber-Physical Systems". In *ACM Transaction on Cyber-Physical Systems* (2017).

[91] F. Sutton, R. Da Forno, D. Gschwend, T. Gsell, R. Lim, J. Beutel, and L. Thiele. "The Design of a Responsive and Energy-Efficient Event-Triggered Wireless Sensing System". In *Proceedings of the International Conference on Embedded Wireless Systems and Networks (EWSN)*. 2017.

[92] M. Suzuki, Y. Yamashita, and H. Morikawa. "Low-Power, End-to-End Reliable Collection Using Glossy for Wireless Sensor Networks". In *Proceedings of VTC Spring*. 2013.

[93] M. Brachmann, O. Landsiedel, and S. Santini. "Concurrent Transmissions for Communication Protocols in the Internet of Things". In *Proceedings of the IEEE Conference on Local Computer Networks (LCN)*. 2016.

[94] C. Herrmann, F. Mager, and M. Zimmerling. "Mixer: Efficient Many-to-all Broadcast in Dynamic Wireless Mesh Networks". In *Proceedings of the ACM Conference on Embedded Networked Sensor Systems (SenSys)*. 2018.

[95] R. Jacob, J. Baechli, R. D. Forno, and L. Thiele. "Synchronous Transmissions Made Easy: Design Your Network Stack with Baloo". In *Proceedings of the International Conference on Embedded Wireless Systems and Networks (EWSN)*. 2019.

[96] A. Dunkels, F. Osterlind, N. Tsiftes, and Z. He. "Software-Based on-Line Energy Estimation for Sensor Nodes". In *Proceedings of the Workshop on Embedded Networked Sensors (EmNets)*. 2007.

[97] K. C. Hewage, S. Raza, and T. Voigt. "Protecting Glossy-Based Wireless Networks from Packet Injection Attacks". In *Proceedings of MASS*. 2017.

[98] B. Großwindhager, C. A. Boano, M. Rath, and K. Römer. "Concurrent Ranging with Ultra-Wideband Radios: From Experimental Evidence to a Practical Solution". In *Proceedings of the IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*. 2018. DOI: 10.1109/ICDCS.2018.00149.

[99] P. Corbalán, G. P. Picco, and S. Palipana. "Chorus: UWB Concurrent Transmissions for GPS-like Passive Localization of Countless Targets". In *Proceedings of the 18th International Conference on Information Processing in Sensor Networks (IPSN)*. 2019. DOI: 10.1145/3302506.3310395.

[100] B. Großwindhager, M. Stocker, M. Rath, C. A. Boano, and K. Römer. "SnapLoc: An Ultra-Fast UWB-Based Indoor Localization System for an Unlimited Number of Tags". In *Proceedings of the 18th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. 2019.

[101] T. Wang, H. Zhao, and Y. Shen. "An Efficient Single-Anchor Localization Method Using Ultra-Wide Bandwidth Systems". In *Applied Sciences* 10.1 (2020), p. 57.

[102] Y. Jiang and V. C. M. Leung. "An Asymmetric Double Sided Two-Way Ranging for Crystal Offset". In *Proceedings of ISSSE*. 2007.

[103] M. Ridolfi et al. "Analysis of the Scalability of UWB Indoor Localization Solutions for High User Densities". In *Sensors* 18.6 (2018), p. 1875.

[104] C. McElroy, D. Neirynck, and M. McLaughlin. "Comparison of Wireless Clock Synchronization Algorithms for Indoor Location Systems". In *Proceedings of the IEEE International Conference on Communications Workshops (ICC)*. 2014. DOI: 10.1109/ICCW.2014.6881189.

[105] J. Tiemann, F. Eckermann, and C. Wietfeld. "Multi-User Interference and Wireless Clock Synchronization in TDoA-Based UWB Localization". In *Proceedings of the International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. 2016. DOI: 10.1109/IPIN.2016.7743696.

[106] J. Tiemann, F. Eckermann, and C. Wietfeld. "ATLAS - An Open-Source TDoA-Based Ultra-Wideband Localization System". In *Proceedings of the International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. 2016. DOI: 10.1109/IPIN.2016.7743707.

[107] A. Sgora, D. J. Vergados, and D. D. Vergados. "A Survey of TDMA Scheduling Schemes in Wireless Multihop Networks". In *ACM Computing Surveys (CSUR)* 47.3 (2015), pp. 1–39.

[108] J. Yeo, H. Lee, and S. Kim. "An Efficient Broadcast Scheduling Algorithm for TDMA Ad-Hoc Networks". In *Computers & Operations Research* 29.13 (2002), pp. 1793–1806.

[109] R. B. Langley. "Dilution of Precision". In *GPS World* 10.5 (1999), pp. 52–59.

[110] S. Angarano, V. Mazzia, F. Salvetti, G. Fantin, and M. Chiaberge. "Robust Ultra-wideband Range Error Mitigation with Deep Learning at the Edge". In *Engineering Applications of Artificial Intelligence* (2021).

[111] V. Barral, C. J. Escudero, J. A. García-Naya, and R. Maneiro-Catoira. "NLOS Identification and Mitigation Using Low-Cost UWB Devices". In *Sensors* 19.16 (2019), p. 3464.

[112] K. Yu, K. Wen, Y. Li, S. Zhang, and K. Zhang. "A Novel NLOS Mitigation Algorithm for UWB Localization in Harsh Indoor Environments". In *IEEE Transactions on Vehicular Tech.* (2018).

[113] C. Di Franco, A. Prorok, N. Atanasov, B. Kempke, P. Dutta, V. Kumar, and G. J. Pappas. "Calibration-Free Network Localization Using Non-Line-of-Sight Ultra-Wideband Measurements". In *Proceedings of the ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. 2017.

[114] J. Tiemann, Y. Elmasry, L. Koring, and C. Wietfeld. "ATLAS Fast: Fast and Simple Scheduled TDoA for Reliable Ultra-Wideband Localization". In *Proceedings of the International Conference on Robotics and Automation (ICRA)*. 2019. DOI: 10.1109/ICRA.2019.8793737.

[115] Bitcraze. *The Loco Positioning System*. Online; last accessed May 25, 2019. 2019. URL: https://www.bitcraze.io/loco-pos-system/.

[116] F. Giopp. "Real-time Tracking of Multiple Users Using UWB: A TDoA-based Approach". M.Sc. Thesis. University of Trento, 2018.

[117] S. Wang, Z. Bao, J. S. Culpepper, and G. Cong. "A Survey on Trajectory Data Management, Analytics, and Learning". In *ACM Computing Surveys (CSUR)* 54.2 (2021), pp. 1–36.

[118] Y. Zheng and X. Zhou. *Computing with Spatial Trajectories*. Springer Science & Business Media, 2011.

[119] R. H. Güting, M. H. Böhlen, M. Erwig, C. S. Jensen, N. A. Lorentzos, M. Schneider, and M. Vazirgiannis. "A Foundation for Representing and Querying Moving Objects". In *ACM Transactions on Database Systems (TODS)* 25.1 (2000), pp. 1–42.

[120] E. Zimányi, M. Sakr, and A. Lesuisse. "MobilityDB: A Mobility Database Based on PostgreSQL and PostGIS". In *ACM Transactions on Database Systems (TODS)* 45.4 (2020), pp. 1–42.

[121] Y. Zheng. "Trajectory Data Mining: an Overview". In *ACM Transactions on Intelligent Systems and Technology (TIST)* 6.3 (2015), pp. 1–41.

[122] C. Parent, S. Spaccapietra, C. Renso, G. Andrienko, N. Andrienko, V. Bogorny, M. L. Damiani, A. Gkoulalas-Divanis, J. Macedo, N. Pelekis, et al. "Semantic Trajectories Modeling and Analysis". In *ACM Computing Surveys (CSUR)* 45.4 (2013), pp. 1–32.

[123] S. Spaccapietra, C. Parent, M. L. Damiani, J. A. de Macedo, F. Porto, and C. Vangenot. "A Conceptual View on Trajectories". In *Data & Knowledge Engineering* 65.1 (2008), pp. 126–146.

[124] Y. Yoshimura, A. Krebs, and C. Ratti. "Noninvasive Bluetooth Monitoring of Visitors' Length of Stay at the Louvre". In *IEEE Pervasive Computing* 16.2 (2017), pp. 26–34.

[125] C. Martella, A. Miraglia, J. Frost, M. Cattani, and M. van Steen. "Visualizing, Clustering, and Predicting the Behavior of Museum Visitors". In *Pervasive and Mobile Computing* 38 (2017), pp. 430–443.

[126] A. Kontarinis, K. Zeitouni, C. Marinica, D. Vodislav, and D. Kotzinos. "Towards a Semantic Indoor Trajectory Model: Application to Museum Visits". In *GeoInformatica* 25.2 (2021), pp. 311–352.

[127] P. Centorrino, A. Corbetta, E. Cristiani, and E. Onofri. "Managing Crowded Museums: Visitors Flow Measurement, Analysis, Modeling, and Optimization". In *Journal of Computational Science* 53 (2021), p. 101357.

[128] L. Bermingham and I. Lee. "A Probabilistic Stop and Move Classifier for Noisy GPS Trajectories". In *Data Mining and Knowledge Discovery* 32.6 (2018), pp. 1634–1662.

[129] Y. Zheng, L. Zhang, Z. Ma, X. Xie, and W.-Y. Ma. "Recommending Friends and Locations Based on Individual Location History". In *ACM Transactions on the Web (TWEB)* 5.1 (2011), pp. 1–44.

[130] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, et al. "A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise." In *Kdd*. Vol. 96. 34. 1996, pp. 226–231.

[131] M. L. Damiani, F. Hachem, H. Issa, N. Ranc, P. Moorcroft, and F. Cagnacci. "Cluster-based Trajectory Segmentation with Local Noise". In *Data Mining and Knowledge Discovery* 32.4 (2018), pp. 1017–1055.

[132] H. A. Blom and Y. Bar-Shalom. "The Interacting Multiple Model Algorithm for Systems with Markovian Switching Coefficients". In *IEEE Transactions on Automatic Control* (1988).

[133] J. H. Kang, W. Welbourne, B. Stewart, and G. Borriello. "Extracting Places from Traces of Locations". In *ACM SIGMOBILE Mobile Computing and Communications Review* 9.3 (2005), pp. 58–68.

[134] M. Buchin, A. Driemel, M. Van Kreveld, and V. Sacristán. "Segmenting Trajectories: A Framework and Algorithms Using Spatiotemporal Criteria". In *Journal of Spatial Information Science* 3 (2011), pp. 33–63.

[135] A. T. Palma, V. Bogorny, B. Kuijpers, and L. O. Alvares. "A Clustering-based Approach for Discovering Interesting Places in Trajectories". In *Proceedings of the 2008 ACM Symposium on Applied Computing*. 2008, pp. 863–868.

[136] S. Hwang, C. VanDeMark, N. Dhatt, S. V. Yalla, and R. T. Crews. "Segmenting Human Trajectory Data by Movement States While Addressing Signal Loss and Signal Noise". In *International Journal of Geographical Information Science* 32.7 (2018), pp. 1391–1412.

[137] T. Luo, X. Zheng, G. Xu, K. Fu, and W. Ren. "An Improved DBSCAN Algorithm to Detect Stops in Individual Trajectories". In *ISPRS International Journal of Geo-Information* 6.3 (2017), p. 63.

[138] Z. Li, J. Han, M. Ji, L.-A. Tang, Y. Yu, B. Ding, J.-G. Lee, and R. Kays. "Movemine: Mining Moving Object Data for Discovery of Animal Movement Patterns". In *ACM Transactions on Intelligent Systems and Technology (TIST)* 2.4 (2011), pp. 1–32.

[139] S. Hwang, C. Evans, and T. Hanke. "Detecting Stop Episodes from GPS Trajectories with Gaps". In *Seeing Cities Through Big Data*. Springer, 2017, pp. 427–439.

[140] F. Gottifredi and E. Varriale. *Navigation System for Exploring And/Or Monitoring Unknown and/or Difficult Environments*. US Patent 8,473,118. 2013.

[141] N. Janicijevic, P. Corbalán, T. Istomin, G. P. Picco, and E. Varriale. "Small Plans Towards Mars: Exploiting Ultra-Wideband for Self-Localizing Rover Navigation". In *Proceedings of the International Conference on Embedded Wireless Systems and Networks*. 2018, pp. 199–200.

[142] J. Zhu and S. S. Kia. "A SPIN-Based Dynamic TDMA Communication for a UWB-Based Infrastructure-free Cooperative Navigation". In *IEEE Sensors Letters* 4.7 (2020), pp. 1–4.

[143] N. Macoir, J. Bauwens, B. Jooris, B. Van Herbruggen, J. Rossey, J. Hoebeke, and E. De Poorter. "UWB Localization with Battery-powered Wireless Backbone for Drone-based Inventory Management". In *Sensors* 19.3 (2019), p. 467.

[144] *Rover Wheels - NASA Mars*. `https://mars.nasa.gov/mars2020/spacecraft/rover/wheels/`, accessed 2022-05-05.

[145] DecaWave Ltd. *DWM1001 Battery Lifetime Estimation*. 2019.