Towards Robust General-purpose Backscatter with Commodity Radios

by

Si Chen

M.S., Simon Fraser University, 2018
LL.M., Tsinghua University, 2011
B.B.A., China University of Geosciences, 2007

Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of
Doctor of Philosophy

in the
School of Computing Science
Faculty of Applied Sciences

Declaration of Committee

Name:	**Si Chen**
Degree:	**Doctor of Philosophy**
Thesis title:	**Towards Robust General-purpose Backscatter with Commodity Radios**
Committee:	**Chair:** Ouldooz Baghban Karimi Lecturer, Computing Science
	Jiangchuan Liu Supervisor Professor, Computing Science
	Jiannan Wang Committee Member Associate Professor, Computing Science
	Jie Liang Examiner Professor, Engineering Science
	Amiya Nayak External Examiner Professor, Electrical Engineering & Computer Science (SEECS) University of Ottawa

Abstract

Over the past decade, backscatter nodes have received booming interest for many emerging mobile applications, such as sports analytics and interactive gaming. However, problems remain in making backscatter communication into general-purpose battery-free data transfer for IoTs. First, backscatter networks are not ready to provide a high-throughput and stable communication platform for billions of such mobile nodes. The common mapping paradigm that chooses the optimal rate based on RSSIs is hardly adaptable to hardware diversity, and the current probing processes are not optimized for mobile scenarios due to inefficient probing trigger. Second, although backscatter communication can already be achieved with ambient signals, the current system suffers from several key issues, including redundant modulation, productive-data dependency, and lack of interference countermeasures.

To address those issues, we propose a mobility-aware rate adaptation link-layer that fully exploits the mobility hints from PHY information to deliver a high-throughput link-layer for mobile backscatter networks. The key insight is that mobility-hints can greatly benefit link-layer design, including rate selection and channel probing. Moreover, we introduce robust modulation designs with ambient signals, e.g., BLE, WiFi. Specifically, we propose direct frequency shift modulation with the single tone generated by an excitation BLE device, making robust single-bit modulation possible. Besides, we present a novel backscatter modulation design that can take uncontrolled OFDM WiFi signals as excitations and efficiently embeds tag data at the single-symbol rate.

The prototype is implemented using COTS RFID readers, commodity radios, commercial RFID tags, and customized tags with FPGAs. Our extensive experiments demonstrate that the mobility-aware link-layer achieves up to 3.8x throughput gain over the state-of-the-art methods across a wide range of mobility, channel conditions, and tag types. Besides this, the BLE modulation design achieves more than 17x uplink goodput gains over FreeRider under indoor and outdoor environments. And the maximum throughput of our WiFi modulation design is 3.92x and 1.97x better than FreeRider and MOXcatter.

Keywords: Backscatter networks; link layer; modulation design; RFID; commodity radio; tags

Acknowledgements

Foremost, I would like to express my sincere gratitude to my supervisor Dr. Jiangchuan Liu for the continuous support of my study and research, for his patience, enthusiasm, and immense knowledge. I could not have imagined having a better advisor and mentor for my study. I would also like to extend my thanks to the rest of my thesis committee members: Dr. Jiannan Wang, Dr. Jie Liang, Dr. Amiya Nayak and Dr. Ouldooz Baghban Karimi for their encouragement, insightful comments, and valuable input. My sincere thanks also go to all my fellow labmates for their great support and encouragement. Last but not least, I appreciate my family for their unconditional love and support.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Backscatter technology

In recent years, backscatter communications have been dramatically revolutionizing traditional active-radio sensors [36] as they can provide a battery-free, small form-factor, and cheap alternative while achieving comparable sensing capabilities. Various novel new backscatter nodes have been designed and applied in a range of mobile applications [69, 90, 80, 30, 31, 35]. For example, [63] presents a backscatter design to constantly check the pressure for sports balls. [26] proposes a tag-free solution to monitor human activities for elderly healthcare. Even though researchers have made decent progress in backscatter communication, we observe that backscatter networks that promise to deliver a high-throughput interconnected platform are not well prepared. Several advances, like rateless coding [76] and parallel decoding [43], have been proposed to enhance backscatter communication throughput significantly, but they are incompatible with existing standards and thus leave billions of deployed backscatter nodes, like RFID tags, behind.

Meanwhile, to get rid of the dependence on the dedicated excitation signal generator (such as RFID readers), a seminal work, ambient backscatter [54], begins a new era of using ambient signals as excitation. Since then, many researchers work along this line and various backscatter systems have emerged, e.g., WiFi backscatter [17, 88, 85, 82, 40, 22, 23, 55, 16], Bluetooth backscatter [25, 87, 45, 47, 53], ZigBee backscatter [78], LoRa backscatter [42, 69], FM backscatter [73], LTE backscatter[21], and others [27, 28, 66, 50, 71, 56, 44, 72, 81, 83]. BackFi [17] provides a high-throughput WiFi backscatter system using full-duplex techniques. WiFi-backscatter uses CSI/RSSI modulation to transmit tag data and achieves rates of up to 1 kbps. The common goal is to make backscatter communication into general-purpose battery-free data transfer for IoTs.

Thanks to widespread Bluetooth and WiFi radios in our daily life, e.g., smartphones, speakers, and access points, Bluetooth-based and WiFi-based backscatter systems have received ever-increasing interest for embedded electronics. FS-backscatter [88] can successfully demodulate backscattered BLE (Bluetooth Low Energy) signals, but requires hacking into a specific chip and thus is not a general solution. Interscatter [46] novelly presents how to backscatter BLE signals into Zigbee and WiFi but fails to interface with Bluetooth receivers. BLE-backscatter [25] introduces how to produce BLE backscatter signals using a dedicated continuous wave (CW) generator. None of the above provides a backscatter solution that is completely compatible with commercial Bluetooth radios.

To get closer to universal backscatter, some researchers also explore new backscatter systems that can work with uncontrolled ambient signals. The packet-level WiFi solutions are proposed first. The idea is that tag data can be embedded in the signal-strength changes of packets [88, 48]. However, such methods suffer from low data rates, typically ∼1 kbps, since a packet can only carry one bit and they are susceptible to environmental changes. The most recent advances have increased the data rate to some extent. FreeRider [87] and MOXcatter [89], can provide symbol-level data transmissions for tags based on ambient OFDM WiFi signals, realizing about 60 kbps. To ensure decent BERs for the receiver, most prior arts employ multi-symbol modulation for tag data. Although backscatter communication can already be achieved with ambient signals, they suffer from several issues, including redundant modulation, low data rates, and controlled excitation signals.

1.2 Background

1.2.1 Rate Adaptation

Although state-of-the-art systems have proposed several advances, like rateless coding and parallel decoding to enhance backscatter communication throughput significantly, they are incompatible with existing standards. We discuss rate adaptation in the following three aspects.

Mapping based: Usually, the common assumption of mapping-based schemes is that the best rate is highly related to SNR metrics, e.g., BER, RSSI. Along this line, a number of proposals are designed according to different SNR metrics. FARA [62] is designed for OFDM protocols, which allows each subcarrier to choose the best modulation and code rate. Its rate selection module contains an SNR-rate mapping table. Differently, ESNR [41] proposes a delivery model by leveraging CSIs to deal with frequency-selective fading. Also, it employs effective SNR, a new metric for looking up optimal rates in a table. Blink [86] and CARA [33] share similar ideas and exploit both the RSSI and loss rate to combat multipath self-interference. Yet, both of them cannot deal with hardware diversity [84, 31].

Throughput based: In contrast, throughput-based methods can be made universal and robust for different environments. Besides Minstrel [8] and Ath9k [3] from commercial implementations, the most famous solution of this category is SampleRate [18]. However, it cannot make timely responses to location changes because of unawareness of mobility states. To address this challenge, a mobility-assisted rate adaptation is designed [68]. Unfortunately, its coarse-grained mobility hints do not work for backscatter scenarios.

New Backscatter Designs: There are also some works trying to overhaul the current network layers of the C1G2 standard [59]. Buzz [76] innovatively introduces a new PHY layer based on rateless coding and compressive sensing and achieves significant throughput gains. Flit [39] designs a new MAC protocol for computational RFIDs by exploiting burst transmission and uses duty-cycling to achieve power efficiency. Yet, such designs are not standard-compatible. In contrast, our proposed system, MobiRate, is readily available to benefit tons of currently deployed RFIDs that comply with the C1G2 standard.

In summary, inspired by the prior works, our work's novelty lies in obtaining real-time fine-grained mobility-hints and using these hints to design an efficient and C1G2-compatible solution exclusively for backscatter networks.

1.2.2 BLE Backscatter

Our work is inspired by recent progress on backscatter systems. The closely related work and corresponding differences are discussed as follows.

FreeRider[87] extends codeword translation to Bluetooth technologies using two-step modulation. As shown in Section 3.2, it suffers from serious problems and cannot make robust modulation. Although FreeRider is the first full BLE backscatter system, it requires two BLE receivers to decode while we only need one. FS-backscatter [88] modulates tag data on Bluetooth exciting signal through amplitude-shift keying (ASK) modulation so it requires hacking into a specific Bluetooth chip to demodulate the amplitude information. Interscatter [46] enables WiFi backscatter through reversely-whitened BLE signals. It proposes reversely-whitening techniques to generate the BLE single-tone. This inspires us to generate a single-tone as the carrier for BLE backscatter. Although our single-tone generation uses reversely-whitening techniques proposed by Interscatter, Interscatter's goal is totally different from ours. Specifically, Interscatter intends to generate WiFi 802.11b signals using DBPSK modulation while we need to regenerate BLE signals using BFSK modulation.

[24] first fine-tunes and applies BFSK modulation to BLE backscatter. BLE-backscatter [25], an expanded version of [24], improves the tag implementation by changing the laboratory instruments including the arbitrary waveform generator to MSP430. BLE-Backscatter [25] requires a specialized CW generator while we use commodity BLE devices, reducing the deployment cost of the system. Moreover, the specialized CW generator solution can completely jam a BLE channel and has no downlink capability whereas we employ an envelope detector to identify exciting BLE signals and to receive commands from excitation BLE devices. As far as the modulation part is concerned, the changes we made compared to BLE-backscatter [25] are: changing the carrier generator from a specialized device to a commodity BLE device and conducting BER performance evaluations to find the optimal modulation index for BLE backscatter. [64] applies BLE-Backscatter [25] to uplink for wireless neural recording. NeuroDisc BLE-compatible backscatter [65] improves BLE

Backscatter [25] by introducing single sideband (SSB) backscatter and CPFSK to BLE backscatter. It improves the spectral efficiency of BLE backscatter, but still shares the same limitations as BLE backscatter. Nevertheless, none of the above systems had in-depth examined backscatter reliability issues. Relacks [47] proposes a closed-loop configuration selection algorithm that uses frequency, antenna, and wake-up source diversity to deliver reliable transmissions in indoor environments. Relacks implements frequency reconfiguration through channel reconfiguration of the BLE transceiver and the tag simply frequency shift by a fixed amount. In this way, the tag does not need to have the capability of dynamic channel configuration. System complexity migrates to the BLE transceiver.

Inspired by prior work, we build RBLE using only commodity BLE radios and provide direct frequency shift modulation, dynamic channel reconfiguration, and adaptive encoding techniques to improve backscatter reliability.

1.2.3 WiFi Backscatter

WiFi backscatter has attracted much attention since WiFi excitations are ubiquitous, e.g., in offices, homes, and malls. Researchers have made considerable effort to improve throughput and ranges of WiFi backscatter systems [48, 87, 89, 85, 49, 17, 88]. Early WiFi backscatter systems suffer from extremely low rates due to packet-level modulation, such as FS-Backscatter [88] and WiFi Backscatter [48]. [48] attempts to piggyback information on top of WiFi signals at a packet level and achieves 1 kbps and 2.1 m uplink range. Later, the focus shifts to the sub-packet level and many novel solutions are proposed. By using a dedicated device generating continuous waves or specialized hardware, much higher rates are achieved, e.g., [17, 49]. [49] employs a dedicated CW generator to backscatter 802.11b packets using 4-5 orders of magnitude lower power than normal WiFi. Subsequently, backscatter systems with commodity radios receive significant interest since it is easy to deploy [42, 85, 87, 46, 89, 90]. Hitchhike [85] and FreeRider [87] leverage novel codeword translation to build commodity WiFi backscatter systems for 802.11b and 802.11n. MOXcatter further provides the design of how to backscatter spatial stream WiFi [89], and X-Tandem introduces the first multi-hop backscatter paradigm that enables multiple tags to transmit sensor data on the same carrier packet [90], attempting to combine PHY and routing design together; therefore large-scale and high-throughput backscatter networks can be made possible. The modulation schemes of the WiFi backscatter tags [85, 49, 87, 89] are mainly based on Phase Shift Keying (PSK) modulation because 802.11b WiFi uses DBPSK modulation, while 802.11n WiFi uses QAM modulation that combines phase and amplitude modulation. Multiscatter [37] is the first work to identify different protocols and pave the way for leveraging various excitation signals. However, those WiFi backscatter solutions cannot deal with single-symbol decoding with uncontrolled OFDM WiFi. In contrast, our work identifies the root cause of incompatibility between OFDM WiFi operations and codeword translation and proposes novel deinterleaving-twins decoding that works perfectly on the single-symbol level.

Other backscatter paradigms. Meanwhile, there are lots of backscatter systems that take advantage of popular RF signals. Some works propose to backscatter LoRa packets for long-range low-power communication [69, 71]. Apart from digital backscatter, FM backscatter [73] and HD backscatter [57] present how to backscatter analog signals, opening the door for low-power audio and video streaming. Yet, those backscatter excitations are not as popular and abundant as OFDM WiFi. We intend to investigate the interoperability between OFDM WiFi and other RF sources in the future.

1.3 Motivation

In this thesis, we target at delivering a fast and reliable link layer and modulation design for mobile backscatter networks (MBN) in a standard-compatible way, which can benefit tons of currently deployed backscatter devices. To achieve this, however, there are several key challenges:

Hardware-Dependent Rate Selection. Rate adaptation is of high importance to all kinds of wireless networks. Ideally, a good rate adaptation method should timely choose the optimal rate to maximize network throughput based on changing channel conditions that are time-varying and location-dependent. Although prior methods [86, 33] can adapt rates through a well-trained 2D map, they suffer from hardware-dependent problems. We observe that those trained maps are drastically different from tag to tag, and thus they would experience severe performance degradation when applying one well-trained map to other types of tags. The root cause is that the tags are from various vendors or designed for diverse usages, which inevitably exhibit diverse signal responses due to different antenna designs, circuit structures, and manufacturing processes.

Probing cost is a critical factor in throughput optimization. Backscatter networks have a unique characteristic, self-interference. When it happens, both the received signal strength indicator (RSSI) and loss rates of received signals are high, which is quite different from normal cases: high RSSIs indicate low loss rates. Therefore, the unique self-interference problem makes obtaining accurate channel conditions even more difficult [86].

Unreliable modulation. The requirement for specialized and expensive readers has long bedeviled this near zero-power technology's wider adoption. Therefore, a bunch of new backscatter paradigms that work with commodity radios have been proposed recently [42, 69, 51, 52, 91, 70, 37]. For example, Passive WiFi [49] can decode backscattered WiFi signals using commercial WiFi radios. BLE-backscatter [25] introduces how to produce BLE backscatter signals using a dedicated continuous wave (CW) generator. The most recent work, FreeRider [87], for the first time realizes this goal with only commodity BLE radios. However, several key reliability issues hinder the progress of interacting with commodity radios, including unreliable modulation, productive-data dependency, and lack of interference countermeasures. For example, to be compatible with BLE signals, FreeRider employs two-step modulation. It first shifts the exciting signal to the target channel, e.g., 6 MHz frequency shift, and then uses an additional frequency shift, e.g., 500 kHz, to do codeword translation. Doing so would inevitably introduce unreliability due to self-interference or no solid signal in the target frequency. Hence, a new modulation scheme that does not rely on codeword translation is needed to fix this problem.

Redundant modulation. Although backscatter communication can already be achieved with ambient signals, a controlled excitation signal is required. For the systems based on controlled excitations, the dedicated device sends single tones as carriers, and tags generate standard packets using backscatter, e.g., WiFi [49], LoRa [69], ZigBee [46]. The advantage of such a system is that it can deliver standard-compliant data rates for low-power tags. But generally, the widespread signals in life are not continuous waves, so these systems can't reuse the existing ambient signals carrying productive data. Some researchers explore new backscatter systems that can work with uncontrolled ambient signals. The packet-level WiFi solutions are proposed first[88, 48], but suffer from low data rates. FreeRider [87] and MOXcatter [89], can provide symbol-level data transmissions for tags based on ambient OFDM WiFi signals, realizing about 60 kbps. However, to ensure decent BERs for the receiver, they employ multi-symbol modulation for tag data. Such redundancy is introduced to combat the nonlinearity brought by tag modulation. When using every single OFDM symbol for tag modulation, those systems fail due to highly unstable demodulation errors.

1.4 Thesis Contribution

This thesis tackles these challenges and seeks to enable robust general-purpose backscatter with commodity radios.

1.4.1 Contribution Summary

The key contributions of this thesis are as follows:

- MobiRate - A Mobility-aware link layer for rate adaptation in backscatter networks. MobiRate achieves overall throughput gains of 3.8x over Blink and 2.9x over CARA on average.

- RBLE - A robust BLE backscatter modulation design that works with an excitation BLE device and a single BLE receiver. RBLE achieves more than 17.3x and up to 78.8x goodput gains in LoS cases and achieves more than 17.1x and up to 66.0x goodput gains in NLoS scenarios.

- RapidRider - A novel WiFi backscatter design that can take uncontrolled OFDM WiFi signals as excitations and efficiently embeds tag data at the single-symbol rate.

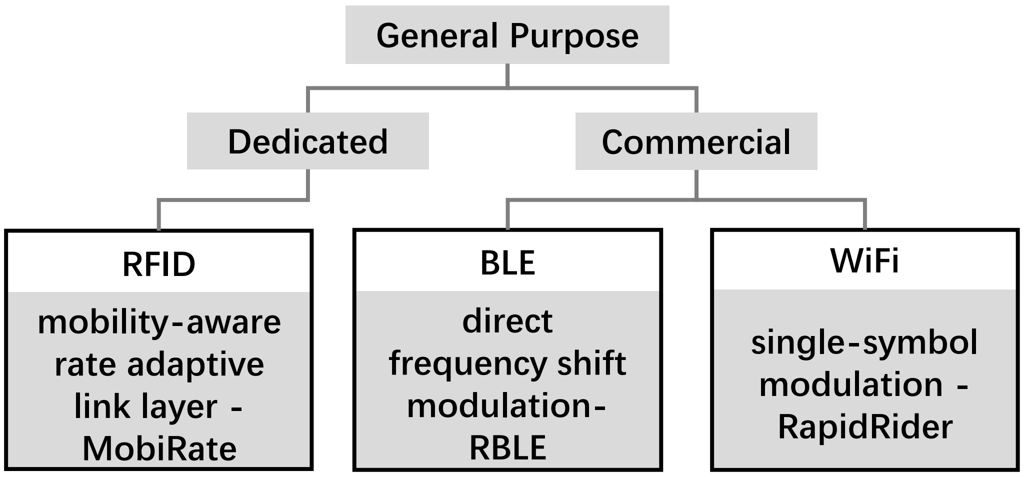

General Purpose
Dedicated
Commercial
RFID
mobility-aware rate adaptive link layer - MobiRate
BLE
direct frequency shift modulation- RBLE
WiFi
single-symbol modulation - RapidRider

Figure 1.1: An overview of thesis.

The maximum throughput of RapidRider is 3.92x and 1.97x better than FreeRider and MOXcatter.

1.4.2 Thesis Overview

A framework of the thesis is shown in Fig. 1.1. This framework includes three components to address the three key challenges of backscatter systems.

1.4.3 Rate adaptive approach for link layer - MobiRate

In chapter 2, we propose a Mobility-aware Rate adaptation method using PHY information for backscatter networks. The whole design eliminates hardware dependency by introducing a throughput-based rate adaptation framework and extensively uses fine-grained mobility hints to optimize probing processes. It has four major components. First, it introduces a velocity-based loss-rate estimation, which makes use of PHY information to deduce tag velocity, resulting in more accurate packet-loss estimation based on mobility history. Second, it employs a mobility-assisted probing trigger that uses both position and direction to considerably reduce unnecessary probes. Third, it designs a selective probing method that novelly leverages the built-in system command to enable per-tag probing, eliminating potential MAC collisions. Fourth, a hybrid self-interference detection scheme is proposed to combat high-loss cases where the received strong RSSIs do not come with low packet losses. The design philosophy is to achieve high throughput while being compatible with standard and commercial RFID readers. We prototype MobiRate using a Thingmagic M6e reader and evaluate it with 12 types of tags across different vendors. MobiRate achieves overall throughput gains of 3.8x over Blink and 2.9x over CARA on average.

1.4.4 Reliable backscatter design with BLE - RBLE

In chapter 3, we propose a robust BLE backscatter design that works with an excitation BLE device and a single BLE receiver. Upon receiving signals from excitation BLE devices, the tag modulates sensor data onto them and backscatters new BLE packets. Prior BLE backscatter system FS-backscatter [88], Interscatter [46], BLE-backscatter [25] can not provide a backscatter solution that is completely built by commercial Bluetooth radios. FreeRider [87], for the first time realizes this goal with only commodity BLE radios but suffers from unreliable two-step modulation. To address this problem, RBLE uses BLE signals with partial single tones as excitations and finds the optimal modulation index, enabling robust BLE packet regeneration bit-by-bit. Compared to FreeRider, RBLE achieves more than 17.3x and up to 78.8x goodput gains in LoS cases and achieves more than 17.1x and up to 66.0x goodput gains in NLoS scenarios.

1.4.5 Single-symbol backscatter with WiFi - RapidRider

In chapter 4, We present RapidRider, the first WiFi backscatter design that can take uncontrolled OFDM WiFi signals as excitations and efficiently embeds tag data at the single-symbol rate. We believe such a design brings us closer to pervasive backscatter communications. Thanks to RapidRider, for the first time, we can use the abundant uncontrolled WiFi signals around us freely for backscatter. To do so, we first explain why previous systems fail in single-symbol backscatter and then propose our solution based on deinterleaved data. Further, a deinterleaving-twins decoding scheme, incorporating a forward deinterleaver and a backward deinterleaver, is designed to enable using uncontrolled excitations. Moreover, we deeply explore how the MCS of excitation affects tag demodulation, and further design and discuss several translation methods for the first time. To break the two-receiver limit, we introduce a novel aggregated transmission mechanism that allows productive data and tag data on the same packet. The key insight is to take the pilot symbol as the reference symbol for tag data demodulation; thus a single receiver is adequate for decoding both data.

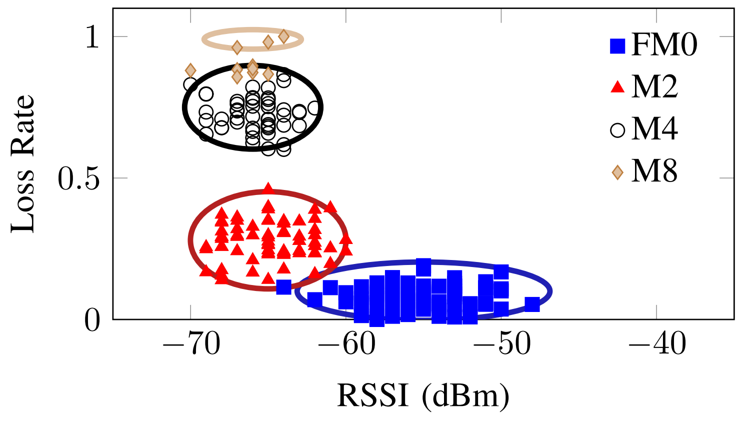
Chapter 2

A Mobility-aware Link Layer Design

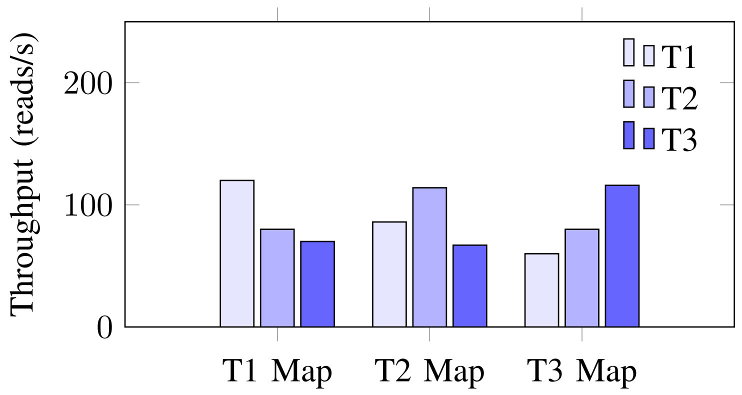
2.1 Background and Motivation

2.1.1 Backscatter Primer

In contrast to active radios, a backscatter link includes a downlink (Reader-to-Tag) and an uplink (Tag-to-Reader). As the capabilities of tags, e.g., computations and storage, are quite limited, the downlink usually adopts a simple amplitude modulation, Pulse Interval Encoding (PIE), which can be decoded using simple circuits on tags. In C1G2, the length of a bit '0' is defined as Tari (Type A Reference Interval), and the length of a bit '1' is between 1.5Tari and 2Tari. As PIE is the only encoding scheme for downlinks, the downlink rate is solely determined by Tari values. C1G2 specifies three options for Tari values, 6.25, 12.5, and 25 μs, corresponding to the maximum downlink rates, 160, 80, and 40 kbps if all the downlink bits are '0's. If more bit '1's are included, the practical rate would be a bit lower. On the contrary, the decoding ability of the reader is strong, which allows more flexible encoding schemes for uplinks. C1G2 describes four encoding schemes for uplinks, FM0, Miller2, Miller4, Miller8 [1]. The uplink rate can be configured by the encoding scheme and BLFs (Backscatter Link Frequency). For example, if BLK=250 kHz, the uplink rates are 250, 125, 67, and 34 kbps where FM0, M2, M4, and M8 are used, respectively. *Note that all the above configurable parameters, including Tari, uplink encoding, BLF, are completely controlled by the reader.* C1G2 also includes various frame formats and commands between the reader and tags, such as Query, ACK, and RN16. Basically, what the tag needs to do is follow the reader's commands and respond accordingly.



(a) Rate selection map trained using a Higgs 3 tag. The BLF is fixed at 250 kHz for simplicity



(b) Throughput comparison using different maps

Figure 2.1: Experiments showing the hardware diversity across 7 types of tags: T1-High point piano tag; T2-SMARTRAC DogBone; T3-ImpinJ Monza 4D; T4-Alien Higgs 3; T5-Vulcan folded tag; T6-Vulcan Windshield tag; T7-confidex steelwave tag.

2.1.2 Observation

Rate adaptation is of high importance to all kinds of wireless networks. Ideally, a good rate adaptation method should choose the optimal rate to maximize network throughput based on changing channel conditions that are time-varying and location-dependent. Key issues of rate adaptation include channel estimation and rate selection. In other words, how to obtain accurate channel status and then choose the optimal rate for it.

In mobile backscatter networks (MBN), Blink [86] is the first work that proposes to exploit both the loss rate and RSSI as reliable channel measurements because RSSIs alone are not adequate and may lead to inaccurate prediction due to multipath self-interference. Thus, Blink builds a rate selection map to select the optimal rate according to the probed

[1]We use M2/M4/M8 to denote Miller2, Miller4, and Miller8.

Table 2.1: RSSIs and loss rates for 7 types of tags at the same location where FM0 is the optimal rate: T1-High point piano tag; T2-SMARTRAC DogBone; T3-ImpinJ Monza 4D; T4-Alien Higgs 3; T5-Vulcan folded tag; T6-Vulcan Windshield tag; T7-confidex steelwave tag.

	T1	T2	T3	T4	T5	T6	T7
RSSI(dBn)	-55	-39	-48	-43	-44	-43	-61
Loss Rate (%)	99	90	94	93	99	95	72

loss rate and RSSI, as shown in Figure 2.1a. CARA [33] follows this approach and makes it more universal by utilizing both spatial and frequency diversity. While these two work well in their proposals, we observe that such mapping-based methods can experience significant performance loss when faced with various types of tags.

We first investigate how channel measurements and optimal rates are related across different types of tags. In this experiment, we test seven kinds of tags from different manufacturers [2]. Table 2.1 depicts a group of results averaged over 20 traces where all the tags are tested on the same spot and the optimal rate is FM0. The only difference is the tag type. From those results, we observe that *the channel measurements, including loss rates and RSSIs, differ significantly across diverse tags, even for almost the same channel quality,* indicated by the optimal rate (FM0) and the same location. Specifically, for T1 and T5, they have the same loss rates. Yet, the RSSI gap is -44-(-55)=11, which is huge. Moreover, the ranges of RSSIs and loss rates are considerably wide, namely, RSSIs$\in$[-61,-39], loss rates$\in$[0.72,0.99]. The root cause for this phenomenon is that those tags have diverse signal responses due to hardware diversity, including different antennas, circuits, and manufacturing processes. For example, T7, a confidex steelwave tag designed for mounting on steel, has the best loss rate. It has an IP67 rating, which means it can be protected from dust and is capable of withstanding water immersion between 15 cm and 1 meter for 30 minutes. We consistently observe the same phenomenon through similar experiments in different environments.

To further examine how hardware diversity impacts network throughput, we select three tags as a group, including T1, T2, and T3. Then we adopt Blink's map learning algorithm and apply the maps learned from T1, T2, and T3 against each other. The results averaged across 50 different locations are shown in Figure 2.1b. We have two observations here. First, the tag has the best performance using its own trained map. Specifically, T1, T2, and T3 achieve 120, 114, and 116 reads/s in this case. Second, the throughput degrades remarkably when the maps are from others. For example, the reading rate of T3 drops from 116 reads/s to 60 reads/s using T1-trained map.

[2]The specific types of tags are included in the caption of Figure 2.1

From the above experiments, we know that mapping-based methods are difficult to deal with different hardware-dependent channel measurements and thus could degrade network throughput significantly.

2.1.3 Contribution

We make the following contributions:

- We present a novel rate adaptation framework that efficiently uses PHY information and interference awareness for MBN.
- We introduce a mobility-aware channel estimation scheme that accurately predicts channel statues in a lightweight way.
- We design a hybrid self-interference detection method that can handle RFID-specific high-loss problems.
- We build a working prototype with commodity C1G2 devices, and extensive experimental results show that MobiRate outperforms prior arts in various cases, including interferences, hardware diversity, and diverse mobility.

We prototype MobiRate using a Thingmagic M6e reader and evaluate it with 12 types of tags across different vendors. We compare MobiRate against two state-of-the-art solutions, Blink and CARA, in a wide range of settings. Through extensive experiments, we show that

- When tested with different scenarios (classroom, lounge, etc) and velocities under self-interferences, our countermeasures can first detect such interferences stably with an accuracy of more than 90%, and then shift probing direction for better rate selection, delivering 1.21x throughput gains on average.
- On average, MobiRate reduces probing cost significantly by 7.5x compared to Blink, and by 6.1x compared to CARA; MobiRate's rate selection module achieves throughput gains of 2.4x over Blink and 1.9x over CARA.
- MobiRate achieves overall throughput gains of 3.8x over Blink and 2.9x over CARA on average.

2.2 System Design

2.2.1 Overview

The previous observations have demonstrated that mapping-based methods, e.g., Blink and CARA, cannot deal with hardware diversity very well. Hence, we seek another way around. Unlike mapping-based methods, throughput-based methods are widely adopted in

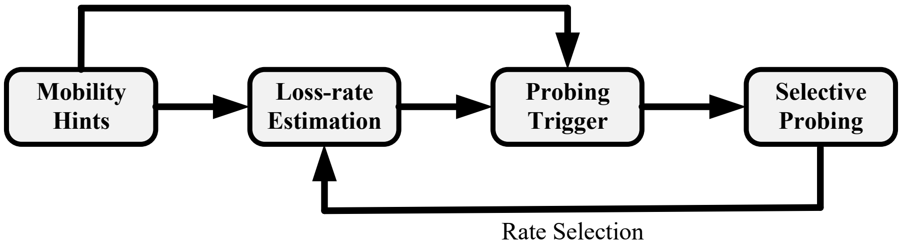


Figure 2.2: Framework of MobiRate. It is throughput-based and includes loss-rate estimation, probing trigger, and selective probing that helps choose the optimal rate.

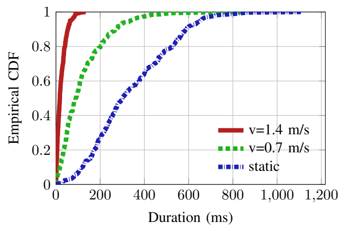


Figure 2.3: ECDF of durations that a current rate remains optimal for three cases, v=0, 0.7, 1.4 m/s.

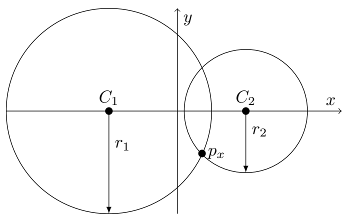


Figure 2.4: Single-antenna localization using two anchor points.

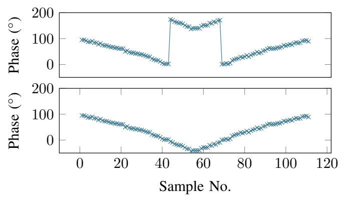


Figure 2.5: Phase unwrapping. The tag moves outward 8 cm (corresponding to a phase-rotation, π) and then backward 8 cm. The up sub-figure is for wrapped phases and the down is for unwrapped phases.

WiFi networks [18, 60], and some have been successfully deployed in commercial products, e.g., Minstrel [8] for Linux OS, ATH9k [3] for Atheros WiFi cards. The basic idea of throughput-based methods is simple. Different from mapping-based methods using signal strength-related indicators, throughput-based methods primarily rely on the *loss rate* deduced from the packet delivery history, which is more reliable and more direct for optimizing throughputs across various environments. A typical drawback of throughput-based methods [18, 60, 8, 3], however, is that while they work very well with static channels, they are unable to promptly respond to mobile channels because there is no fast motion indicator available in the PHY layer.

To achieve the best of two worlds, we design a novel link layer based on loss-rate estimation and show its framework in Figure 2.2. The key distinction of this estimation from those for WiFi networks is that it is not fixed but velocity-based. Such a design is preferred in MBNs because when the node moves faster, the channel should change faster, and the optimal rate should become out-of-date faster. In addition, it has a mobility-assisted probing trigger that can eliminate unnecessary probing based on velocity and position estimates. Third, it introduces selective probing that creatively makes use of the built-in C1G2 commands to avoid MAC collisions and enables per-tag rate adaptation, saving significant probing time. Finally, we compare the probing result against the current rate and choose the rate that has the smallest predicted average packet transmission time.

2.2.2 Velocity-based Loss-Rate Estimation

Exponential Moving Average. In this section, we show how mobility hints can help rate adaptation and how we can obtain those hints in a lightweight way. At the start, we investigate how the optimal rate changes with mobility [3]. As depicted in Figure 2.3, we see that there is a strong correlation between the optimal rate and the intensity of mobility. In particular, when the tag is static, it is easy to catch up with the optimal-rate change using a large time window; but when it is mobile, we will have to put more weight on the recent history. In light of this, MobiRate introduces a packet (throughput) based rate adaptation method with a mobility-based smoothing factor. It does not require any training like mapping-based methods [86, 33]. Specifically, the loss rate is estimated as follows,

$$p_c' = \eta p_r + (1 - \eta) p_c, \tag{2.1}$$

where p_c is the current loss rate at the current rate, p_r is the most recent packet delivery state, p_c' is the new current loss-rate estimate, η is the smoothing factor that is adjusted according to the velocity. p_c' is updated for every packet sent. To measure loss rates, we adopt the classic probing-based method [18, 60, 8, 3]. For example, if we transmit 10 probing packets and receive 8 acknowledgments, then we estimate the loss rate as 20%. Given the estimated loss rate, p_c', and ground truth rate, p_g', the loss rate error would be $p_e' = p_c' - p_g'$.
Smoothing Factor. The above equation has been adopted by several prior works of WiFi networks [8, 3], but they cannot adapt to different mobility. Because η is the key here. Intuitively, a larger η would put less weight on p_c, the current loss rate, which is the previous history. Thus, a proper high η can ideally smooth out the effects of the past when the mobility is high, and a proper low η should work well when the mobility is minimal or even 0. Currently, we set η empirically based on the trace analysis. In particular, we collect optimal-rate traces based on different velocities ranging from 0 m/s to 1.5 m/s with a step of 0.1 as [41]. Then, we generate 10 random trajectories for each velocity setting and simulate the mobility for different discrete η values. Some typical results are reported in Figure 2.6. For example, when the velocity is 0.3 m/s, the maximal throughput is achieved with $\eta = 0.24$. Similar trends can be observed for other velocity settings, which means optimal η values do exist. Therefore, we put those results into three equally spaced groups, $v = 0$ m/s, $0 < v < 0.8$ m/s, $v \geq 0.8$ m/s. Then we average their optimal η values within the group and the obtained group η values are 0.07, 0.28, 0.39, respectively. Note that our smoothing

[3]The optimal rate is obtained from trace-based analysis as in [41].

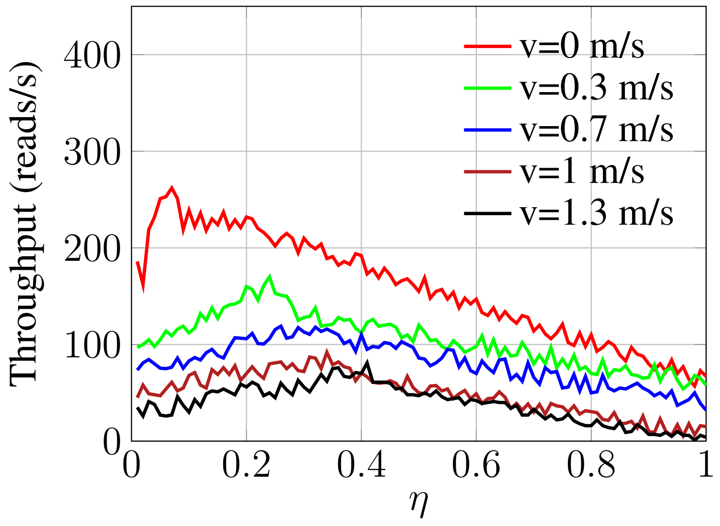


Figure 2.6: We investigate how different η values affect link layer throughput under various velocity settings. We observe that optimal η values can always be found for different mobility cases.

factor settings are not optimal in terms of throughput maximization. For example, finer-grained η searching can be realized by either introducing more velocity groups ($\gg 3$) or more subgroups for η discretization.

Deducing Mobility Hints. Different from prior work in WiFi that adjusts η indirectly, our design is able to choose the best η according to the directly measured mobility. To accurately set η, we need to estimate the velocity of the tag. Here, we introduce a novel *single-antenna* based tracking method that can predict position and velocity in real-time for rate adaptation. Although there are lots of related work of RFID real-time localization, they do not fit our purposes. They either introduce undesired tracking delay (at least seconds [79]) or require antenna arrays [77, 34, 75] to do localizations. Our solution is simple and elegant, which uses only a single antenna and phases, a PHY-hint, to measure distances. Phase measurements are supported in most commercial readers as specified in the LLRP standard [7]. For every successful identification process, the reader outputs a phase reading. This reported phase can be used to measure the distance between the reader and tag, denoted as R. From the electromagnetic theory, we can model R and the measured phase, θ, as follows [79],

$$\theta = 2\pi\frac{2R}{\lambda} + \theta_D + \theta_R + N\pi + n, \tag{2.2}$$

where λ is the wavelength, θ_D, θ_R, are phase errors brought by tag and antenna diversity, and reflection characteristics, respectively, N is the integer ambiguity as the measured phase is with period π, n is the noise. If we know two phases at two locations, then the distance between the two can be approximated as

$$\Delta R \approx \frac{\lambda}{4\pi}\Delta\theta, \tag{2.3}$$

where θ_D, θ_R, and N are perfectly canceled out, and only noise n is left.

The insight of our single-antenna tracking method is to trade the number of anchor points with the number of antennas. Because we need to adapt the rate for each antenna, we require the tag to pass two (known) anchor points. Afterward, when the tag moves to any new position, we can localize it using Equation 2.3 by leveraging the distances between the new position and the two anchor points. As shown in Figure 2.4, let C_1 and C_2 denote two anchor points, and p_x is the current position, which is unknown. Using Equation 2.3, it is easy to obtain r_1 and r_2 reliably. Then by intersecting two circles, we can deduce the position of p_x. When there are two intersection points, we can remove this ambiguity by simply adding another anchor point or using additional constraints.

The complete details of this estimation process can be found in [74]. Here, a brief estimation process is described as follows. As a general time-varying system, we can model the tracking transition between snapshots $k+1$ and k as a Gaussian process,

$$\mathbf{\Theta_{k+1}} = \mathbf{A}\mathbf{\Theta_k} + \mathbf{s_k} = \begin{pmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \mathbf{\Theta_k} + \mathbf{s_k} \tag{2.4}$$

where Δt is the time difference, $\mathbf{s_k}$ is the system-state Gaussian noise, $\mathbf{A}$ is the system matrix depicting mobility. Both $\mathbf{\Theta_k}$ and $\mathbf{s_k}$ follow Gaussian distribution, i.e., $\mathbf{\Theta_k} \sim \mathcal{N}(\mathbf{m_k}, \mathbf{\Sigma_k})$, $\mathbf{s_k} \sim \mathcal{N}(0, \mathbf{Q_k})$. Besides the above transition equation, we need to model the observation process. In particular, the observed distance, $\mathbf{z_k}$, can be modeled as two points in Euclidean space as follows,

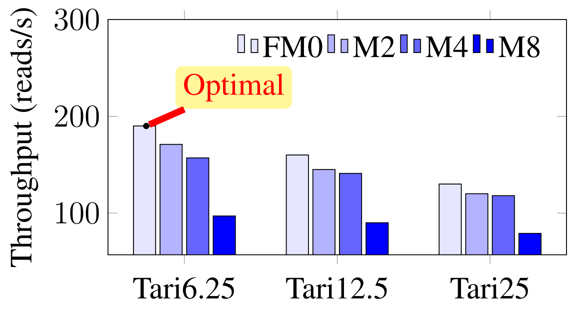
$$\mathbf{z_k} = (\sqrt{(p_{x,k} - C_{x,i})^2 + (p_{y,k} - C_{y,i})^2}) + \mathbf{u_k}. \tag{2.5}$$

where $C_{x,i}$ and $C_{y,i}$ are the positions of the antenna i that reports ϕ_k, and $\mathbf{u_k} \sim \mathcal{N}(\mathbf{0}, \mathbf{R_k})$ is Gaussian noise. When transition and observation equations are ready, we can use Kalman filters to iteratively estimate future states. Note that since the above observation equation is nonlinear, we need to use the Extended Kalman Filter. Hence, we change the observation equation into

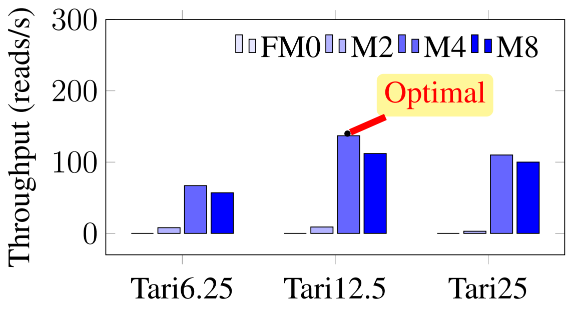
$$\mathbf{z_k} = \mathbf{D_k}\theta_\mathbf{k} + \mathbf{u_k},$$

where $\mathbf{D_k} = (\frac{\partial \mathbf{z_k}}{\partial P_{x,k}}, \frac{\partial \mathbf{z_k}}{\partial P_{y,k}}, \frac{\partial \mathbf{z_k}}{\partial V_{x,k}}, \frac{\partial \mathbf{z_k}}{\partial V_{y,k}})$, is the observation matrix. After this, we just follow standard Kalman filter processes to do prediction and calibration.

One thing to note is that we can see that N can be cancelled in Equation 2.3. Yet, the remaining phase may still contain a fractional part (between 0 and π) and an integer part.



(a) A good link



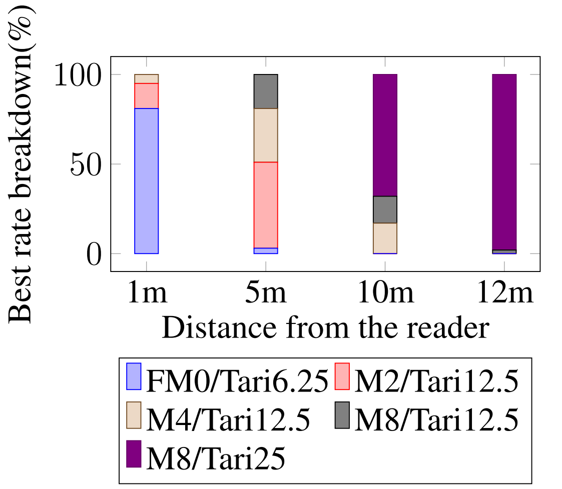
(b) A bad link

Figure 2.7: (a) is an example of a good channel, which favors the fastest uplink rate (FM0) and downlink rate (Tari=6.25); (b) is an example of a bad channel. Specifically, both FM0 and M2 encoding settings do not work, and the performance of Tari 6.25 is even worse than that of Tari 12.5, which suggests Tari 6.25 is an aggressive choice.

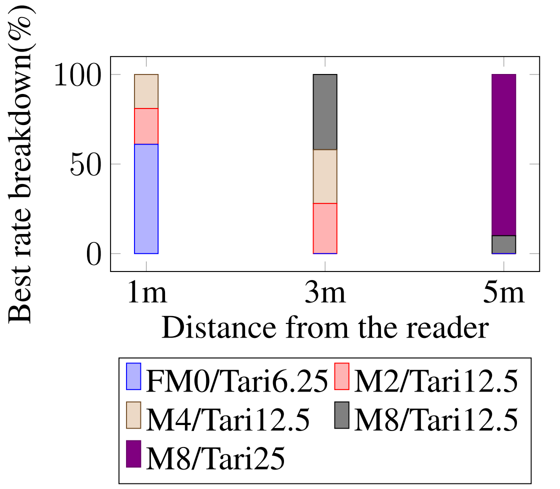
Therefore, we employ phase unwrapping [29] to track this integer, as shown in Figure 2.5. Otherwise, the distance from Equation 2.3 would be always less than $\lambda/4$.

2.2.3 Mobility-Assisted Probing Trigger

After knowing how to adaptively compute the current loss rate based on the velocity, the next question is when we may need to probe channels and in which directions. Generally, MobiRate starts with the highest rate. When there are four successive failures, it will probe the next lower rate, which is similar to SampleRate [18]. To avoid being trapped in low rates, if it stays at some rate over *a probing interval*, it tries to probe the next higher rate.
Probing Direction. Traditionally, Blink and CARA only consider the downlink rate, because they think uplinks do not make too much difference in rate adaptation. On the contrary, MobiRate takes both the uplink and downlink rates into account. To better show why the uplink rate is essential to backscatter networks, we present two examples in Figure 2.7. Recall that the downlink rate is decided by Tari values, and the uplink rate is controlled by BLFs and FM0/M2/M4/M8. In this test, we fix the BLF at 250 kHz. To examine how different Tari values affect the throughput, we put the tag at a known spot. As shown in Figure 2.7a, in the beginning, the link quality is good where faster rates deliver better throughput. The optimal rates in this case are Tari=6.25 and FM0. From this case, we know that in the case of good channels, we would miss the chance to increase the throughput if a too conservative Tari is adopted. Later, we move the tag 1-meter away, we observe dramatically different behaviors. As shown in Figure 2.7b, the backscatter link is experiencing difficulties because the throughputs of both FM0 and M2 are almost 0. The optimal rates become Tari=12.5 and M4 for this case, which tells that too aggressive rates may even hurt the overall throughput when the channel condition is not good. The root cause for this is that *if the downlink rate is not properly set, the uplink would be discontinued.* Thus, different



(a) optimal rate breakdown when the tag is within the antenna's beam



(b) optimal rate breakdown when the tag is at the beam edge

Figure 2.8: Optimal rate distribution at different distances. a) optimal rates are diverse within 10 m but become dominated by M8/Tari25 around 12 m; b) the similar phenomenon is observed around 5 m.

from Blink which only focuses on uplink rates, MobiRate always probes in both directions whenever it needs to test the next higher or lower rate.

Lower-Rate Probing Control. Besides a fixed interval that is used for probing higher rates, our MobiRate also controls the times of lower-rate probing to reduce probing costs. There are two special cases we need to take care of. First, when the tag is in dead zones, we need to probe at the lowest rate, M8/Tari25, and cut all the probing triggers. There are two kinds of dead zones we observed: a) within the antenna's beam, about 12 m from the reader; b) at the beam's edge, about 5 m from the reader. This rule comes from the observations in Figure 2.8. We find that while the optimal rate distributions are not the same across different distances, M8/Tari25 is dominating in the above two zones. The reason of such dead-zone formation is that when path loss becomes the dominating factor for channel quality, no one can survive but the lowest rate. Fortunately, our mobility hints can help us quickly judge whether tags are in those dead zones.

Second, when the tag is moving towards the reader, we increase the number of successive failures to 8, which is twice of the original setting, in order to suppress a lower-rate probing. The rationale of doing so is that while the tag moves closer to the reader, more transient failures can be caused by the fast fading or multipath effects. As observed from Figure 2.8, when the closer the tag is from the reader, the higher the percentage (probability) the faster rates become optimal. For example, when the tag is 1 m away, FM0/Tari6.25 takes 81% share in Figure 2.8a. Thus, to keep from being trapped into lower rates and save probing

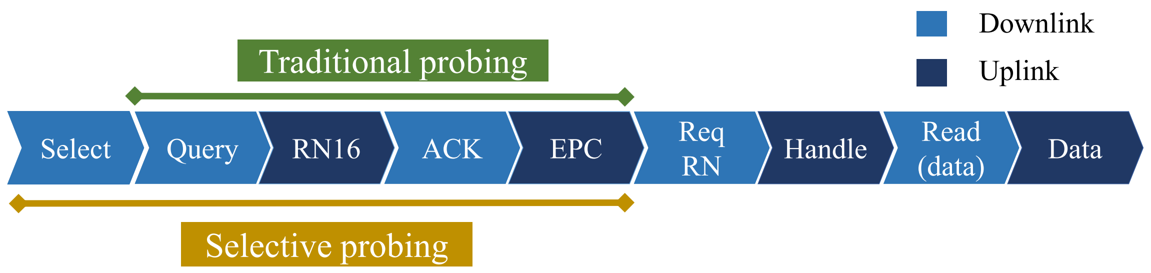


Figure 2.9: Comparison of traditional and selective probing by the message breakdown. The additional *SELECT* command is only 46-bit long excluding the *MASK* field.

Table 2.2: Measurements of RSSIs and loss rates along different lines.

LineA-Distance (m)	RSSI (dBm)	Loss rate (%)
1	-61	9.4
1.7	-51	98.6
3	-64	23.2
5	-66	71.8
LineB-Distance (m)	**RSSI (dBm)**	**Loss rate (%)**
1	-61.5	10.1
2	-62	14.6
3.2	-53	89.8
4	-64.5	53.8

time, we choose to allow more transient interferences by increasing the limit of tolerable successive failures.

2.2.4 Selective Probing

After solving the problem of when to probe, we come to the next one: how to probe and deduce accurate channel measurements. As shown in Figure 2.9, since we intend to keep full compatibility with C1G2, the minimum unit for probing is the identification process where the downlink should have *Query* and *ACK* commands, and the uplink requires *RN16* and *EPC* responses. After a correctly decoded *EPC* is received, we mark the probe as positive, otherwise negative. Blink [86] adopts similar strategies but has a main drawback: multiple tags could collide, which is incorrectly counted towards packet loss. To solve this, CARA [33] proposes to estimate the collision probability. Yet, it still suffers from the problem of unpredictable capture effects [20]. To overcome the aforementioned difficulties, we observe the opportunity brought by the built-in command, *SELECT*.

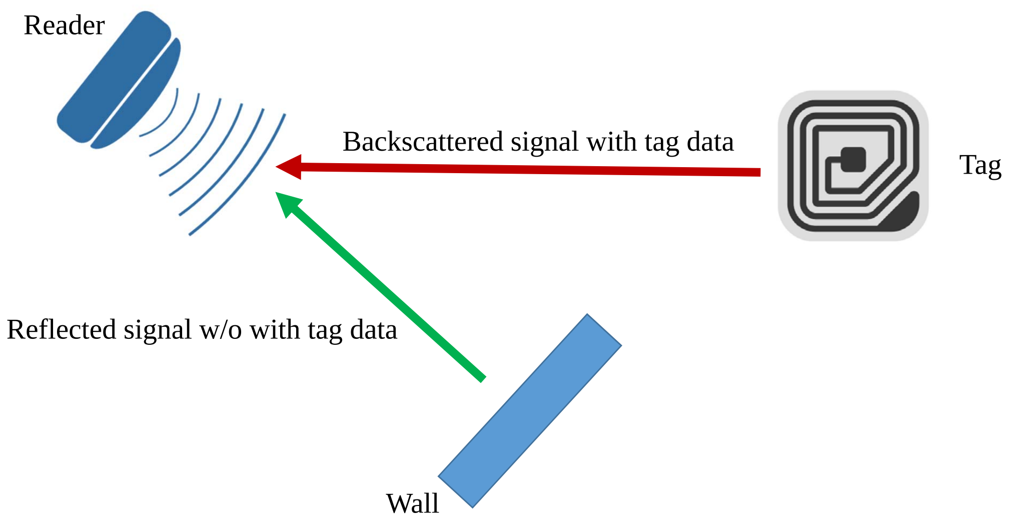

Reader
Backscattered signal with tag data
Tag
Reflected signal w/o with tag data
Wall

Figure 2.10: Illustration of self-interference. When the reader receives the reflected signals from the tag and the wall at the same time, the signal strength could be high due to constructive interference, but the loss rate is high as the backscattered signal is not strong.

2.2.5 Self-Interference

For backscatter communication, self-interference is another phenomenon that affects wireless channels [86]. Different from multipath that usually contains multiple copies of the same signal at different delays, self-interference happens when the reader receives both reflected signals from the wall (not backscattered by the tag) and backscattered signals from a tag, as shown in Figure 2.10. Those two kinds of signals may result in high RSSI if both two superimpose constructively. This is problematic because usually high RSSI indicates a good wireless channel. In this case, however, the channel for backscatter signals may be poor (highly likely) but the reader is 'fooled' by high RSSIs brought by self-interference. Further, such fake high RSSIs can mislead the rate adaptation probing mechanism.

Self-interference observation. To deal with it, we first need to examine how self-interference affects wireless communication. Through extensive experiments, we observe that there is a distinct feature with self-interference, which exhibits itself as high RSSIs and loss rates. To examine how it behaves, we move the tag along different lines and measure RSSIs and loss rates at different distances. The results of two typical lines are shown in Table 2.2. Along Line A, the tag at 1.7 meters away experiences self-interference. It has an RSSI of 51 dBm, much higher than the RSSIs at 1 meter and 3 meters. Meanwhile, it has a loss rate of 98.6%, which is even greater than loss rates at 3 and 5 meters. Similarly, we observe that when the tag moves along Line B, it experiences a loss rate of 89.8% and an RSSI of -53 dBm, which shows an unreasonably high loss rate than even further distances. Actually, the same phenomena are observed for other line movement results not shown in Table 2.2.

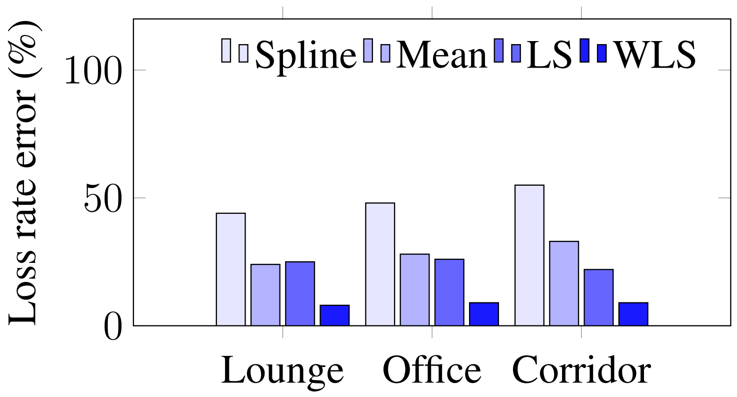


Figure 2.11: Loss rate estimation for self-interference cells by different interpolation approaches.

Self-interference detection. From the above observations, we design a detection strategy to capture self-interference. The key idea is to filter out the measurements that have higher RSSI and loss rates than usual. First, we do a site survey. In particular, we empirically build a grid of 10 m $\times$ 10 m, with each column and row of 50 cm width. Each unit records the average of RSSIs and loss rates. After we obtain this initial measured map, we need to remove self-interference cells that have high RSSIs and loss rates.

Second, we need to refill those blank cells after self-interference removal. Here, we employ interpolation for refills. In particular, we compare several classic approaches, including mean, least square (LS), spline, and weighted least square (WLS), and choose the best one. From the results in Figure 2.11, it is easy to tell that WLS is always the best among all four interpolation methods in different scenarios, so we adopt it for estimating self-interference cells due to its robustness. The primary reason for this is that WLS properly estimates the RSSI first and then uses the RSSIs as the weights for loss rate estimation. For the ground truth channel statues, we use a USRP to emulate the tag to do channel estimation by filling out the interference signals from the reader.

After the map is fully rebuilt without self-interference, we empirically set two thresholds, R_t and L_t. For each measured R_m and L_m, we compare them against the closest unit based on the estimated location. If the deviations are more than R_t and L_t at the same time, we estimate the state to be self-interference. Otherwise, it is self-interference free.

Probing after detection. Based on the states detected, we need to take different probing measures. If there is no self-interference, we just follow the previously bi-directional probing based on the current rate. However, if self-interference is detected, bi-directional probing is wasting time. Because in this case, it is the SINR (Signal to Interference and Noise Ratio) that determines the rates. The real backscattered signal is weak, and thus there is no chance that a higher rate can provide better throughput. As such, the probing strategy is to check with lower rates to jump out of the high 'loss rate' trap. The pseudo-code for self-interference of rate adaption is outlined in Algorithm 1.

Algorithm 1 Self-interference detection for rate adaption

```
//Site survey, N=400 in a 10x10m area
for $i = 0$ to $N - 1$ do
    $R(i)$=average RSSIs in this cell
    $L(i)$=average loss rates in this cell
end for
Remove self-interference cells.
Refill those cells using WLS interpolation.
//self-interference detection
Find the localization cell $x$ //Given $R_m$ and $L_m$
Obtain the corresponding $R(x), L(x)$
if $R(x) - R_m > R_t$ and $L(x) - L_m > L_t$ then
    self-interference-state = true
else
    self-interference-state = false
end if
//Probing after detection
if self-interference-state then
    probe the lower rates
else
    bi-directional probing for the best rate
end if
```

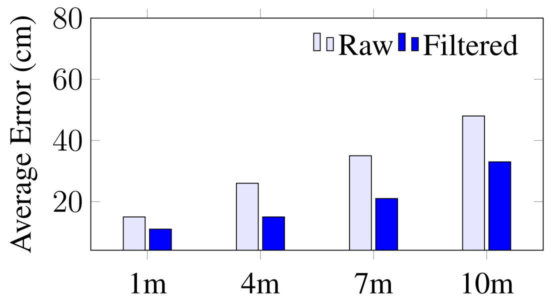
2.3 Implementation

In our implementation, we employ a Thingmagic M6e reader, which is fully compatible with the C1G2 standard. The tags we test include 12 types, such as the Alien Higgs 3 tag (T4), the metal resistant tag (T7), the SMARTRAC ShortDipole tag (T12), which are the same as [30]. As the sensor data is not important for transmissions, we simply write random bits into those tags. The rate adaptation programs are written in C# using Mercury API SDK v1.29.3.

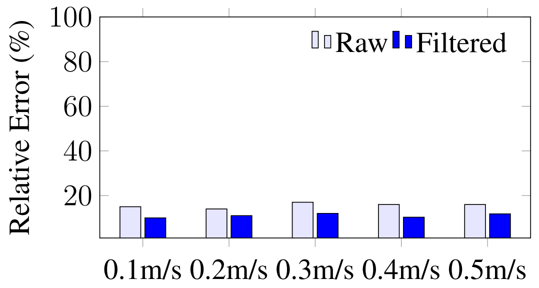
Other experimental parameters include: BLK 250 kHz, Tari 6.25/12.5//25, uplink encoding methods FM0/M2/M4/M8, ID 96-bit long, reader power 30 dBm. We also compare MobiRate against Blink [86] and CARA [33], two state-of-the-art MBN rate adaptation methods.

2.4 Evaluation

Mobility Hints. At the start, we examine how the single-antenna tracking works. For the first group of experiments, we put the tag at different locations but within the reader's

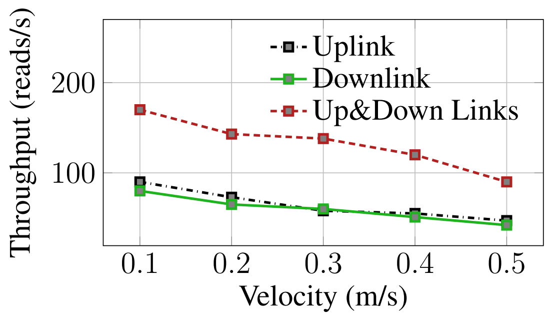


(a) Localization errors at different distances from the reader

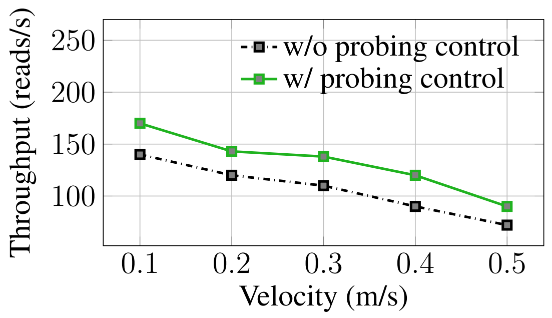


(b) Velocity relative errors at different mobility

Figure 2.12: Comparison of raw estimates and Kalman filtered results. (a) The filtered results are always better than raw estimates and the localization errors grow with the distance due to attenuated RSSIs; b) The filtered velocity estimates are also better than raw results, while the velocity error is well under control across different mobility.



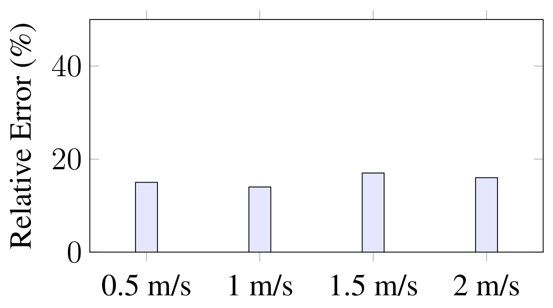
(a) Comparison of adapting uplink rates, downlink rates, and the both.



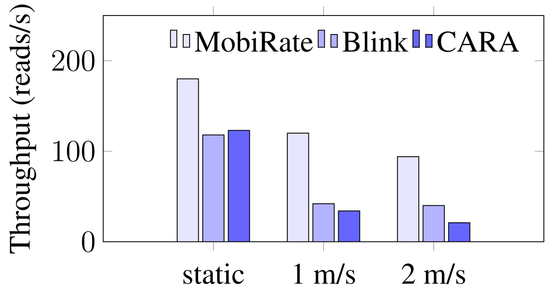
(b) Comparison of cases that are with and without our lower-rate probing control mechanism.

Figure 2.13: (a) shows that adapting both rates is necessary as adapting a single link can only achieve around half of the optimal throughput; (b) shows that the lower-rate probing control mechanism helps optimize throughput about 25% across different velocities.

beam. They are 1 m, 4 m, 7 m, and 10 m away. The results are depicted in Figure 2.12a. We can see that the average localization errors of the raw and Kalman filtered estimates are 31 cm and 20 cm, which are adequate for rate adaptation. This is achieved using only a single antenna, and all the processing are done is real-time as well. In addition, we want to investigate the quality of velocity estimates. To do so, we use a programmable robot (*iRobot Create 2*) carrying a tag, whose velocity could be controlled by reprogramming. As demonstrated in Figure 2.12b, the velocity relative error is within a reasonable range for different velocities. Specifically, the errors of all the filtered velocity estimates are less than 12%. In a word, our mobility deduction is rewarded since it can deliver high-quality location and velocity estimates in real-time.



(a) Velocity errors



(b) System performance

Figure 2.14: Estimation errors with high mobility and its impact on system performance.

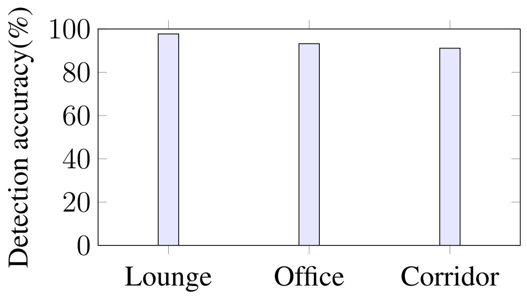
Probing Trigger. Next, we continue to examine how well our probing module performs. Here, we focus on two things: 1) adapting both the uplink and downlink; 2) the lower-rate probing control mechanism. First, we conduct a comparison for three scenarios: adapting only the uplink, only the downlink, and both. As shown in Figure 2.13a, we see the throughput of adapting both links is always better than the other two. In particular, when the tag moves at 0.3 m/s, adapting only the uplink brings 58 reads/s while adapting only the downlink has a throughput of 60 reads/s. On the contrary, when both links are considered, the throughput improves to 138 reads/s. Similar trends are observed for different mobilities. This confirms that both rates should be well adapted to maximize network throughput.

Furthermore, we perform a microbenchmark test on our lower-rate probing control mechanism. As shown in Figure 2.13a, the system with a probing-control mechanism has better performance. For example, when the robot travels at 0.4 m/s, the system without a probing-control mechanism can only deliver 90 reads/s throughput while it improves to 120 reads/s after probing-control is introduced, corresponding to a gain of 1.34x. In short, we prove the lower-rate probing control mechanism is effective and useful, which delivers a 1.25x throughput gain on average.

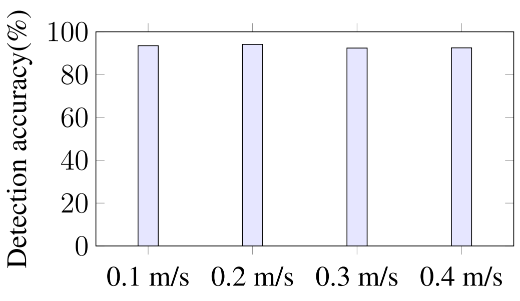
Performance with High Mobility. Previously, we have shown velocity estimation errors with low mobility under 0.5 m/s, which is mainly due to the moving speed of the robot. In the case of channel hopping, our method supports up to 2.67 m/s mobility [4]. Within a channel, the supporting velocity increases to 10 m/s [5]. Although high mobility robots are not affordable (e.g., more than $100,000) at the moment for us, we ask our volunteers to carry tags moving at 0.5 m/s, 1 m/s, 1.5 m/s, and 2 m/s, respectively. Also, we use

[4]Since the gap of phase measurements from two channels is 30 ms and as long as the phase rotation is less than π, which corresponds to 8 cm displacement, there is no ambiguity.

[5]The time of two consecutive phase measurements in a channel is around 8 ms.

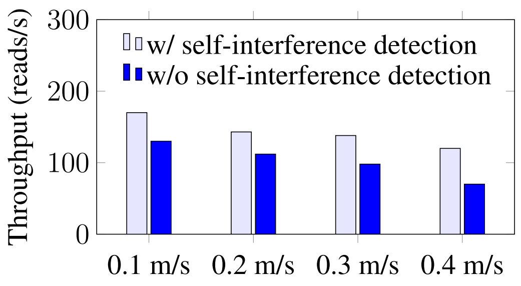


(a) Detection accuracy for different scenarios

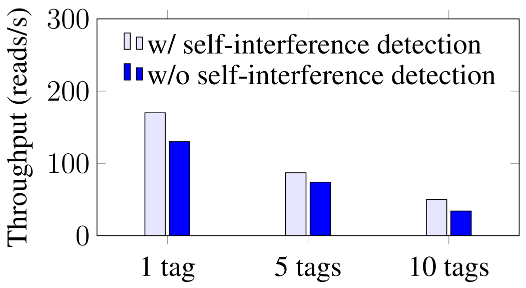


(b) Detection accuracy for velocities

Figure 2.15: Detection accuracy for different scenarios in (a) and tag velocities in (b). Both show that our self-interference detection scheme is robust and achieves more than 90% accuracy for all cases.



(a) Impact of self-interference detection on mobile tags.



(b) Impact of self-interference detection on tag populations.

Figure 2.16: Throughput comparison for mobile tags in (a) and tag populations in (b). The results show that the throughput with the self-interference detection is on average more than 20% better than that without such detection.

an OptiTrack system [15] to ensure the human moving speed is as close to our predefined setting as possible.

The results of high mobility are shown in Figure 2.14. From Figure 2.14a, we can see that our velocity estimation errors are within 20%, demonstrating that even though our estimation scheme is lightweight but still quite robust to higher mobility. In particular, when the tag moves at 0.5 m/s and 1.5 m/s, the velocity estimation errors are 13% and 14.3%, respectively. Then we investigate how high mobility affects overall reading performance. Besides different mobility statuses shown in Figure 2.14b, for each status, we try three different tag populations, which are 1, 5, 10 tags, and report the average. We observe that all three systems are negatively impacted by mobility, but MobiRate stays strong whereas the other two surrender. Particularly, when the tags moves at 2 m/s, MobiRate achieves 2.4x and 4.5x throughput gains over Blink and CARA. CARA is the worst because it lacks mobility hints.

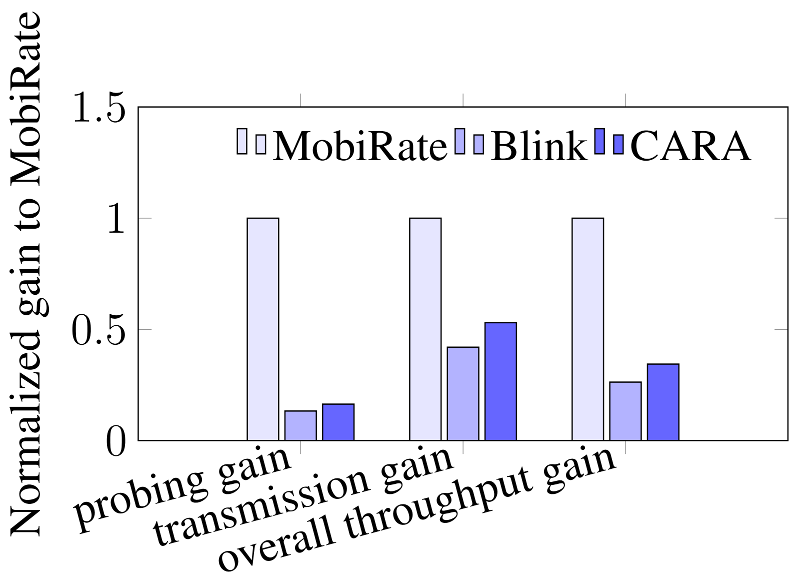


Figure 2.17: Overall throughput comparison.

Self-Interference Detection. Next, we examine how our self-interference detection works. First, we perform tests in different scenarios, including lounge, office and corridor. Results in Figure 2.15a demonstrate that our detection is very robust because the accuracies are all above 90% in all three cases. As offices and corridors tend to have more reflectors than lounges, which brings more self-interferences, the detection achieves the best accuracy in lounges. Also, we need to investigate how the detection scheme works with tag mobility. Figure 2.15b shows that decent accuracies are achieved for a range of different velocities, which means our detection is resilient to tag mobility. It is mainly due to our accurate single-antenna localization scheme. It not only outputs precise locations but also reliable velocity estimates. Thus, the detection can make use of such estimates to find the right closest unit of the empirical grid.

Besides, we conduct microbenchmark tests to check the effectiveness of probing based on the detection results. For a single mobile tag, we let it move at different velocities and compare the throughputs of our probing scheme against the partial implementation without self-interference detection. Figure 2.16a plots the results. We observe that for tags of different velocities, the probing with self-interference is consistently better than that without detection, which is a 1.28x throughput gain on average. This is because of shorter probing time as we only need to probe lower rates once self-interference is found. Also, we examine how self-interference detection impacts different tag populations. Results in Figure 2.16b show that successful self-interference detection also benefits the whole tag population, which is on average a 1.21x throughput gain. It can be primarily attributed to high-accuracy detection and selective probing, which avoids unnecessary time waste on empty slots.

Overall Performance. In the end, we study the overall performance. In particular, we conduct more than 120 tests for a wide range of parameters. The velocity of tags varies from 0 to 1 m/s and tag populations are from 1 to 15. Those experiments are performed for 10 days at two different places, an indoor office and a lobby. We report the overall gains and their breakdown in Figure 2.17. From those, we observe that MobiRate is the best and achieves overall throughput gains of 3.8x over Blink and 2.9x over CARA on average. To break down these gains, MobiRate reduces probing cost significantly by 7.5x compared to Blink, and by 6.1x compared to CARA. This is attributed to the design of mobility-assisted probing trigger and selective probing. In the meantime, MobiRate is 2.4x and 1.9x better than Blink and CARA in terms of data transmission. Those gains are primarily due to throughput-based rate selection that can adapt rates according to both the uplink and downlink.

2.5 Discussion

Probing Process: MobiRate uses full identification processes for probing, which incurs some unnecessary delays, like collecting EPC. However, such limitation is mainly due to COTS device compatibility. Therefore, for a lighter probing scheme, we may introduce an SDR reader that can probe tags using much shorter packets and ACKs, like *Query* or other customized commands. Also, we may borrow the WiFi probing packet design from [8], where the length of the payload is made as short as possible.

Opportunity-based Fairness: Currently, MobiRate is fully compatible with C1G2, which means it follows C1G2 MAC protocols. In C1G2 protocols, not every tag has equal access time because it only provides opportunity-based fairness. Actually, opportunity-based fairness methods are widely adopted in mainstream wireless networks, e.g., WiFi [2] and Bluetooth. The key to opportunity-based fairness methods is to provide equal chances to all the nodes while access time and communication throughput are not guaranteed. In most real-world applications, like asset tracking [79], opportunity-based fairness methods work well since they can easily adapt to complex wireless environments.

Starving Tags: In our current design, there may be starving tags, i.e., some tags may never be interrogated. To deal with such a problem, we may design a throughput-based access protocol, where it is guaranteed that each participant can achieve equal throughput. For such a design, usually the rate is fixed (e.g., FM0) in the link layer because it would be quite challenging to ensure the same throughput for multiple rates. One solution for multi-rate cases is that we may try to provide equal access time to all the nodes, instead of throughput. In the future, we intend to improve the current link layer design to ensure no starving tags.

Smoothing Factor: In theory, the main factors that affect the smoothing parameter are mobility and environment. In this paper, we mainly employ velocity to denote mobility, and future work may further investigate acceleration and orientation. Meanwhile, modeling the environments includes RF propagation properties, frequency fading, polarization, etc. In addition, we guess deep reinforced learning may be an excellent way to set the smoothing factor better. For example, we can infer η by collecting more data traces and employing the interaction between η and real-world rate feedback. This way, we can dynamically change η according to different environments and mobility statutes. However, this method may incur unwanted processing delays; thus, how to make it lightweight is worth further examination.

2.6 Conclusion and Future Work

We have presented MobiRate, a mobility-aware rate adaptation for MBNs. Specifically, we have designed a novel single-antenna tracking method, which can accurately deduce mobility hints from the PHY layer. Future work includes distributed rate adaptation for unsync readers, fair MAC accesses, and extensions to other general-purpose backscatter systems, including WiFi, Bluetooth, and LTE.

Chapter 3

A Reliable Modulation Design For BLE Backscatter

3.1 Background and Motivation

3.1.1 BLE Primer

BLE is a massive overhaul of the previous Bluetooth specifications by providing significant power saving for peripheral devices that do not require high data rates but the lengthened battery life, such as healthcare and fitness applications. It achieves huge market success and thus most operating systems including iOS, Android, as well as macOS, Linux, Windows 8 and 10, natively support BLE [10, 11, 4, 9]. BLE operates at 2.4 GHz, the unlicensed ISM band, and has a 1 Mbps raw data rate. It has 40 pre-defined channels ranging from 2400 to 2483.5 MHz, each of which is 2 MHz wide. To build connections and transmit data, the specification defines two kinds of packets: advertising and data packets. BLE devices send advertising packets to broadcast data and allow other devices to find and connect to them. The BLE device advertises on three channels, channel 37 (2402 MHz), channel 38 (2426 MHz), and channel 39 (2480 MHz).

3.1.2 Problems of Prior System

The backscatter solutions offered by FS-backscatter [88], Interscatter [46], and BLE-backscatter [25] cannot be entirely built using off-the-shelf Bluetooth radios. However, the latest work in this area, FreeRider [87], has achieved this goal for the first time, using only commodity BLE radios. It, however, suffers from several key reliability issues.

1. *Unreliable two-step modulation.* To be compatible with BLE signals, FreeRider employs two-step modulation. It first shifts the exciting signal to the target channel, e.g., 6 MHz frequency shift, and then uses an additional frequency shift, e.g., 500 kHz, to do codeword translation. Detailed later in Section III, doing so would inevitably introduce unreliability due to self-interference or no solid signal in the target frequency.

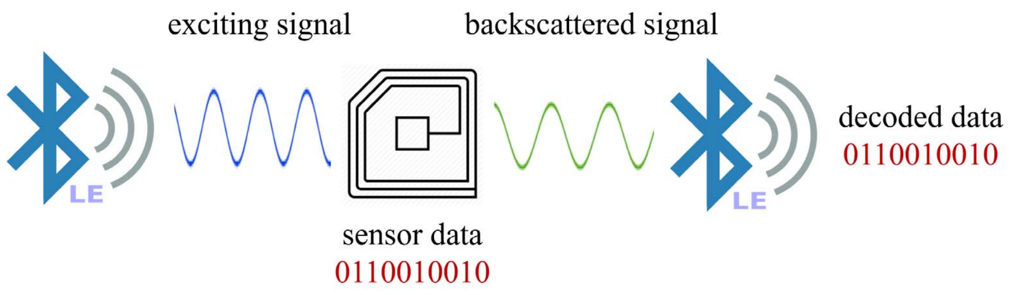


Figure 3.1: RBLE conceptual design. The RBLE tag modulates its sensor data on BLE exciting signals and backscatters new BLE packets that any commodity BLE device can decode.

Hence, a new modulation scheme that does not rely on codeword translation is needed to fix this problem.

2. *Productive-data dependency.* Besides the unreliable modulation, FreeRider has another serious issue that it requires the data sequence of exciting signals to decode the tag data, which means if the data sequence of the original channel is corrupted, there is no way to successfully decode the tag data even when the backscatter data sequence is error-free. Though working with productive data is a nice property, such productive-data dependency would significantly impact the BER of the tag data, especially when the quality of the original channel becomes unstable due to mobility or occlusion.

3. *Lack of interference countermeasures.* None of the previous Bluetooth-based backscatter systems has provided proper countermeasures to interference. In fact, those systems do not perform well in the presence of overlapping channel interference caused by other wireless technologies, including WiFi, ZigBee, cordless phone, microwave oven, which simultaneously work on the crowded 2.4 GHz ISM band.

3.1.3 Contribution

To make BLE backscatter reliable, RBLE makes the following technical contributions:

- RBLE uses BLE signals with partial single tones as excitations and finds the optimal modulation index, enabling robust BLE packet regeneration bit-by-bit.
- RBLE designs dynamic channel configuration that allows RBLE tags to perform channel hopping to bypass seriously interfered channels while none of the previous BLE-based backscatter systems has provided proper countermeasures to overlapping channel interference.

- RBLE presents BLE packet regeneration that uses adaptive encoding to further enhance backscatter reliability for challenging situations, e.g., low SNRs. The introduced adaptive encoding for BLE backscatter significantly reduces BERs.

We prototype RBLE using TI CC2540 radios and customized tags implemented by FPGAs. Through extensive empirical evaluation, our main results are summarized as follows.

- Compared to FreeRider, RBLE achieves more than 17.3x and up to 78.8x goodput gains in LoS cases, and achieves more than 17.1x and up to 66.x goodput gains in NLoS scenarios.
- The maximum goodput RBLE can achieve is 16.6 kbps, which is 95% of the theoretical capacity for a single excitation commodity BLE radio. Also, the maximum uplink ranges of RBLE are 25 m for indoors and 56 m for outdoors.
- By the help of dynamic channel configuration, RBLE with channel hopping achieves 1.92x goodput gain over the one without such help in the presence of strong WiFi interference.
- Our adaptive encoding can significantly reduce BERs. When the uplink distance is 5 m, the BER can be reduced from 0.56% using M1 encoding to 0.1% using M8, and when the uplink distance increases to 20 m, the BER drops from 9.2% using M1 to 0.45% using M8.
- We also implement RBLE with only off-the-shelf phones, including iPhones and Android phones. The experiments show that RBLE is able to work with both data and advertise packets from smartphones as carriers, and the maximal uplink range is 16 m for an iPhone as the receiver.

3.2 System Design

3.2.1 Overview

Figure 3.2 shows the framework of RBLE. A BLE device generates exciting BLE signals to control RBLE tags. Upon BLE signals are detected, the RBLE tag parses commands including hopping sequence, channel dwelling time, encoding coefficient, etc. Those core command parameters drive a state machine to dynamically configure channels. Those channel parameters together with tag data are used for BLE packet regeneration to produce raw binary bits, and to decide how to modulate the exciting BLE signals.

3.2.2 Modulation Using Direct Frequency Shift

The core of RBLE is how to modulate an exciting BLE signal into another BLE signal. Although we share the same motivation as the seminal work, FreeRider, our solutions are

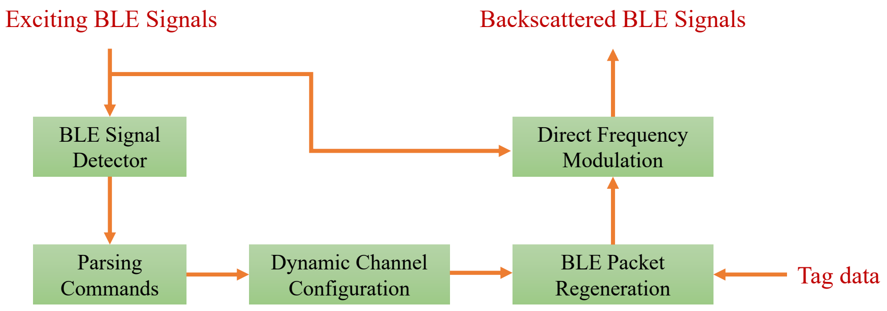
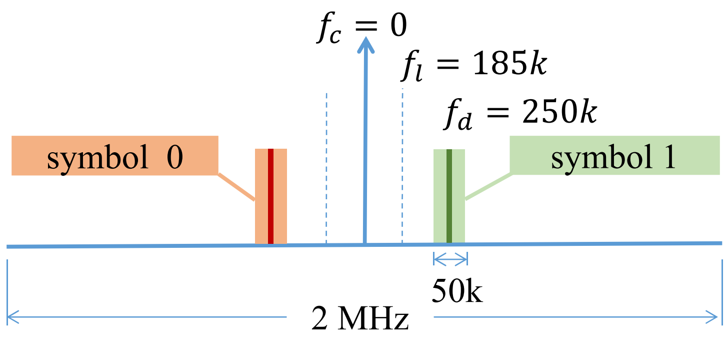


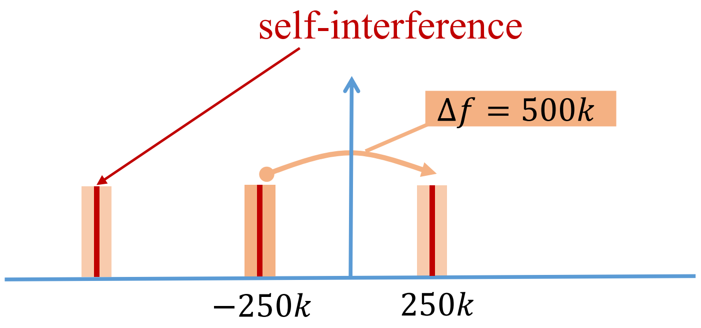
Figure 3.2: RBLE framework. After the RBLE tag detects exciting signals, commands are parsed using packet length demodulation. Then it drives dynamic channel configuration module together with tag data to regenerate corresponding BLE signals using direct frequency modulation.

quite different. Before the discussion of how well FreeRider performs, let us briefly review how original BLE signals modulate. As shown in Figure 3.3(a), in a channel of 2 MHz, the symbol 1 is represented by a position frequency deviation f_d, 250 kHz, and the symbol 0 is represented by a negative frequency deviation of the same amount, following Gaussian frequency shift keying (GFSK). According to BLE specifications, there are two mandatory tests to make sure that first, such deviations should be within 225 kHz and 275 kHz, which is 50 kHz wide, and second, at least 99.9% of all deviations must be greater than f_l, 185 kHz.

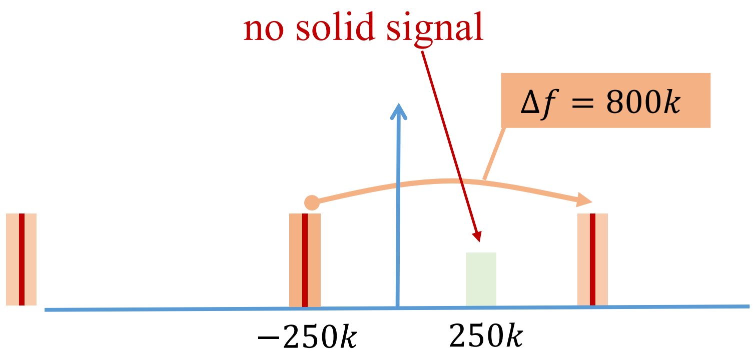
In order to make backscattered signals still BLE legitimate, FreeRider novelly proposes codeword translation, which translates one codeword to another. Note that this codeword translation happens after a frequency shift that moves the exciting signal from the original channel to the target channel. So we call it two-step modulation. While this proposal works really well with phase modulation for WiFi signals, some unexpected issues arise with GFSK modulation. As in GFSK, there are only two codewords, 0 and 1, which means codeword translation has to encode information by changing 0 to 1 or 1 to 0. In particular, such requirement of GFSK-based codeword translation leaves FreeRider only two choices, which is a dilemma. The first choice is shown in Figure 3.3(b), where the original symbol is 0 and FreeRider wants to make it 1. So it directly performs a frequency shift, $\Delta f = 500$ kHz, then the shifted symbol becomes 1. However, this operation inevitably produces another unwanted copy at $-250 - 500 = 750$ kHz, which is still within the BLE channel, creating self-interference. As said in FreeRider [87], single side-band cancellation solutions do not apply here because codeword translation is not aware of the original symbol and thus does not have any idea which side should be cancelled. Hence, the other choice left for FreeRider is to move this unwanted copy out of the channel, then built-in filters on the BLE receiver would ignore it, as depicted in Figure 3.3(c). So for the same purpose translating 0 to 1, Δf has to be more than 750 kHz. In this case, it indeed moves the unwanted copy signal



(a) BLE GFSK modulation



(b) FreeRider: Case 1



(c) FreeRider: Case 2

Figure 3.3: The BLE specifications require a symbol 0 to be encoded as a negative frequency deviation around 250 kHz and a symbol 1 as a positive 250 kHz deviation, shown in (a). When FreeRider does codeword translation, it makes two cases. Case one chooses frequency shift as 500 kHz to translate 0 to 1 as in (b) brings self-interference caused by the unwanted copy at -750 kHz. Case two is to move this unwanted copy outside the channel by performing a frequency shift of more than 750 kHz as in (c), leaving no solid signal at 250 kHz. Both cases lead to unreliable modulation for FreeRider.

out of the channel but leaves no solid signal at 250 kHz deviation, where the BLE receiver looks for. In other words, case 2 makes translated codeword susceptible to noise. From the above, we can see that both choices result in unreliable modulation for FreeRider.

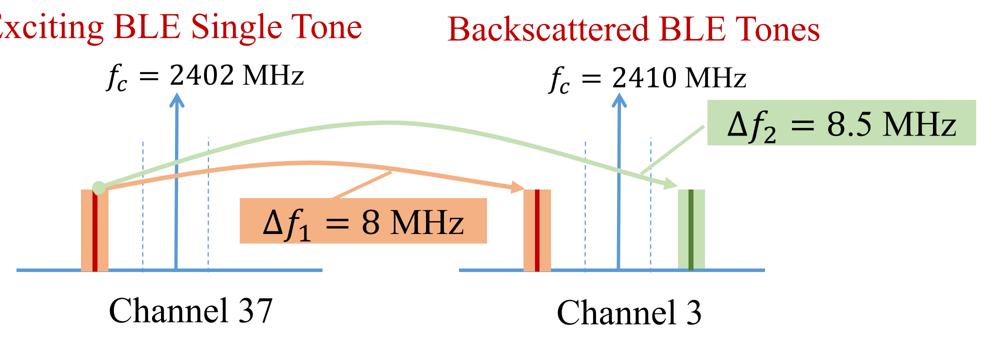


Figure 3.4: Direct frequency shift modulation example. To modulate a symbol 0 to target channel 3, we only need to shift a frequency of 8 MHz on the zeros single tone of the exciting signal on channel 37. A frequency shift of 8.5 MHz is for the symbol 1.

We empirically test FreeRider for various Δf and find that for single-bit modulation, BERs are all above 30% when Δf is between 500 kHz and 1 MHz, which presents huge challenges for practical applications. By investigating this problem in depth, our direct frequency-shift works as follows. First, we make the single tone part of a BLE signal as the modulation carrier. Inspired by Interscatter [46], by properly performing reversely-whitening techniques, the payload of the BLE signal can be made all ones or zeros, which are single tones. Next, we directly apply frequency shifts to modulate 0 or 1 in the target channel.

As shown in Figure 3.4, suppose we have all zeros single tone in advertising channel 37, and our target channel is data channel 3. If we need to modulate a symbol 0, a frequency shift of 8 MHz, Δf_1, would be chosen to modulate. Similarly, a frequency shift of 8.5 MHz, Δf_2, would be able to modulate a symbol 1. While such a design is simple, it is efficient and easy to implement. Also, it removes the productive-data dependency brought by FreeRider and requires only a single BLE receiver to decode. Moreover, while the Cyclic Redundancy Check (CRC) result of FreeRider packets is always erroneous, ours can be made right, which is important when the receiver is a smart device supporting BLE because many application software cannot display CRC-error packets [14, 13, 12]. In addition, for the unwanted copy of our frequency shift, though it would never fall into our target channel, we can still either apply single side-band cancellation techniques [46, 85] or just rely on the filters on the BLE receiver to take care of it, reducing unnecessary interference for other channels.

3.2.3 Dynamic Channel Configuration

Most existing backscatter systems [85, 46, 87] are all static, which means targeting a single channel. Meanwhile, those systems are working on the same 2.4 GHz ISM band, which is quite crowded and thus full of interference. So, we intend to design a dynamic channel configuration scheme to enable channel hopping for RBLE and to reduce interference impact. To do so, first, we need a configurable clock generator that produces two different clocks

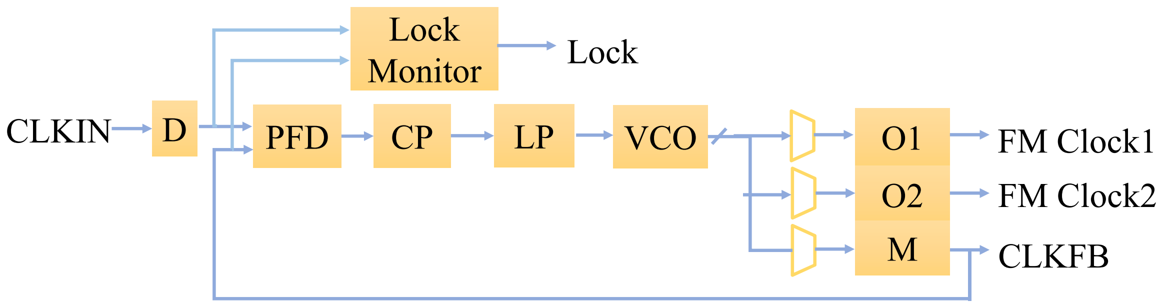


Figure 3.5: Configurable clock generator providing two clocks for modulation.

for our modulation. Hence, we design a Phase Locked Loop (PLL) based clock generator as shown in Figure 3.5. A phase-locked loop is a negative feedback control circuit that utilizes the voltage generated by phase synchronization to tune the voltage-controlled oscillator to produce a target frequency. The clock generator circuit mainly consists of a voltage-controlled oscillator (VCO), a phase frequency detector (PFD), a charge pump (CP), a loop filter (LP) and a lock monitor. The heart of the clock generator is the VCO. The VCO outputs a signal, part of which is divided by the input frequency to generate clock1 and clock2. The other part (CLKFB) is phase-compared with the reference signal in PFD to realize feedback; therefore the VCO can output a stable clock signal. O1, O2, M, and D are programmable frequency dividers with configuration registers that adapt the VCO to our clock generator. By configuring these registers, clock generator can generate the required frequencies. The VCO frequency can be determined by:

$$F_{VCO} = F_{CLKIN}(\frac{M}{D}) \tag{3.1}$$

$$F_{VCOMIN} \leq F_{VCO} \leq F_{VCOMAX} \tag{3.2}$$

Frequency of two frequency modulation (FM) clocks can be calculated using this formula:

$$F_{FM} = F_{CLKIN}(\frac{M}{D \cdot O}) \tag{3.3}$$

where F_{VCOMIN} and F_{VCOMAX} represent the controllable frequency range of the VCO, and F_{CLKIN} is the frequency of CLKIN. The M, D, O1, and O2 values come from configuration registers of programmable frequency dividers. The values of M, D must be chosen appropriately to keep the VCO within its frequency range. Based on two generated clocks, we can feed them into the modulation module that uses different clocks to control the RF switch accordingly.

To enable channel hopping, we need multiple sets of different clocks. One way is to generate multiple sets of clocks at the same time and select one set for output at a time. Nevertheless, it inevitably boosts power consumption if too many clocks are involved. For

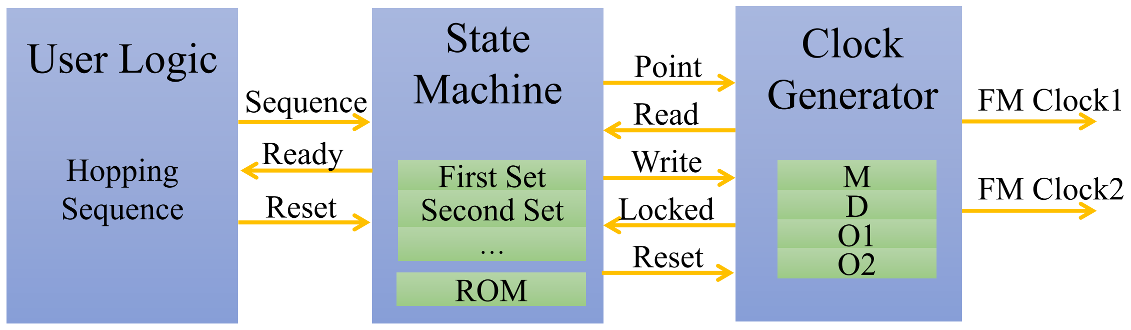


Figure 3.6: Channel hopping happens from user logic that parses hopping commands from exciting signals and then tells a state machine where to hop. Upon receiving this command, the state machine controls the reconfigurable clock generator through a set of ports, including Point, Read, Write, Locked, and Reset. After the generator state is locked, it produces two different clocks for the target channel.

example, there are 40 channels for BLE, which requires 80 clocks for hopping across them. To get around this, we design a dynamic reconfiguration technique to produce the required two clocks for each hopping. Thanks to our previous configurable clock generator design, dynamic reconfiguration only needs to vary the voltage level of VCO and reconfigure the registers of programmable frequency dividers. Specifically, dynamic reconfiguration is to dynamically change values of M, D, O1, and O2 in our configurable clock generator.

Our dynamic reconfiguration is performed through a reconfiguration port which provides access to the configuration bits stored in configuration registers. We design a state machine to drive the reconfiguration port. The state machine is used to generate control and data signals for the reconfiguration port using pre-computed values stored in ROM. This state machine ensures the configuration registers in clock generator are controlled and reconfigured in the correct sequence. Specifically, after the state machine receives the hopping signal from user logic, it first points to the configuration register of the clock generator through the reconfiguration port and then reads the previously configured parameters of the output frequency. Next, it masks the range of bits that need to be changed in configuration registers and chooses the appropriate ROM address according to the reconfiguration state. After that, it writes the updated values of M, D, O1, and O2 to configuration registers. Finally, it waits for the locked signal from the clock generator which indicates the completion of this reconfiguration event. When the frequencies of two clocks are changed, the state machine becomes ready again for the next reconfiguration. The configuration parameters including the addresses, masks, and new configuration values, are stored in a pre-initialized ROM. Figure 3.6 shows the block diagram of our channel hopping process. The state machine provides multiple reconfiguration states for user logic. The first set is the default state. Other sets correspond to user-configuration loaded into the configuration registers. Each state has a set of pre-computed values of M, D, O1, and O2. The user logic is generated by parsing commands from exciting signals, e.g., hopping sequence.

3.2.4 Packet Regeneration

After we have direct frequency shift modulation and channel hopping support, it comes to BLE packet regeneration. Suppose that we already have an exciting advertising packet where the Adv payload is single tone using reversely-whitening techniques [46]. To backscatter a legitimate packet with tag data, we have two options. First, we backscatter it to another advertising packet in a different advertising channel. In this case, we only need to modulate the tag data onto the Adv Payload field and keep all the fields before it unchanged. The second case is we can backscatter the exciting packet into a data packet. To do so, we need to fabricate the Preamble, Access Address, and Header for the regenerated data packet, and modulate tag data in the Data Payload field. By enabling regenerating both advertising and data packets, RBLE can hop freely and communicate across all 3 advertising channels and 37 data channels. To enable modulating on the single tone part of the signal, we need to find the starting position of the Adv Payload. Particularly, we employ an RF signal power detector and a voltage comparator. There are lots of off-the-shelf solutions. For example, AD8313 can convert an RF signal to an equivalent DC voltage with high accuracy, and has 40ns signal response time, which fits BLE applications. After the detector discovers a BLE signal, we can have a positive edge generated by the following voltage comparator. Then it skips 104 μs, which is the length of the preamble, access address, header, and adv address. Our modulation starts right after this at the rate of 1 μs per bit. Although our modulation is quite robust as a single-bit solution, it could become relatively unstable when the channel condition changes fast. To tackle this, we borrow the idea in the C1G2 standard [5, 32, 33] that uses Miller encoding to achieve a tradeoff between channel conditions and data rates. Specifically, RBLE enables 3 other different encoding coefficients: 2, 4, 8, which correspond to 2 μs, 4 μs, 8 μs encoding rates. This way, the tag can adapt different encodings based on channel qualities.

3.2.5 Downlink Communication

Control communication is required between the tag and the receiver because the tag and receiver need to be synchronized. The BLE receiver performs channel hopping according to the channel quality. Before the receiver hops to a new receiving channel, it needs to send channel parameters to the transmitter. The transmitter forwards these parameters to the tag through downlink communication and then continuously transmits exciting signals for backscatter until it receives the new parameters from the receiver again. Note that most complexities, e.g., when to hop channels, and what kind of encoding should be used, lie at the excitation and receiver BLE devices, the RBLE tag just follows orders from standard BLE devices, similar to the RFID tag design. Upon BLE signals are detected, the

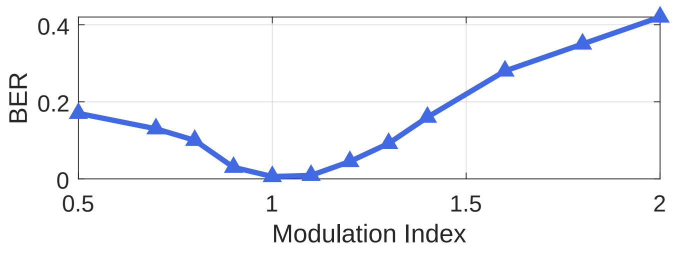


Figure 3.7: Adjustment of modulation index. We empirically test RBLE for various modulation indexes and found that when the modulation index is set to 1, the system has the lowest BER.

RBLE tag parses commands including hopping sequence, channel dwelling time, encoding coefficient, etc. Those core command parameters drive a state machine to dynamically configure channels and choose the encoding coefficient for BLE packet regeneration. For downlink communication mainly responsible for disseminating parameters, we adopt Packet Length Modulation (PLM) that is widely used by other state-of-the-art systems [90, 87, 89]. For better efficiency and confusion avoidance, the predefined sequence using PLM is used to trigger the command parsing process.

3.2.6 Modulation Index of RBLE

RBLE tag regenerates BLE packets using direct frequency shift modulation. Direct frequency shift is essentially a binary frequency shift keying (BFSK) modulation adapted to backscatter. BFSK uses a pair of discrete frequencies to transmit binary (0s and 1s) information [61].

The modulation index of commodity BLE is 0.5. The modulation index indicates how much the modulated variable varies around its unmodulated level. The modulation index (M) is given by:

$$M = 2 \cdot f_d / R \tag{3.4}$$

where R is the symbol rate of BLE signal which is 1 Mbps. f_d is frequency deviation. 0.5 is the smallest BFSK modulation index that can be chosen such that the waveforms for symbol 0 and symbol 1 are orthogonal. To regenerate a valid BLE signal, we first set the parameter like this when performing direct frequency shift modulation:

$$f_d = |\Delta f_2 - \Delta f_1|/2 = 250kHz \tag{3.5}$$

$$|\Delta f_2 + \Delta f_1|/2 = |f_{c2} - f_{c1}| \tag{3.6}$$

That is, the RBLE modulation index is set to 0.5, which is the same as commodity BLE transmitters. However, through experiments, we find that when the modulation index is 0.5, the modulation performance is poor. This is due to the fact that the GFSK modulation

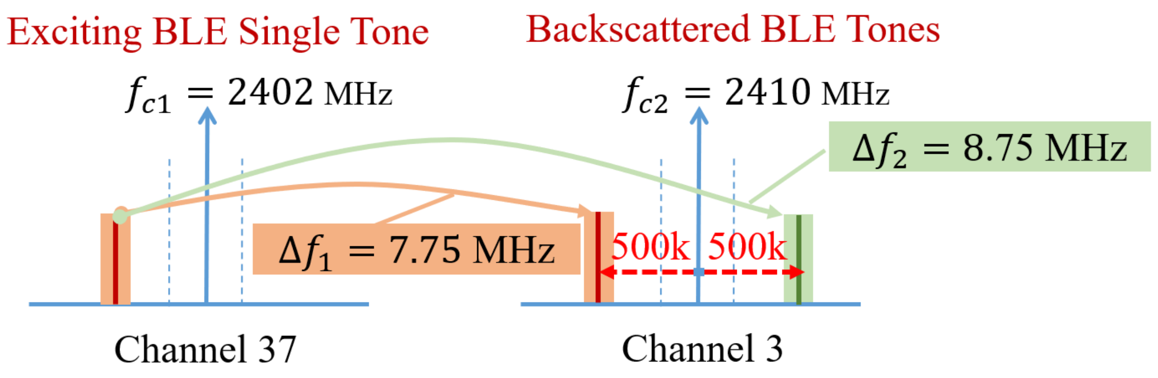


Figure 3.8: Direct frequency shift modulation example with a modulation index of 1. In this case, we set the parameter like this: $|\Delta f_2 - \Delta f_1|/2 = 500kHz$; $|\Delta f_2 + \Delta f_1|/2 = |f_{c2} - f_{c1}|$.

of commodity BLE is more complex than the modulation scheme of RBLE tag. GFSK supports a modulation index of 0.5, which is not the optimal modulation index for RBLE's BFSK modulation. So we need to find the most suitable modulation index for RBLE. We can easily adjust the modulation index of RBLE by changing Δf_1 and Δf_2 used in direct frequency shift. Since frequency deviation must be less than 1 MHz (Channel bandwidth of BLE is 2MHz), the modulation index should be in the range of 0.5 to 2. We empirically test RBLE for various modulation indexes and found that when the modulation index is set to 1, the lowest BER can be obtained at the receiver, as shown in Figure 3.7. The selected modulation index (1) ensures a sufficiently reliable modulation performance. In summary, we set the parameter like this when performing direct frequency shift modulation:

$$f_d = |\Delta f_2 - \Delta f_1|/2 = 500kHz \tag{3.7}$$

$$|\Delta f_2 + \Delta f_1|/2 = |f_{c2} - f_{c1}| \tag{3.8}$$

The bandwidth of the frequency-modulated signal is related to the modulation index. The approximate bandwidth of a frequency-modulated signal can be estimated by Carson's bandwidth rule:

$$BW = (M + 1) * 1/T_s \tag{3.9}$$

where T_s is the symbol duration of BLE (1 μs). M is the modulation index. Setting the modulation index to 1 achieves the lowest BER for BLE backscatter, but the bandwidth of the backscatter signal is increased accordingly. It is worth noting that this does not affect the reception of the backscatter signal by the BLE receiver because the backscatter signal is still within the receiving channel with a bandwidth of 2 MHz.

(a) BLE transceivers

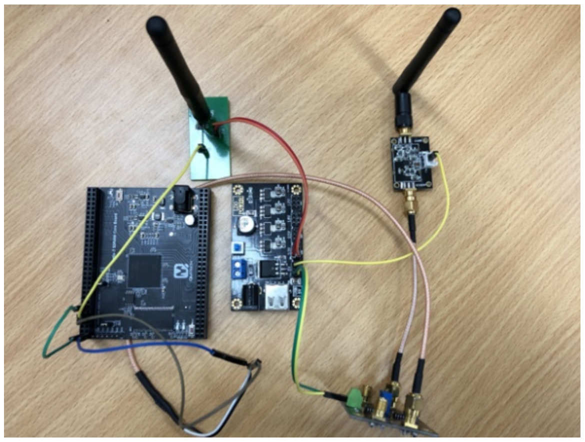

(b) RBLE tag prototype

Figure 3.9: The photos of TI CC2540 BLE transceivers and RBLE tag prototype.

3.3 Implementation

We build a prototype of RBLE using off-the-shelf BLE radios and customized backscatter tags. The implementation is detailed as follows.

BLE transceiver. We use TI CC2540 radios for both the excitation device and the receiver as shown in Figure 3.9(a). It transmits at 1 Mbps raw data rate and supports transmission power at 0 dBm and 4 dBm. The frequency deviation is 250 kHz and the modulation index is 0.5. The reversely-whitened sequence for each channel is computed offline.

RBLE tag. Our RBLE tag prototype shown in Figure 3.9(b) has two antennas, one for receiving and the other for backscatter. We use the AD8313 envelope detector connected to the receiving antenna. Its detection delay is measured around 0.4 μs, which is negligible for BLE applications. The backscatter antenna is connected to an ADG902 RF switch and baseband processing is implemented using an XILINX Artix-7. The modulation index is set to 1.

Low-power tag design. The biggest challenge for low-power tag design is the oscillator. Our solution is to employ the ring oscillator from [88, 46] that can generate a 35 MHz clock at the power consumption of 28 μW. We simulated the RBLE tag design using TSMC 65 nm technology and the overall power consumption is around 37 μW. 23 μW is consumed by the 30 MHz clock and 11 μW is needed for the RF switch. All the rest goes to running the control logic and modulation.

Experiment setup. Our experiments are conducted in indoor environments with line-of-sight and non-line-of-sight deployments. The downlink (transmitter-to-tag) distance is 0.3 m. For line-of-sight deployment, we move the receiver along a straight line to increase the uplink (tag-to-receiver) distance. For non-line-of-sight deployment, we place both the transmitter and tag in the room, then move the receiver in the corridor.

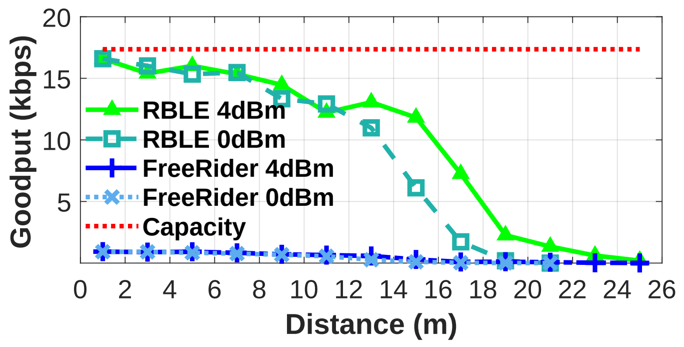
Metric. One of the performance metrics is the uplink goodput. There are two types of goodput. One is total goodput, which counts all the bits the tag modulates on the single tone. The other is the payload goodput, which only counts tag data bits transmitted. Note that there is no difference between these two goodput types for adv target channels because the backscattered adv packet is of the same length as the original exciting packet. But for data target channels, the total goodput is always higher than the payload goodput. Unless otherwise specified, we refer payload goodput as goodput in the rest of this paper. Although the random delay of the link layer for advertising events is unknown, we empirically confirm that the maximum advertising packet rate is stable at around 70 packets/s. Therefore, the goodput capacity using a single excitation device is about 17.4 kbps.
Competition. We mainly compare our system with FreeRider because it is so far the only backscatter system that works entirely with commodity BLE radios. Our FreeRider implementation is based on the Github codes published by the original author [6]. We set the FreeRider encoding length as 16-symbol for two reasons. The published data [6] uses this setting and for shorter encoding lengths, BERs surge significantly according to our experiments.

3.4 Evaluation

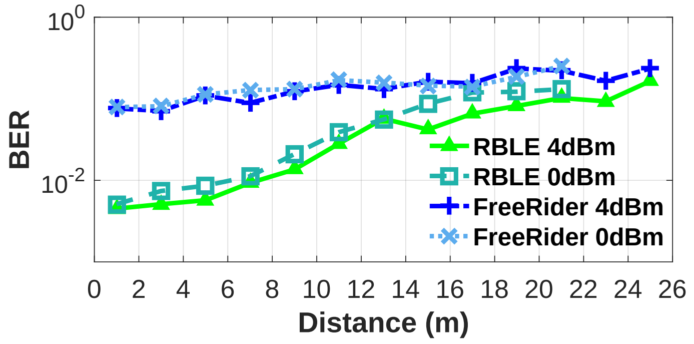
3.4.1 End-to-End Performance

Line-of-sight Deployment. Figure 3.10 shows the RBLE performance with increasing uplink distances in LoS deployment. The maximum goodput RBLE can achieve is 16.6 kbps. Given that the goodput capacity is 17.4 kbps, our system reaches up to 95% of the theoretical capacity. When the uplink range increases, RBLE is still able to achieve more than 10 kbps rates within 15 m. In addition, backscattered packets of RBLE can be decoded as far as 25 m. Such goodput and coverage are sufficient for many IoT applications.

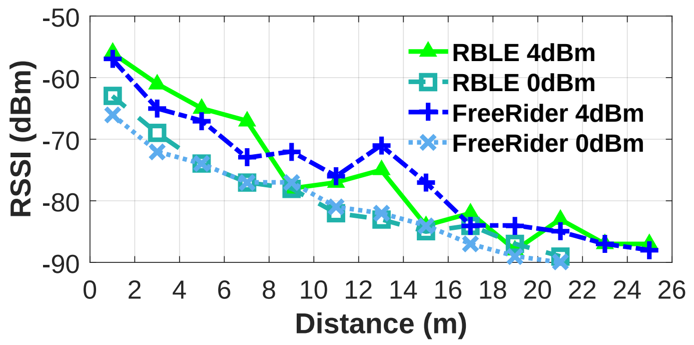
More importantly, RBLE achieves significant goodput gains over FreeRider for all cases. In particular, RBLE achieves a goodput of 15.4 kpbs at the distance of 3m and a goodput of 11.8 kbps at 15 m away, which are 17.3x and 40.2x better than counterparts of FreeRider. An interesting observation is that while RSSI gaps are not that obvious for RBLE and FreeRider shown in Figure 3.10(c), the BER gaps are distinct as in Figure 3.10(b). This is mainly because our direct frequency shift modulation is much more robust than codeword translation which either causes self-interference or leaves no solid signal at the desired frequency. In other words, our single-bit modulation can achieve much lower BERs than 16-symbol(bit) modulation of FreeRider. We note that the two-step modulation of FreeRider does not cause a significant impact on the power of the backscattered signal. Compared to RBLE, we did not add an analog mixer when replicating FreeRider. We use the FPGA to complete a digital mixing of the clock for codeword translation and the clock for frequency



(a) Goodput



(b) BER

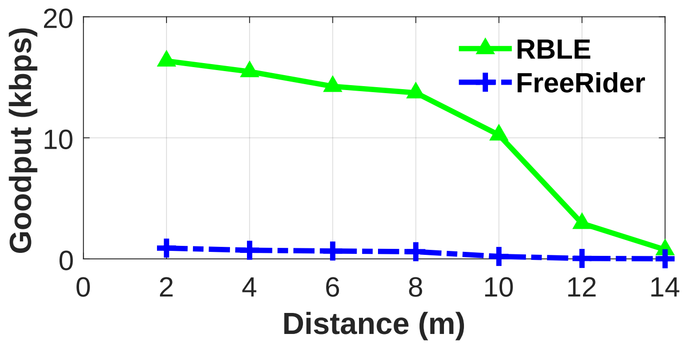


(c) RSSI

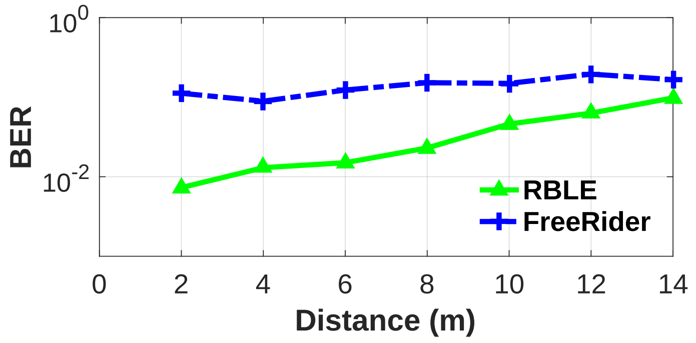
Figure 3.10: Backscatter goodput, BER, and RSSI across uplink distances in the line-of-sight deployment (downlink distance is 0.3 m).

shifting to the target backscatter channel. The intermediate frequency (IF) signal output from the GPIO port of FPGA will be mixed with the exciting signal in the RF switch.

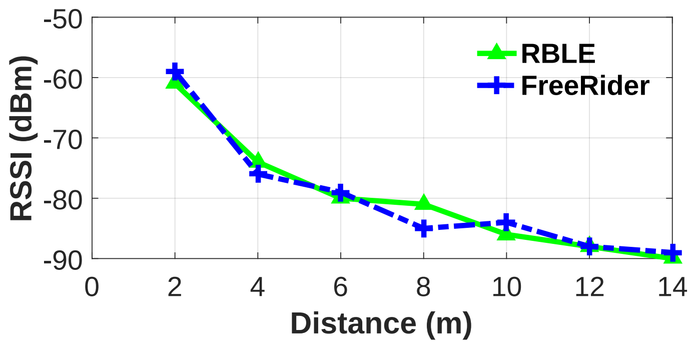
Non-line-of-sight Deployment. We also investigate the performance of RBLE in the non-line-of-sight scenarios. The exciting signal power is set at 4 dBm and both the transmitter and tag stay in the room while the receiver moves in the corridor. As shown in Figure 3.11(a), RBLE is still capable of decoding backscattered signals at the uplink distance of 14 m.



(a) Goodput



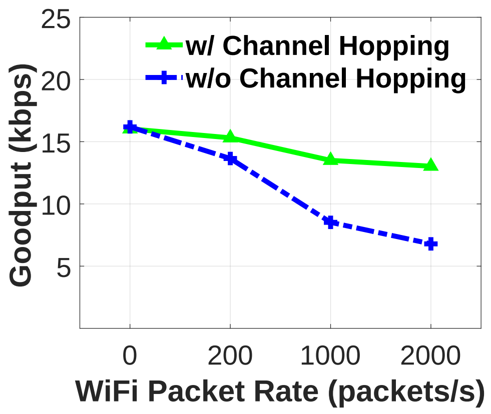
(b) BER



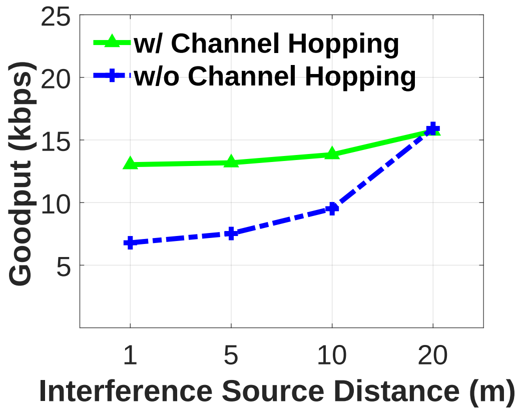
(c) RSSI

Figure 3.11: Backscatter goodput, BER, and RSSI across uplink distances in the non-line-of-sight deployment (downlink distance is 0.3 m).

Similar to LoS cases, the maximum achieved goodput is 16.4 kbps at a distance of 2 m. At further distances, RBLE still achieves more than 10 kbps within 10 m, which confirms that our RBLE is able to deliver decent BLE transmissions for short-range IoT applications, like headphone, smartwatch, and other personal electronics. Figure 3.11(b) shows that RBLE achieves low BERs even in NLoS cases, and the RSSI degrades to -90 dBm at 14 m so backscatter communication stops as well. This is because when the uplink distance increases



(a) Different WiFi packet rates.



(b) Different WiFi distances.

Figure 3.12: Impact of channel hopping on interfering WiFi sources with different WiFi packet rates and different interfering distances.

more than 14 m, the signal has to penetrate more walls. In consequence, the backscattered signal becomes too weak to decode the packet preamble.

3.4.2 Micro Benchmarks

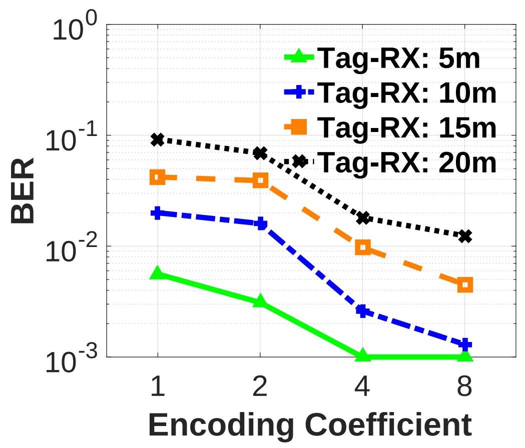
Impact of BLE Transmission Power. Next, we evaluate the performance of RBLE when the exciting BLE signals are transmitted at 0dBm and 4dBm. Figure 3.10(a) shows the goodput results and RBLE again outweighs FreeRider significantly. When the distance is less than 7 m, RBLE's BERs are basically below 1% for both power levels. In contrast, FreeRider's BERs are always above 7% even if it uses 16-symbol encoding, which confirms the superiority of direct frequency shift modulation over the two-step modulation of FreeRider. Regarding RSSIs in Figure 3.10(b), we can see that stronger transmission power brings longer uplink ranges. When the transmission power is 0 dBm, the backscattered packets can be received within 21m, and when the power increases to 4 dBm, the uplink distance reaches 25 m. There is no obvious gap between RBLE and FreeRider.

Impact of Channel Hopping. Then, we conduct a set of comparisons to examine the impact of channel hopping in the presence of WiFi interference. We let the interfering WiFi source work at 2412 MHz of 20 MHz wide. The center frequency of the target BLE data channel is 2414 MHz (data channel 5) where this BLE channel is completely within the range of the interfering WiFi channel. Exciting signals are transmitted on adv channel 38. The downlink distance is 0.3 m and the uplink distance is 5 m. We compare performance for two cases, without channel hopping and with channel hopping where the RBLE tag hops across a number of channels set by the exciting BLE signals. We refer to the channel hopping mechanism of active BLE and pre-set a channel hopping sequence on the receiving end. The receiver and tag perform channel hopping according to this sequence. The pre-set channel hopping sequence can not ensure that the tag will jump directly out of the interfered channels, but the tag will perform channel hopping across about 80 MHz. After the channel hopping is activated, the communication time on the interfered channels covered by the interference source will be significantly reduced. In the experiment, we set the receiver to calculate the BER every 10 seconds. If the BER exceeds a pre-set threshold, the receiver will hop to a new receiving channel. Before the hopping, it sends parameters (including the index of the new channel) to the transmitter, which forwards these parameters to the RBLE tag through downlink communication. Since we use one packet to modulate one bit, the downlink data rate is about 50 bps.

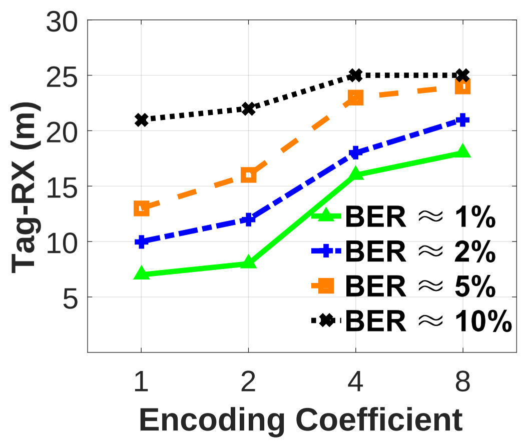
Figure 3.12(a) shows the goodput comparison for those two cases with different WiFi transmission rates. The interfering WiFi source is 1 m away from the BLE receiver. We observe that as the interfering level increases, the goodput of RLE without channel hopping degrades much more than that of RLE with channel hopping. For instance, when the WiFi packet rate increases from 200 to 2000, the goodput without channel hopping drops 6.9 kbps, from 13.7 kbps to 6.8 kbps, whereas the goodput with channel hopping only degrades 2.3 kbps, from 15.3 kbps to 13 kbps. In other words, RBLE with channel hopping achieves 1.92x goodput gain over the one without channel hopping in the presence of strong WiFi interference. Undoubtedly, such gains are brought by our dynamic channel configuration that enables channel hopping as soon as the BLE receiver discovers there are interfering sources and notifies RBLE tags. Figure 3.12(b) compares two cases against different interference source distances where the WiFi packet rate is fixed at 2000 packets/s. The results confirm that significant performance gaps exist between RBLE with channel hopping and the one without such. As the interference distance becomes further away, RBLE tends to get closer to the theoretical capacity goodput, 17.4 kbps. When the WiFi interference source is 20 m away, it basically does not affect our backscatter transmission.

Impact of Adaptive Encoding. In order to examine the impact of adaptive encoding on backscatter transmission, we conduct two sets of experiments. Results in Figure 3.13 demonstrate that with the help of adaptive encoding, BERs can be greatly reduced across a range of different uplink distances. For example, the first group of experiments shown in Figure 3.13(a) is to examine the effect of adaptive encoding on BER for different uplink ranges. When the uplink distance is 5 m, the BER can be reduced from 0.56% using M1 to 0.1% using M8. Also, when the uplink distance increases to 20 m, the BER drops from 9.2% using M1 to 0.45% using M8. The same observation can be made in the second group of tests shown in Figure 3.13(b). To keep the BER at a predefined level, increasing the encoding coefficient can enlarge uplink ranges accordingly.

Goodput of Adv and Data Channels. We examine the goodput of our regenerated adv and data packets. We compare the frequency modulation performance (BER) and goodput of five different target channels, which are adv channel 37, data channel 5, data channel 21, data channel 25 and adv channel 39. We set RBLE to hop across these channels.

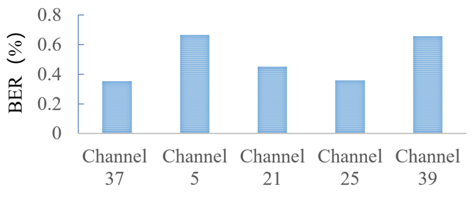


(a) BER VS Encoding coefficient.

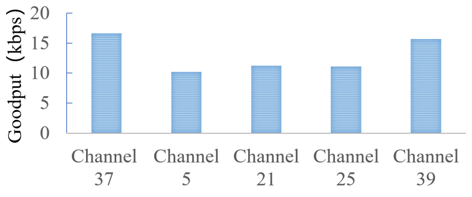


(b) Uplink distance VS Encoding coefficient.

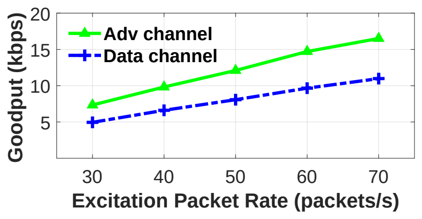
Figure 3.13: Impact of adaptive encoding on backscatter transmission. Figure 3.13(a) shows the impact of adaptive encoding on BER for different uplink distances. Figure 3.13(b) shows to keep the BER at a predefined level, increasing the encoding coefficient can enlarge uplink ranges.



(a) BER



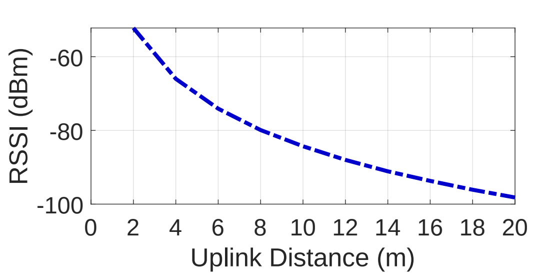
(b) Goodput



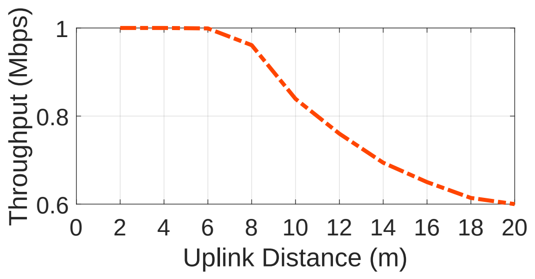
(c) Goodput under different excitation packet rates

Figure 3.14: BER and goodput for five different target channels. Figure 3.14(c) shows goodput differences between Adv and Data channels under different excitation packet rates.

Figure 3.14(a) shows the performance results of five channels. While the five channels use different frequency shifts, the bit error rates are all less than 1%. Figure 3.14(b) shows the backscatter goodput of the five channels. We can see that the goodput of data channels is less than that of adv channels. The main reason is that we reuse the preamble, access address, and header of the excitation packet while regenerating advertising packets. In



(a) Estimated RSSI



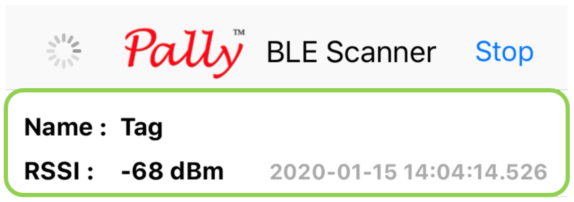
(b) Estimated throughput

Figure 3.15: Estimated RSSI and throughput at a variety of uplink distances.

particular, a regenerated adv packet has a payload of 31 bytes while a regenerated data packet has only 21 bytes for the data payload. The same observation can be made from Figure 3.14(c) as well where we examine goodput differences between Adv and Data channels under different excitation packet rates. We can see that the goodput varies almost linearly with the excitation packet rate.

Link Budget. Borrowing the theoretical model for bistatic radar [38, 25], we can estimate RSSIs given link parameters. The received power at the BLE receiver is estimated as $P_R = \frac{P_t G_t \Delta\sigma G_r \lambda^2}{(4\pi)^3 D_1^2 D_2^2}$, where D_1 is the downlink distance. D_2 is the uplink distance. P_t and G_t are the CW source output power and gain of the transmitter's antenna, respectively. G_r is the gain of the receiver's antenna. λ is the carrier-frequency wavelength. $\Delta\sigma$ is the differential radar cross-section (RCS) [58] given by $\Delta\sigma = \frac{\lambda^2}{4\pi} G_N^2 \left|\Gamma_1^* - \Gamma_2^*\right|^2$, where G_N is the antenna gain of the tag and Γ^* is the conjugate match reflection coefficient $\Gamma^* = \frac{Z_a^* - Z_L}{Z_a + Z_L}$ for a resonant antenna impedance Z_a and the complex load impedance Z_L.

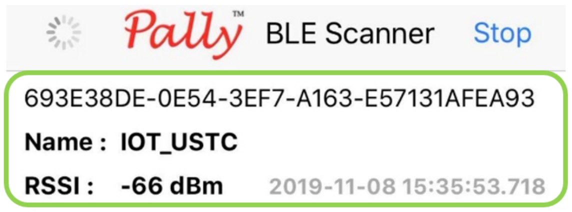
Z_{L1} and Z_{L2} are measured at the frequency of the exciting signal (2426 MHz in our case), representing impedance of the ADG 902 switch in both on and off states. Values for Z_{L1}, Z_{L2}, and Z_a are measured as $7 - j\ 52\ \Omega$, $12 + j\ 9\ \Omega$, and $50 + j\ 0\ \Omega$ respectively. The estimated RSSI at a variety of uplink distances D_2 is displayed in Figure 3.15(a). The downward trend of the estimated and measured RSSI is roughly similar. We can also roughly estimate the throughput through simulation, given the backscatter signal strength and the environmental noise floor (-85 dBm). We assume the modulation scheme of the tag is standard BFSK modulation and the demodulation method of the BLE receiver is standard BFSK demodulation. We also assume that the propagation channel is an additive white Gaussian noise (AWGN) channel. The estimated throughput at a variety of uplink distances D_2 is displayed in Figure 3.15(b). The physical layer data rate of BLE is 1 Mbps. SNR decreases as the uplink distance increases. As BER gradually increases, throughput gradually decreases.



Name : Tag

RSSI : -68 dBm 2020-01-15 14:04:14.526

(a) BLE advertising packets as excitation.



693E38DE-0E54-3EF7-A163-E57131AFEA93

Name : IOT_USTC

RSSI : -66 dBm 2019-11-08 15:35:53.718

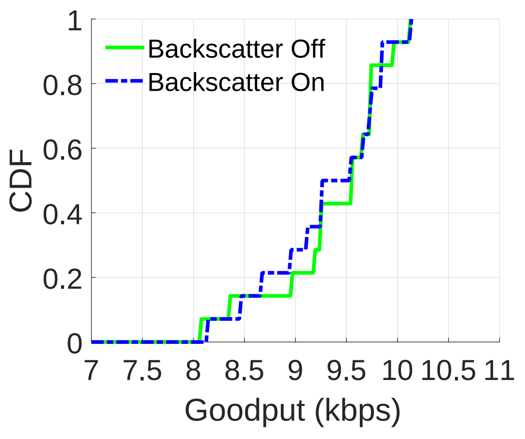
(b) BLE data packets as excitation.

Figure 3.16: Generated BLE packets received by an unmodified iPhone. Using BLE advertising packets as excitation, "Tag" can be included in the payload of the regenerated BLE packet. Using longer BLE data packets as excitation, "IOT_USTC" can be included in the payload of the regenerated BLE packet.

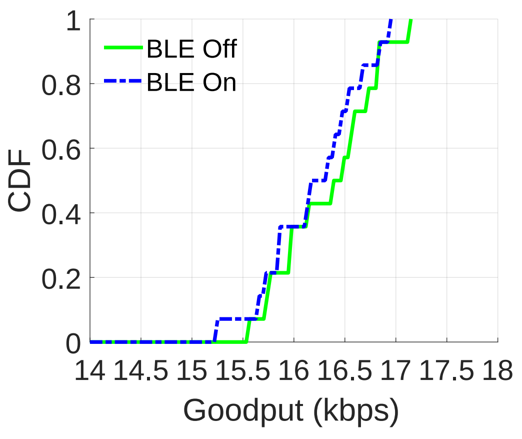
3.4.3 Compatibility with Commercial Radios

Phone-to-Phone Verification. To verify that RBLE is compatible with commodity smart devices with BLE radio, we conduct a phone-to-phone experiment. We use a Huawei P30 and an iPhone 8 Plus as excitation device and receiver respectively. We place the excitation phone at a distance of 0.3 m from the tag. The distance between the receiver (iPhone) and the tag is 2 m. We perform reversely-whitening techniques [46] so that the payload of the BLE advertising packet from the excitation phone can be made all ones or zeros, which are single tones. The target channel of RBLE tag is advertising channel 38. RBLE tag needs to regenerate a complete BLE packet, including the CRC field. Only when the CRC check result is correct, the smartphone will display the information of the received packet. Figure 3.16(a) shows the regenerated BLE packets received by an unmodified iPhone with the "BLE Scanner" software. Using BLE advertising packets as excitation, we can only get a single tone no more than 31 bytes. Thus, the length of the whole regenerated packet is no more than 31 bytes, the payload for modulating tag information is less than 21 bytes. RBLE can also be extended to work with BLE data packets. The data packet has a payload of up to 255 bytes. We reversely-whiten BLE data packets using the seed of a specific data channel so that the payload part of the data packet can be used as a longer single-tone to regenerate a BLE packet carrying more information. As shown in Figure 3.16(b), the name "IOT_USTC" is included in the payload of the regenerated BLE packet while using BLE data packets as excitation.

Co-existence with Ambient BLE Transmission. In addition, we investigate the co-existence of our backscatter system with ambient BLE devices. In this experiment, we deploy our backscatter system 1 m away from an ambient BLE transmitter, which transmits advertising packets on adv channel 37 at 40 packets/s. Our backscatter system's excitation sources are advertising packets on adv channel 38. Our RBLE tag backscatters the excitation signal onto adv channel 37. We temporarily stop channel hopping to fully investigate the interaction between the backscatter and ambient BLE signals on the same channel.

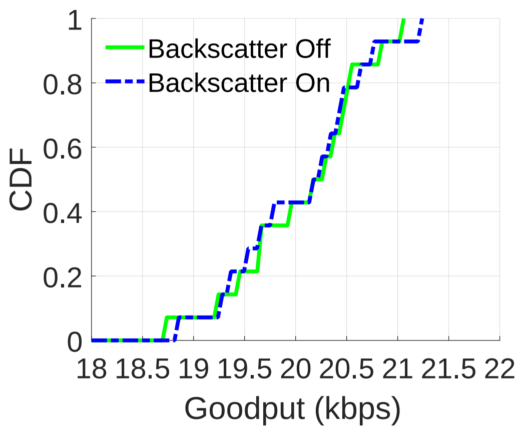


(a) BLE goodput

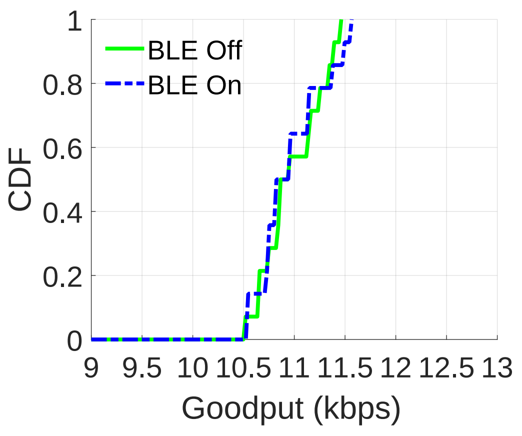


(b) Backscatter goodput

Figure 3.17: Impact of BLE and backscatter goodput on each other when sharing an advertising channel.



(a) BLE goodput



(b) Backscatter goodput

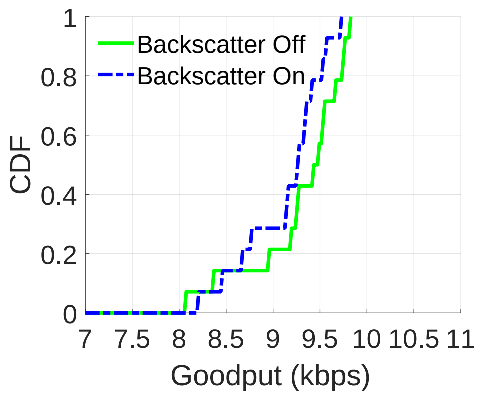
Figure 3.18: Impact of BLE and backscatter goodput on each other when sharing a data channel.

- **Backscatter's Impact on Ambient BLE:** Figure 3.17(a) shows the ambient BLE goodput when our backscatter is present and absent. When we turn on the backscatter transmission, the median BLE goodput is 9.3 kbps. When we turn off the backscatter transmission, the median BLE goodput is 9.5 kbps. Backscatter causes a very slight drop in ambient BLE's goodput, indicating the backscatter communication has no severe interference to the ambient BLE transmission. There are several reasons. First, the backscattered signal strength is usually below -55 dBm, much lower than the signal strength of the ambient BLE transmission. Moreover, our system enables backscatter communication during advertising events, which reduces the time of channel occupancy for backscatter transmissions. In addition, the advertising events have already been perturbed by the random delay, which somehow coordinates the advertising events of different BLE devices. In short, our system does not severely impact ambient BLE devices.

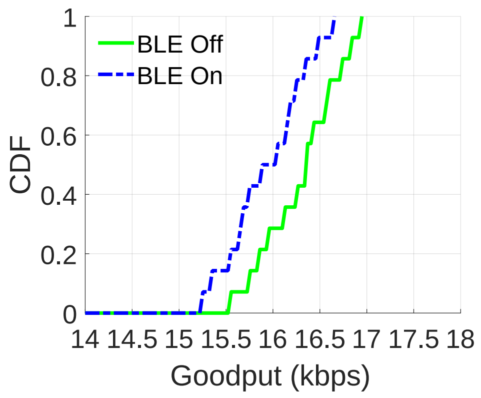
- **Ambient BLE's Impact on Backscatter:** Figure 3.17(b) shows the backscatter goodput when ambient BLE transmission is present and absent. When we turn off the ambient BLE devices, the median backscatter goodput is 16.4 kbps. When the ambient BLE is turned on, the median backscatter goodput decreases to 16.2 kbps. The ambient BLE's impact on backscatter is slight due to the short channel occupancy time of the backscatter transmission.

  We also evaluate the impact of backscatter communication when the backscatter signal is transmitted on a data channel. We deploy our backscatter system 1 m away from two ambient BLE devices which communicate on data channels after establishing a connection. Our RBLE tag backscatters the exciting signal onto data channel 21. As shown in Figure 3.18 the impact of BLE and backscatter goodput on each other when sharing a data channel is small, even smaller than the impact when sharing an advertising channel. We believe that this is because the time for backscatter and active BLE transmissions to share a data channel is extremely short. After two active BLE devices have established a connection, they will communicate in 37 data channels in a frequency-hopping manner, so the time to occupy a specific data channel is short. In addition, the channel occupancy time of the backscatter transmission is also short. As a consequence, the impact of backscatter communication on data-channel BLE transmissions is not obvious. RBLE can coexist well with ambient BLE transmissions on both advertising and data channels.

  We add two sets of experiments to evaluate the co-existence of backscatter and ambient BLE transmissions on the same advertising channel when increasing the backscatter signal strength and increasing the packet rate of the ambient BLE transmission respectively. For the first set of experiments, we add an amplifier to the excitation device to increase the backscatter signal strength. As shown in Figure 3.19(a), if we turn on the backscatter transmission, the median BLE goodput drops from 9.5 kbps to 9.25 kbps. The impact is slightly stronger compared to that in Figure 3.17(a). Relative signal strength impacts the ability of these systems to coexist to a certain extent. For the second set of experiments, we increase the packet rate of the ambient BLE transmission to 70 packets/s, which is the maximum advertising packet rate we can set. As shown in Figure 3.19(b), if we turn on the ambient BLE transmission, the median backscatter goodput drops from 16.3 kbps to 15.9 kbps. The ambient BLE's impact on backscatter is slightly stronger compared to that in Figure 3.17(b). The probability of collision impacts the ability of these systems to coexist. However, when



(a) BLE goodput when increasing the backscatter signal strength.



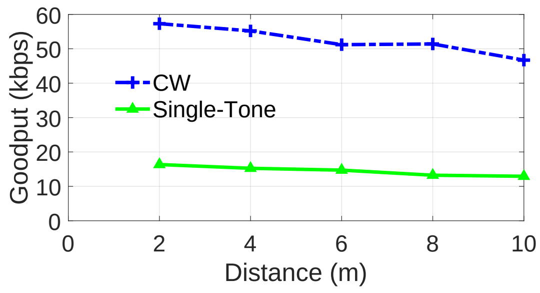
(b) Backscatter goodput when increasing the packet rate of BLE.

Figure 3.19: Impact of BLE and backscatter goodput on each other when increasing the backscatter signal strength and increasing the packet rate of the ambient BLE transmission respectively (on the same advertising channel).

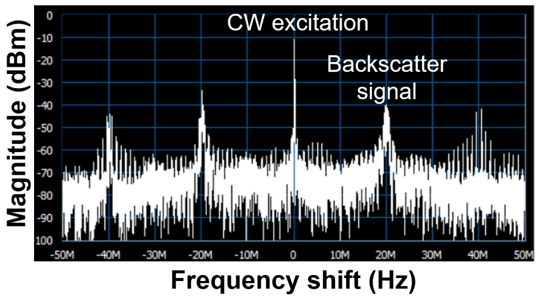
ambient BLE is transmitting packets at the maximum rate, the goodput drop caused by ambient BLE is not severely intolerable.

To summarize the above experimental results, when the backscatter signal and the ambient BLE share the same advertising channel, there is no strong mutual interference. In addition, both the ambient BLE devices and our backscatter tags have a channel hopping mechanism, which would further reduce collision chances. Therefore, our backscatter system can coexist well with ambient BLE transmissions.

Continuous CW Excitation. In addition to the single tone obtained by setting the application layer data of commodity BLE, we can also use the test mode of commodity BLE chips to generate CW signal. Using CW as excitation, we can freely control the packet generation rate of the RBLE tag. We experiment to compare these two types of excitations. For CW excitation, we set the packet generation rate of the tag to 250 packets/s. For single-tone excitation, the BLE transmitter transmits BLE advertising packets at a rate of 70 packets/s which is consistent with previous experimental settings. The tag's packet regeneration rate is limited by the excitation packet rate. We measure the backscatter goodput of two different types of excitations at the BLE receiver respectively. Backscatter goodput across distances in indoor LoS scenario is shown in Figure 3.20(a). When the uplink distance is within 10m, the goodput of CW excitation is always above 46 kbps and the goodput of single-tone excitation is always below 17 kbps. Compared to using single-tone excitation, using CW excitation can greatly increase the packet generation rate and thus improve the final backscatter goodput. We have already seen that the goodput varies almost linearly with the excitation packet rate in Figure 3.14(c). It can be further inferred



(a) Backscatter goodput across uplink distances in LoS scenario.



(b) Spectrum of the CW excitation and the backscattered signal.

Figure 3.20: Backscatter goodput and spectrum when using CW excitation.

that the goodput of RBLE is limited by the excitation packet rate. As long as the excitation packet rate is increased, the goodput of RBLE will also increase.

Figure 3.20(b) shows the spectrum of the CW excitation and the backscattered signal. The backscattered signal (generated BLE signal) is 20 MHz away from the CW excitation. The RBLE tag was configured to generate BLE advertising packets with a frequency shift of 20 MHz away from the CW excitation. We use a NI PXIe-5663 RF Vector Signal Analyzer as our spectrum analyzer. Note that the backscatter signal has harmonic components. These harmonics have a negative impact on the legacy transmissions on the other channels and we cannot guarantee that harmonics are not generated outside of the 2.4 GHz band. However, the negative impact of harmonics is very small. The energy of the first harmonic is about 10 dB lower than the energy of the main signal, and the energy of the higher harmonics will be lower. We have evaluated the coexistence of the main backscatter signal and ambient BLE transmission in subsection 3.4.3 and found that RBLE's main backscatter signal can coexist well with ambient BLE transmissions. The energy of the harmonics is much lower than the energy of the main backscatter signal, so the impact of the harmonics is also limited. Similar to prior systems, RBLE cannot fully eliminate backscatter harmonics, which we intend to investigate in the future.

3.5 Discussion

Productive and unproductive exciting signals. There are trade-offs in choosing an unproductive single tone or a productive data-carrying signal as the carrier. FreeRider requires the data sequence of exciting signals to decode the tag data. If the data sequence of the original channel is corrupted, it is difficult to decode tag data even when the data from the backscattered channel is error-free. Such productive-data dependency would significantly impact the BER of the tag data when the quality of the original channel becomes unstable. In addition, using a productive signal with bandwidth as a carrier will also degrade the BER performance of the modulation.

On the other hand, backscattering with productive exciting signals is a significant step towards exploiting rich ambient signals because it does not need to control the payload content of the exciting signal. We are actively looking for a compromise solution, which can not only use productive exciting signals but also guarantee the reliability of backscatter transmission. One way is to properly control the data content of exciting signals, maintaining a certain amount of productive data goodput.

Choice of codeword scheme. The codeword scheme used by RBLE is essentially a repetition code with several different code rates. The error correction capability of the repetition code is limited. We can adopt more complex codeword schemes with stronger error correction capability, such as BCH codes. If we choose BCH codes as the codeword scheme, we could use different code rates of BCH codes to complete the adaptive encoding.

3.6 Conclusion

In this paper, we have proposed RBLE, a reliable BLE backscatter design that works with a single commodity receiver. The main contributions lie in using BLE signals with partial single tones as excitations for BLE backscatter, leveraging dynamic channel reconfiguration to bypass interfered channels, and using adaptive encoding to further improve reliability for challenging low SNR scenarios. Comprehensive field studies demonstrate significant performance gains over state-of-the-art systems in terms of BER, goodput and uplink range. Our future work includes an investigation of how to extend RBLE to work with Bluetooth 5.x environments, parallel transmission for multiple BLE tags, and interactions with other protocols, e.g., WiFi and Zigbee.

Chapter 4

An Efficient Modulation Design For WiFi Backscatter

4.1 Background and Motivation

4.1.1 WiFi Backscatter Primer

OFDM WiFi The WiFi family consists of a series of over-the-air modulation techniques. 802.11 was the first widely accepted WiFi standard, which mainly relied on the direct-sequence spread spectrum (DSSS). Later, 802.11a/g/n/ac were proposed to provide much higher throughput and anti-interference using orthogonal frequency-division multiplexing (OFDM). Nowadays, OFDM WiFi is so popular that it dominates in-home and in-office net connectivity.

Although each OFDM WiFi protocol has its own distinctions, they all share much the same basic architecture, as shown in Figure 4.1. At the physical layer, we are supposed to transmit a bitstream over the air. The first operation is padding, which mainly patches a bunch of data fields to make the bitstream ready for OFDM processing, including SERVICE, TAIL, and PAD fields. Then it passes through a scrambler aiming to transform the original bitstream into a random bitstream that is not all zeros or ones, which can bring bad Peak-to-Average Power Ratio (PAPR) and degrades the efficiency of power amplifiers. The channel encoding, which mainly refers to BCC encoding for WiFi, is designed to make the scrambled bitstream more robust to channel interference as it can correct errors. Along with BCC, an interleaver comes in to further improve anti-interference performance. Specifically, it uses permutations to deal with cases where the number of burst errors exceeds the BCC correction capability. When all the bit-level operations are over, a constellation mapper will modulate each bit (0/1) to a complex number (IQ) according to modulation modes, e.g., BPSK, QPSK, QAM. Then the IFFT operation transforms the data from the frequency domain into the time domain where subcarriers are orthogonal. Through the final analog upconversion, all the baseband data are sent as RF over the air. Note that here we classify those operations into two categories: symbol-wide and payload-wide. For an operation, if the

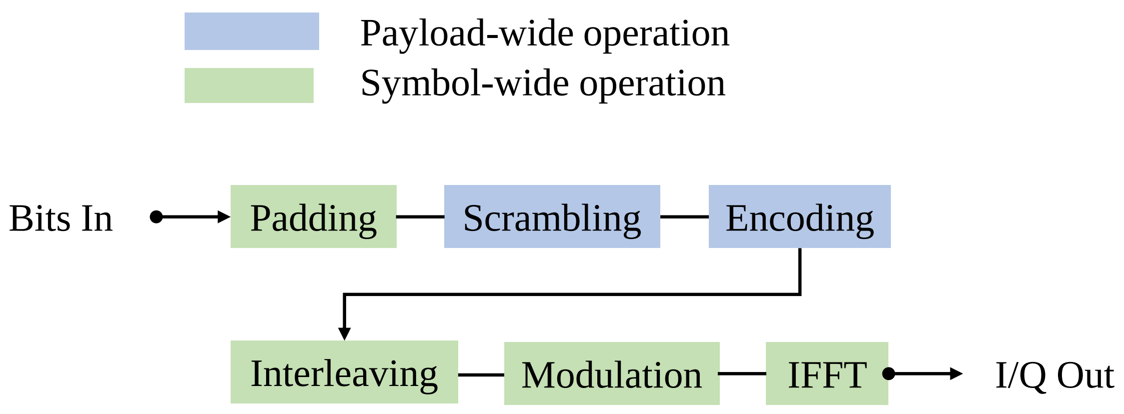


Figure 4.1: Typical transmitter design for OFDM WiFi.

processing in a symbol would not affect other symbols, we call it symbol-wide operation; otherwise, it is payload-wide. As such, padding, interleaving, modulation, and IFFT are symbol-wide operations while scrambling, and encoding are payload-wide operations.

Codeword Translation The seminal work, FreeRider [87], introduces codeword translation to backscatter tag data on the productive carriers. For the first time, this brings us closer to the vision that productive carriers, instead of CW, can be used as backscatter excitations. The idea of codeword translation is simple and elegant. It is common that codewords in the constellation map are related to each other by shifting phases, amplitudes or frequencies. For example, as per BPSK, there are two codewords, $C_1 = e^{j\theta_1}$, $C_2 = e^{j\theta_2}$, where $\theta_1 - \theta_2 = \pi$. So if we want to transmit a tag bit '1', we can translate the current codeword C_x by shifting π, and $C'_x = C_x \cdot e^{j\pi}$ is still a valid codeword. Similarly, if we want to transmit a tag bit '0', the current codeword keeps unchanged, which means no phase shift. As such, the tag bit can be decoded at the receiver by simply XORing, $C_x \oplus C'_x = t_x$.

Although a tag can choose from three degrees of freedom of a signal to do codeword translation, e.g., amplitude, phase, and frequency, the most viable way for OFDM WiFi is shifting phases. For amplitude backscatter, if a tag changes the amplitude on a subcarrier i, other subcarriers, e.g., subcarrier j, experience the same amplitude modification as well. While the amplitude change may lead to a legitimate codeword on subcarrier i, there is no guarantee that the modified codeword on subcarrier j is still valid [87]. For frequency backscatter, it is nearly impossible because all the codewords supported by OFDM WiFi do not have frequency-based relations. Therefore, phase-based codeword translation fits OFDM WiFi the most, and apparently, this operation is symbol-wide.

While codeword translation is easy to implement, it suffers from several drawbacks. One of the biggest challenges is the dependency on redundant modulation. Prior knowledge is that the need for such redundancy comes from burst errors [89] or interference from existing OFDM WiFi processing [87]. However, after extensive experiments and analysis, we discover that the root cause is the incompatibility between symbol-wide and payload-wide operations, which causes inter-symbol errors. Next, we will present how this problem arises and our solution.

4.1.2 Problems of Prior Systems

FreeRider [87] and MOXcatter [89], can provide symbol-level data transmissions for tags based on ambient OFDM WiFi signals, realizing about 60 kbps. They, however, still suffer from three main drawbacks.

- *Redundant modulation.* To ensure decent BERs for the receiver, most prior arts employ multi-symbol modulation for tag data. For example, FreeRider [87] modulates a tag bit using four OFDM symbols, and MOXcatter [89] does so with every two OFDM symbols. Such redundancy is introduced to combat the nonlinearity brought by tag modulation. When using every single OFDM symbol for tag modulation, those systems fail due to highly unstable demodulation errors. While multi-symbol modulation greatly reduces BERs, it leaves much room for throughput improvement.

- *Constrained ambient excitations.* To embed tag data onto legitimate WiFi packets, different excitation constraints have been attached to previous systems. Passive WiFi [49] requires an extra helper device to provide continuous waves. Interscatter [46] relies on the single-tones as carriers generated by de-whitened Bluetooth signals. MOXcatter [89] only supports WiFi excitations that have a payload of all 0s or 1s. In short, none of the state-of-the-art solutions can work with uncontrolled OFDM WiFi excitations at single-symbol rates.

- *Inconvenient two-receiver demodulation.* A seminal work, Hitchhike [85], proposes the first backscatter system that works with productive data at the symbol level. However, it requires two receivers to demodulate tag data. Several other works, e.g. MOXcatter [89], X-tandem [90], extend this idea in different ways and share the same limitation. Not only the accurate synchronization of two separate receivers is demanded, but more hardware costs will be incurred. Also, it's not applicable to personal IoT since today's mobile devices, e.g., smartphones and smartwatches, commonly support one single receiver.

4.1.3 Contribution

We introduce RapidRider, the first WiFi backscatter design capable of utilizing uncontrolled OFDM WiFi signals as excitations and effectively embedding tag data at a single-symbol rate. Our innovative approach brings us one step closer to achieving ubiquitous backscatter communications. Thanks to RapidRider, we can now freely use the abundant uncontrolled WiFi signals around us for backscatter communication.

In this paper, we first explain why previous systems have failed to achieve single-symbol backscatter, and then propose our solution, which is based on deinterleaved data. To enable the use of uncontrolled excitations, we have designed a deinterleaving-twins decoding scheme that incorporates both forward and backward deinterleavers. We have also conducted a detailed exploration of how the modulation and coding scheme (MCS) of the excitation affects tag demodulation, and have developed several translation methods.

To overcome the two-receiver limit, we have introduced a novel aggregated transmission mechanism that enables the transmission of productive data and tag data on the same packet. The key insight behind this mechanism is to use the pilot symbol as the reference symbol for tag data demodulation, thereby making a single receiver sufficient for decoding both types of data.

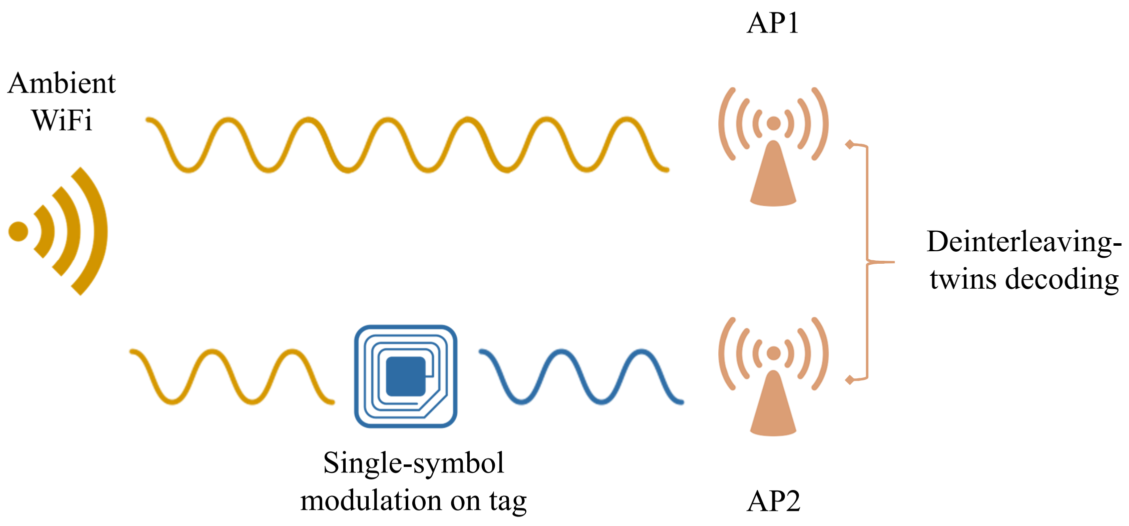
We prototype RapidRider using customized tags implemented by FPGAs, USRP N210, ZedBoards, and commodity NICs. Through comprehensive field studies, we demonstrate that:

- RapidRider achieves a maximum throughput of 235.3 kbps in LoS scenarios, which is 1.97x and 3.92x better than MOXcatter and FreeRider.
- For single-symbol modulation, the BER of RapidRider could be as low as 1.3%, while that of FreeRider is 43% under the same setting, demonstrating that RapidRider is state-of-the-art for single-symbol OFDM backscatter.
- Different MCS settings for excitation lead to different patterns of the bitstream after XOR. The translation methods, which translate the bitstream into a single bit, have a great impact on BER, and majority voting is the most common and efficient one.
- When there is only one receiver available, RapidRider+ achieves an aggregate goodput of productive and tag data about 1 Mbps on average. In particular, when aggregated transmission coefficient $\gamma = 1/3, 1/2, 2/3$, aggregated goodputs are 698.6 kbps, 950.9 kbps, 1,243.5 kbps.
- RapidRider is robust in diverse environments, including LoS and NLoS, indoor and outdoor, different excitation protocols and frequencies, excitation rates, different MCS of excitation, and interferences.

4.2 System design

4.2.1 Overview

As shown in Figure 4.2, the tag takes ambient WiFi as excitations and then applies single-symbol modulation to embed sensor data onto the carrier. To avoid interference, the backscat-



AP1
Ambient WiFi
Deinterleaving-twins decoding
Single-symbol modulation on tag
AP2

Figure 4.2: RapidRider framework.

tered signals are shifted to another WiFi channel from the original ambient channel. At the receiver side, there are two APs, one for receiving ambient signals and the other for backscattered signals. Then our novel deinterleaving-twins decoding method combines forward and backward deinterleaving and successfully recovers tag data at a very low BER. To turn this high-level idea into practice, it has several critical challenges: 1) how to demodulate single-symbol backscattered signals? 2) how to work with uncontrolled ambient signals? 3) how to enable a single AP to receive both tag and productive data?

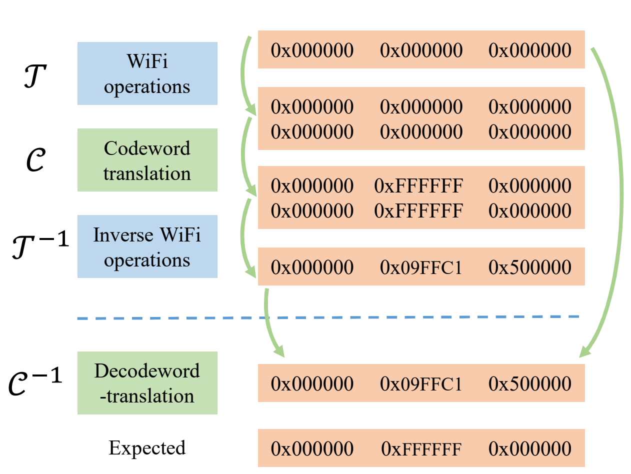
4.2.2 Single-Symbol Modulation

To get a better understanding of the whole backscatter procedure, we go through it in a formal way. At the transmitter side, suppose we have a bunch of OFDM symbols as payload, $t = (t_1, t_2, ..., t_n)$. Then this payload has to go through a number of WiFi operations as shown in Figure 4.1,

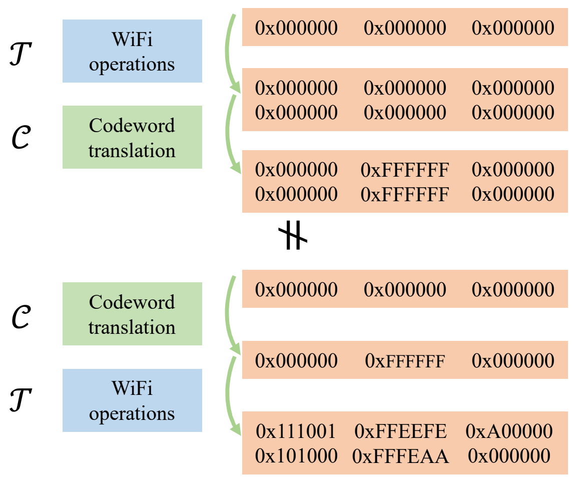
$$\mathcal{T}(\cdot) = \mathcal{T}_{IFFT}(\mathcal{T}_m(\mathcal{T}_i(\mathcal{T}_e(\mathcal{T}_s(\mathcal{T}_p(\cdot)))))), \tag{4.1}$$

where $\mathcal{T}_p$, $\mathcal{T}_s$, $\mathcal{T}_e$, $\mathcal{T}_i$, $\mathcal{T}_m$, $\mathcal{T}_{IFFT}$ denote padding, scrambling, encoding, interleaving, modulation, and IFFT, respectively. After the tag backscatters OFDM signals, the received backscattered signals, b, are the time-domain product of tag signals $s = (s_1, s_2, ..., s_n)$ and excitations,

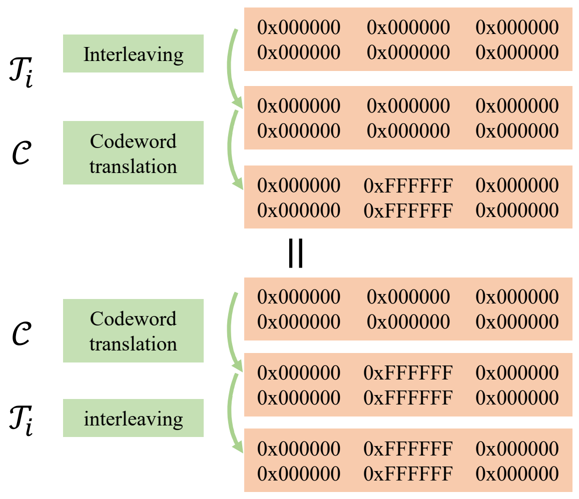
$$b = \mathcal{C}(\mathcal{T}(t), s) = (\mathcal{T}(t)_1 \oplus s_1, ..., \mathcal{T}(t)_n \oplus s_n), \tag{4.2}$$


$\mathcal{T}$
WiFi operations
$\mathcal{C}$
Codeword translation
$\mathcal{T}^{-1}$
Inverse WiFi operations
$\mathcal{C}^{-1}$
Decodeword -translation
Expected
0x000000 0x000000 0x000000
0x000000 0x000000 0x000000
0x000000 0x000000 0x000000
0x000000 0xFFFFFF 0x000000
0x000000 0xFFFFFF 0x000000
0x000000 0x09FFC1 0x500000
0x000000 0x09FFC1 0x500000
0x000000 0xFFFFFF 0x000000

(a) Backscatter decoding process.


$\mathcal{T}$
WiFi operations
$\mathcal{C}$
Codeword translation
0x000000 0x000000 0x000000
0x000000 0x000000 0x000000
0x000000 0x000000 0x000000
0x000000 0xFFFFFF 0x000000
0x000000 0xFFFFFF 0x000000
$\nparallel$
$\mathcal{C}$
Codeword translation
$\mathcal{T}$
WiFi operations
0x000000 0x000000 0x000000
0x000000 0xFFFFFF 0x000000
0x111001 0xFFEEFE 0xA00000
0x101000 0xFFFEAA 0x000000

(b) $\mathcal{T}$ and $\mathcal{C}$ are not commutative.


$\mathcal{T}_i$
Interleaving
$\mathcal{C}$
Codeword translation
0x000000 0x000000 0x000000
0x000000 0x000000 0x000000
0x000000 0x000000 0x000000
0x000000 0x000000 0x000000
0x000000 0xFFFFFF 0x000000
0x000000 0xFFFFFF 0x000000
$\parallel$
$\mathcal{C}$
Codeword translation
$\mathcal{T}_i$
interleaving
0x000000 0x000000 0x000000
0x000000 0x000000 0x000000
0x000000 0xFFFFFF 0x000000
0x000000 0xFFFFFF 0x000000
0x000000 0xFFFFFF 0x000000
0x000000 0xFFFFFF 0x000000

(c) $\mathcal{T}_i$ and $\mathcal{C}$ are commutative.

Figure 4.3: A formal presentation of a complete backscatter decoding process. (a) shows that decoding of single symbol backscatter does not work on the payload level; (b) explains the root cause of such failures comes from WiFi operations $\mathcal{T}$ and codeword translation $\mathcal{C}$ are not commutative; (c) demonstrates that interleave operation $\mathcal{T}_i$ and codeword translation $\mathcal{C}$ are commutative, which is the basis of our deinterleaving-twins decoding scheme.

where $\mathcal{C}$ denotes codeword translation. For payload demodulation, the backscattered signals at a WiFi AP have to go through a series of reverse processing, denoted as $\mathcal{T}^{-1}(\cdot)$. Hence, the backscattered payload can be written as $r = \mathcal{T}^{-1}(b) = (r_1, r_2, ..., r_n)$.

At the same time, from another AP at the original channel, we would correctly receive t. Together, the tag data, $\hat{s}$, is decoded by combining t and r as follows,

$$\hat{s} = \mathcal{C}^{-1}(t, r) = (t_1 \oplus r_1, ..., t_n \oplus r_n). \tag{4.3}$$

The above whole process seems perfect, but in reality, it cannot achieve the goal, which is $t_i \oplus r_i = s_i$.

The devil is in the detail. The root cause is that although all WiFi operations and codeword translation (and their inverse counterparts) are deterministic, there is no way to guarantee that $t_i \oplus r_i = s_i$ unless those WiFi operations are commutative with codeword translation. Let us examine this point in detail. Recall that to decode tag data, it goes through WiFi operations, codeword translation, inverse WiFi operations, and then decodeword-translation, which is $\hat{s} = \mathcal{C}^{-1}(\mathcal{T}^{-1}(\mathcal{C}(\mathcal{T}(t), s)), t)$. The problem is that $\mathcal{T}$ and $\mathcal{C}$ are not commutative [1], so $\mathcal{C}$ and $\mathcal{C}^{-1}$, $\mathcal{T}$ and $\mathcal{T}^{-1}$ cannot cancel each other to obtain s, which means $\hat{s} \neq s$. An toy example is shown in Figure 4.3a, where decoded $\hat{s}$ is (0x000000, 0x09FFC1, 0x500000) while the expected s is (0x000000, 0xFFFFFF, 0x000000). The checking processing of commutativity is shown in Figure 4.3b, indicating that $\mathcal{C}(\mathcal{T}(t), s) \neq \mathcal{T}(\mathcal{C}(t, s))$. It is just because $\mathcal{T}$ and $\mathcal{C}$ do not commute with each other, none of the prior schemes can decode backscattered OFDM signals on the single-symbol level. While those prior schemes, e.g., MOXcatter [89], FreeRider [87], have observed the same difficulty of single-symbol decoding, they all choose redundant encoding. In contrast, we are the first to discover this root cause, the commutativity between codeword translation and WiFi operations. Thus, we are able to design a single-symbol demodulation scheme based on this insight, which is to keep some of the whole WiFi operations that are commutative with codeword translation and to avoid those non-commutative ones. This deinterleaved-data based scheme works as follows.

Unlike the priors that perform decoding on the payload level, our proposal works at the deinterleaved-data level. As shown in Figure 4.5a, after two receivers obtain RF signals, they apply inverse WiFi operations, including FFT, Demodulation, and Deinterleaving, to acquire two deinterleaved data streams. Then we apply decodeword translation (XOR) on the two streams to recover the tag data. Formally, this recovered data can be written as

$$\hat{s'} = \mathcal{C}^{-1}(\mathcal{T'}^{-1}(\mathcal{C}(\mathcal{T'}(t'), s)), t'), \tag{4.4}$$

$$\mathcal{T'}(\cdot) = \mathcal{T}_{IFFT}(\mathcal{T}_m(\mathcal{T}_i(\cdot))), \tag{4.5}$$

[1]The commutative property means for two functions g and f, $g(f(\cdot)) = f(g(\cdot))$.

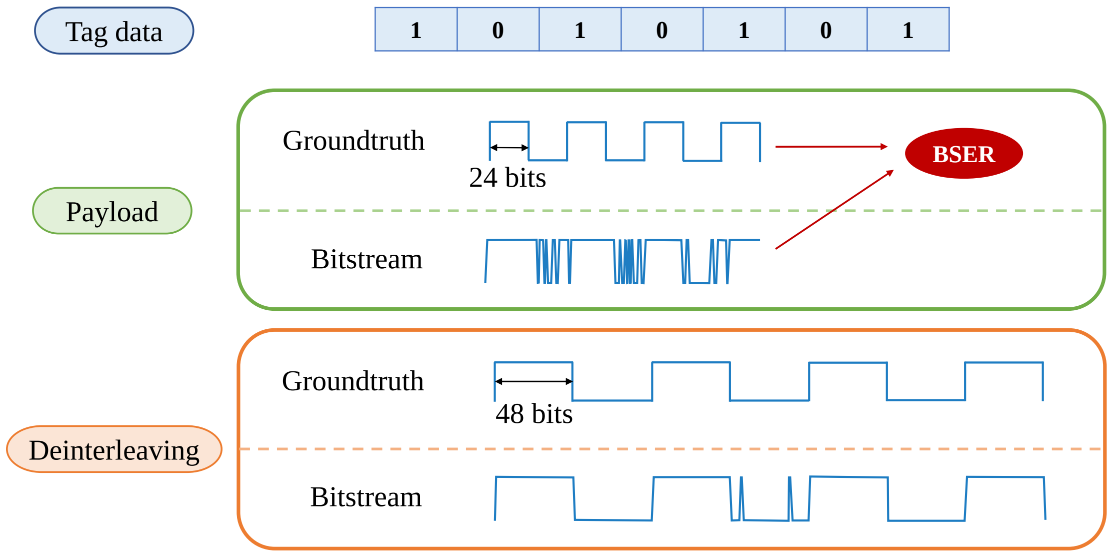


Figure 4.4: The conceptual diagram of Bitstream Error Rate (BSER).

where t' is the data before interleaving, which means $t' = \mathcal{T}_e(\mathcal{T}_s(\mathcal{T}_p(t)))$. Hence, if $\mathcal{T}'(\cdot)$ is commutative with codeword translation,

$$\begin{aligned} \hat{s}' &= \mathcal{C}^{-1}(\mathcal{T}'^{-1}(\mathcal{C}(\mathcal{T}'(t'), s)), t') && (4.6) \\ &= \mathcal{C}^{-1}(\mathcal{T}'^{-1}(\mathcal{T}'(\mathcal{C}(t', s))), t') && (4.7) \\ &= \mathcal{C}^{-1}(\mathcal{C}(t', s), t') && (4.8) \\ &= s. && (4.9) \end{aligned}$$

Now the only thing left is to verify whether $\mathcal{C}$ and $\mathcal{T}'$ commute with each other. Apparently, there are three symbol-wide operations with $\mathcal{T}'$ now and we observe that as long as the operation is symbol-wide, it is commutative with $\mathcal{C}$. In particular, for the first operation IFFT, it is well known for its linearity [67]. For any complex number, a, if $\mathcal{F}(x_n)_k = X_k$, then $\mathcal{F}(ax_n)_k = aX_k$, where $\mathcal{F}$ denotes IFFT (or FFT), and k is the index. Since codeword translation just applies a phase shift, $a = e^{j\theta}$, the FFT's linearity ensures its commutativity for $\mathcal{C}$. The second operation is modulation, which only performs one-on-one mapping process. Because codeword translation is performed within all constellation points, demodulation is obviously commutative to codeword translation. The last operation is interleaving, which re-orders the bitstream symbol by symbol, and thus no cross-symbol interference happens. As shown in Figure 4.3c, interleaving is commutative with codeword translation. Since all three operations commute with $\mathcal{C}$, $\mathcal{T}'$ is commutative with $\mathcal{C}$ as well.

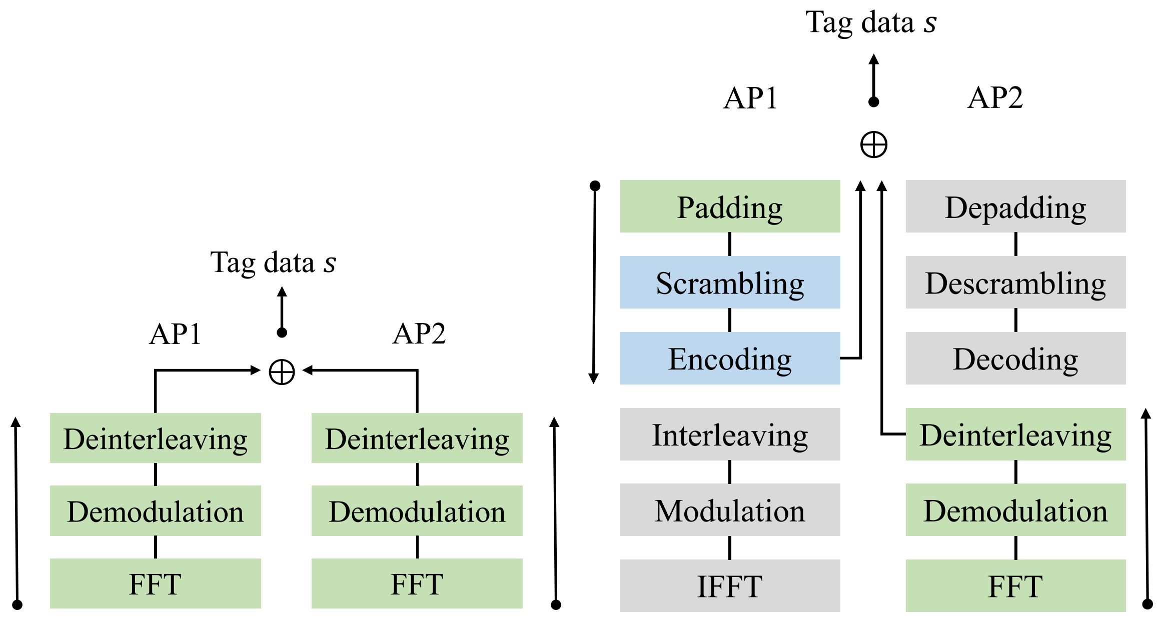
We have qualitatively proved that when the operations are symbol-wide, these processes are commutative with $\mathcal{C}$. However, we still want to be able to quantify the different impacts of symbol-wide and payload-wide operations on commutativity. Although BER, the existing well-known parameter, can describe the bit error rate of tag data well, it can't reflect the error rate of intermediate operations directly. Therefore, a new metric, BitStream Error Rate (BSER), is defined to describe the error rate of the intermediate bitstream to highlight the impact of each operation. For a receiver, the data exists in the form of a bitstream after the demodulation step. So we focus on the later layers. The idea of BSER is shown in Figure 4.4. First, expand tag data into bitstream as groundtruth, in which each tag data is expanded into a subsequence containing the same number of bits as a symbol. In different layers, the number of bits contained in a symbol is different. For example, in the interleaving layer, a symbol contains 48 bits, while the final payload layer contains only 24 bits. Then, obtain a bitstream by XORing the data of the same layer at the two receivers. Finally, calculate the BSER, a ratio of the number of wrong bits to the total number of bits in the bitstream.

There are two main reasons why we use BSER as a new evaluation metric. First, it's more direct than BER and can accurately reflect the performance of each layer. When calculating BER, it is necessary to translate the XORed bitstream into tag data first, and then calculate the error rate. So the value of BER is highly related to the translation method, which will be discussed in 4.2.3. In comparison, BSER calculates the error rate of the bitstream directly without going through the translation process, so it won't be affected by the translation method. Second, by comparing the BSER values of the bitstream before and after a certain operation, we can clearly see the effect of each operation on the accuracy of the final decoding. Theoretically, when going through a symbol-wide operation, the value of BSER is small since the operation is commutative with $\mathcal{C}$. But when the operation is packet-wide, the value of BSER will grow rapidly because of the non-commutativity. All in all, we can intuitively verify our theoretical analysis through BSER.

4.2.3 Deinterleaving-Twins Decoding

While the aforementioned single-symbol demodulation scheme works well in principle, an issue arises in practice: the excitation signal is always intended for commercial NICs, e.g., wireless router, smartphones, and smartwatches, which means if we try to use a software defined radio to acquire deinterleaved data of excitations, it appears not productive as the commercial NIC already recover excitation bits. Hence, we want to reuse the recovered payload bits from the commercial NIC, instead of an SDR, to do codeword translation. One big challenge of using this payload-level bits is we cannot do codeword translation on the payload level due to the mentioned commutativity problem. To address this, we design a deinterleaving-twins strategy as follows.

We first define a forward deinterleaver is that the received IQ data on an AP go through fft, demodulation and deinterleaving, as shown in Figure 4.5a. Similarly, a backward deinterleaver is that the payload data go through padding, scrambling, and encoding. Because all WiFi operations are invertible, it is easy to prove that a forward deinterleaver and a backward interleaver should generate the same deinterleaved data. The difference is a back-



(a) 2 forward deinterleavers. (b) 1 forward, 1 backward.

Figure 4.5: Two deinterleaving-twins decoding paradigms. (a) two forward deinterleavers; (b) one forward deinterleaver and one backward deinterleaver.

ward deinterleaver takes payload data as input and a forward one takes received IQ data. Hence, this backward and forward translation can be written as

$$\mathcal{T}_i^{-1}(\mathcal{T}_m^{-1}(\mathcal{T}_{IFFT}^{-1}(\hat{t}))) = \mathcal{T}_e(\mathcal{T}_s(\mathcal{T}_p(t))), \hat{t} = \mathcal{T}(t). \tag{4.10}$$

Based on this equation, as shown in Figure 4.5b, our deinterleaving-twins decoding contains a backward deinterleaver for payload data from a commercial NIC (AP1), and a forward deinterleaver for received IQ data from backscattered signals (AP2). Everything seems perfect; however, there remains a problem when we do backward deinterleaving: the commercial NIC does not provide access to the scrambler seed, which is the key for scrambling. Our insight is although there is no way to access the scrambler seed on AP1, we can recover AP2's scrambler seed. As long as we do not codeword translate the scrambler seed during backscattering, the AP2's seed would be the same as AP1's seed. To do so, we observe that the scrambler seed is at the beginning fields of the payload, and we can skip these fields to keep them intact. Therefore different from prior schemes where the start position of codeword translation is at the beginning of the payload, we choose this initial position as the next symbol of the scrambler seed. This way, backward deinterleaved data can be generated without any problem, and together with the forward deinterleaved data, recovering tag data just requires XORing and translation operations. XORing is simple, while translation needs to select an appropriate method to translate bitstream after XORing into tag data according to the characteristics of the bitstream.

Now, we can already use uncontrolled environmental signals as excitation. However, another problem is that the MCS of excitation signals in the environment are diverse. Most of the current work is based on the case where the modulation mode of the excitation signal is BPSK. To make our system more universal, we explore the case of different MCS. We focus on the situation when MCS = 0, 1 and 3, and the corresponding modulation modes are BPSK, QPSK, and 16QAM. We use BPSK to modulate the tag data, and the constellation diagrams under the three MCSs are shown in Figure 4.6. When the tag data is 0, the phase does not change under any MCS, so the bitstream after XOR is all 0s. Therefore, we focus on the situation when the tag data is 1. When MCS = 0 or 1, the bitstream after XOR of the original data and the data with a phase shift of 180° is all 1s. This feature has been widely used in the current work. Surprisingly, we have an interesting discovery when MCS = 3. When the modulation mode of the excitation is 16QAM, the patterns after XOR are all 1010. So the bitstream after XOR is an alternating sequence of 1s and 0s, which is quite different from the first two cases.

However, it is not enough to only get the pattern of the bitstream after XOR. To recover the tag data from the bitstream, a suitable translation method is also required. We can see that the pattern is different under different conditions. So, the translation method is very important for the correctness of data recovery, which can be reflected by BER. We'll have an in-depth discussion of translation methods, which has never been done before. The common denominator is that we do all the translation by using a decoding window, and the length of the window is the same as the number of symbols for tag modulation. The core difference is that the translation method needs to be designed according to the pattern in the bitstream.

For the case where the pattern is all 1s, three methods can be used: majority voting, subsequence matching, and Jaccard similarity.

Majority Voting. The idea of majority voting is to calculate the ratio of '1' or '0' in a certain window. When the ratio exceeds the threshold, it is translated to '1' or '0'. Since there are two options to calculate '1'or '0', this method can be further subdivided into two categories. One is that when the ratio of 1 exceeds the threshold, the data is recovered to be '1', otherwise, it is considered to be '0'. And the other is the opposite.

Subsequence Matching. The basis of subsequence matching is that the bitstream after XOR is consecutive '1' or '0', so the idea of this method is to use a subsequence of consecutive 0s or 1s of a certain length for matching. Like the previous one, it can also be subdivided into two categories. One is to use a subsequence of consecutive '1' for matching, if successful, the data bit will be translated as '1', otherwise, it is '0'. The other uses a subsequence of consecutive '0'.

Jaccard Similarity. The third method is based on vector similarity calculation. There are many ways to calculate similarity, and finally, the Jaccard coefficient has been chosen

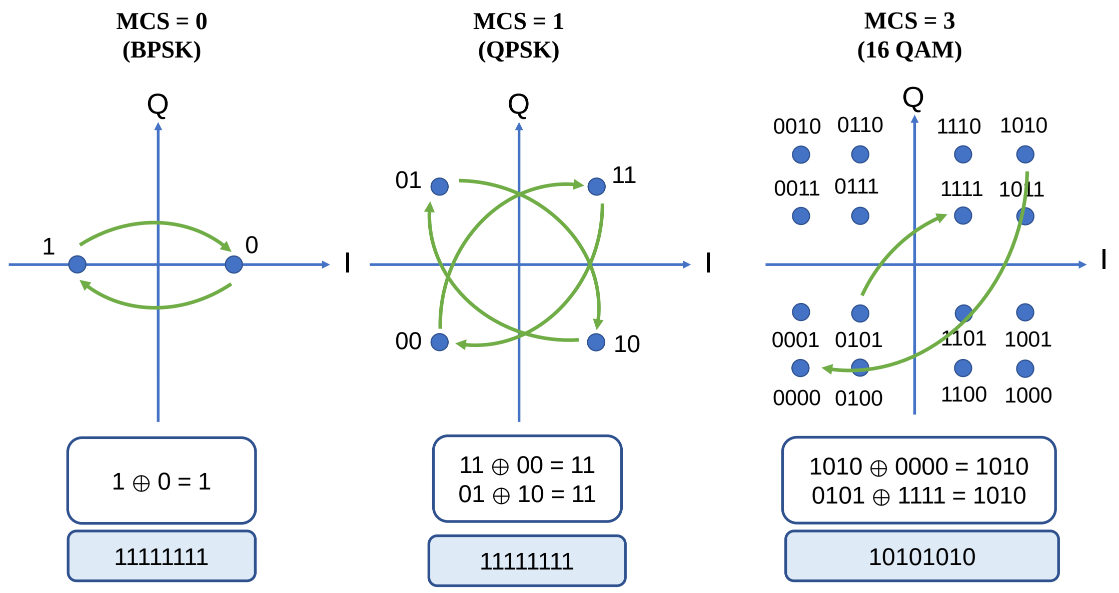


Figure 4.6: Constellation for different MCS when using BPSK modulation on tag data. The pattern is all 1s when MCS $= 0/1$, and is alternating sequence of 1s and 0s when MCS $= 3$.

because it is more suitable for logical vectors, which contain only '0' or '1'. The formula is shown in Eq.4.11. We regard the bitstream in the decoding window as a vector and set a vector of all 1s as the reference template. Then, calculate the Jaccard similarity coefficient of two vectors. When the result exceeds a certain threshold, this data is considered to be '1', otherwise, it is '0'.

$$jac = \frac{M_{11}}{M_{01} + M_{10} + M_{11}} \tag{4.11}$$

For the case where the pattern is an alternating sequence of '0' and '1', although theoretically, the above three methods can be modified and then applied to this situation, it is found in the actual experiment that only majority voting can have satisfactory results. The reason is that there will be some individual '0' or '1' insert in the middle, like '101101' or '010010', which makes it difficult to form a '10' alternating subsequence of stable length. So subsequence matching and Jaccard similarity calculation do not work well in this case. Because the majority voting method only cares about the ratio of '0' or '1', the only thing that needs to be done is to modify the threshold settings.

Although it is difficult to form a continuous '10'-sequence of sufficient length, fortunately, this alternation can be well maintained in most cases. Therefore, a new method, difference summation, is designed based on this phenomenon.

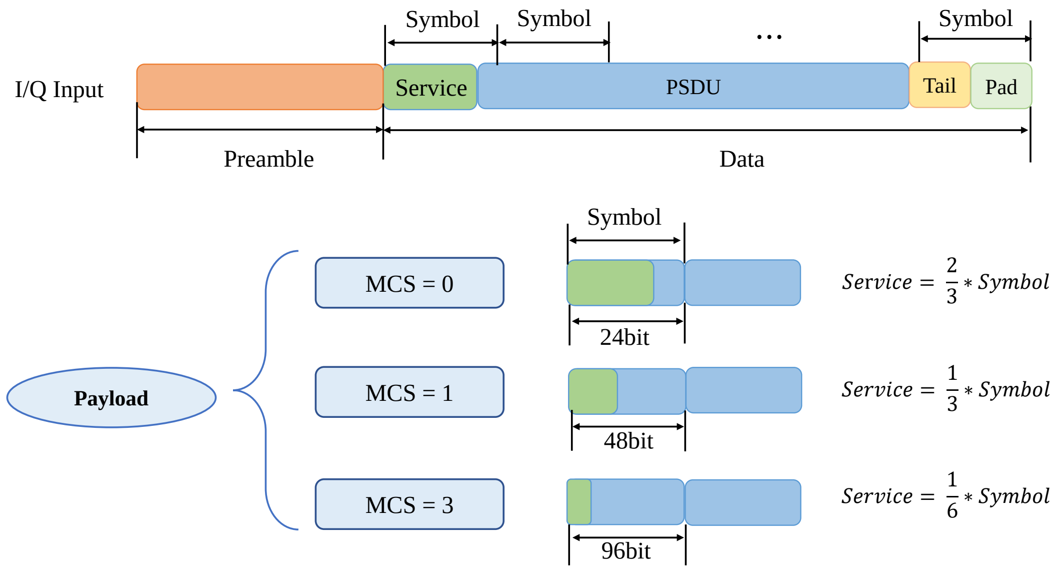


Figure 4.7: For different MCS, the proportion of symbols occupied by the service field is different, which makes the starting position of decoding change with MCS.

Difference Summation. This method is designed to make better use of the alternation features. First, subtract the previous bit from the next bit of the bitstream for differential calculation. Then, take the absolute value of each result obtained by the difference and accumulate them. Finally, divide the accumulated sum by the window size. The formal representation of this method is shown in Eq.4.12, where $\mathcal{V}$ represents the bitstream vector within a decoding window.

$$diff = \frac{sum(abs(\mathcal{V}(2:end) - \mathcal{V}(1:(end-1))))}{length(\mathcal{V})} \tag{4.12}$$

Now, we can already decode tag data under different MCS. Interestingly, when conducting experiments, a new phenomenon is discovered. To recover the data correctly, we need to find the starting position of a symbol as accurately as possible. In the deinterleaving level, the starting position is always an integer multiple of the number of bits contained in the symbol. However, for the payload level, not only is it not an integer multiple, but it will change with the change of MCS. After analysis, we find that the key is the service field.

Service is padded before PSDU in the padding operation. We modulate tag data in units of a symbol and the symbol division of the data part starts from the beginning of the data field including the service. However, when doing decode translation in the payload layer, the service field has been depadded. So, the starting position of demodulation will change according to the proportion of a symbol occupied by the service. The specific situation is

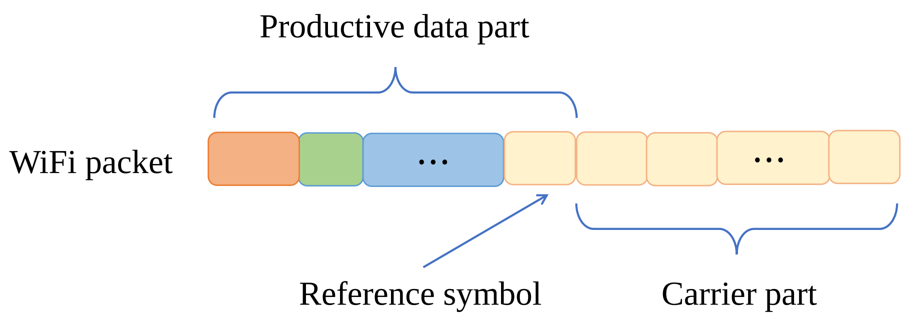


Figure 4.8: RapidRider+ design where the excitation contains two components: productive data part and carrier part that can be modulated by the tag.

also shown in Figure 4.7. When MCS = 0,1,3, service accounts for two-thirds, one-third and one-sixth of a symbol, respectively. Because the number of bits contained in a symbol is different under different MCSs. This phenomenon brings more difficulties to the decoding in the payload layer. Fortunately, this phenomenon does not exist in the deinterleaving layer, since this layer contains the service field, so the positions of modulation and demodulation are the same. Therefore, RapidRider is much more general than others relying on payload data.

4.2.4 Single-Receiver for Aggregated Transmissions

In real-world scenarios, the current codeword translation suffers another serious problem: it requires two receivers to decode tag data. There are at least three drawbacks to such a requirement. First, it needs extra synchronization overhead between two receivers for correct decoding, which is implicitly required by XOR operations. Second, it requires the data from the ambient signals intended for unknown commercial NICs and acquiring such data is not always available. Third, it takes two WiFi channels to transmit both productive and tag data, and thus spectrum efficiency is not fully optimized. Therefore, we design RapidRider+, a backscatter scheme that can transmit both productive and tag data on the same packet, and one receiver is adequate for decoding both.

As shown in Figure 4.8, the design is mainly about excitations. In this design, the payload of an excitation WiFi packet is composed of two parts. The first part is for transmitting productive data, which means the symbols are arbitrary, while the second part is for codeword translation, or more precisely, it is in another form of 'continuous waves' since all the symbols are the same. The last symbol of the first part is designed as the reference symbol, and the carrier part is full of reference symbols. Such a design is based on an insight: codeword translation's key is the reference for decoding. As long as the reference is known at

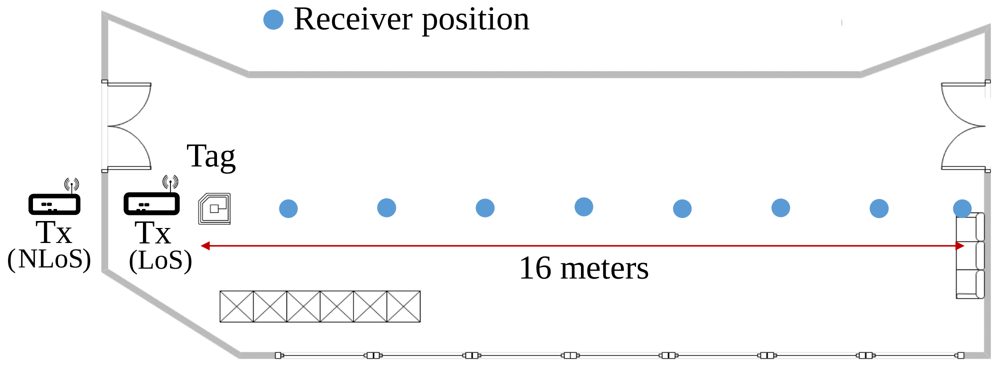


Figure 4.9: Experimental deployment.

the receiver, everything else, e.g., the previous requirement of the original payload, is not necessary. The RapidRider+ encoding and decoding processes are as follows.

Encoding. To embed sensor data, the tag first detects the excitation signals, and then jumps over the preamble fields, and the productive data part of the payload. Specifically, it means no phase shift but just frequency shift on those fields. Then starting from the carrier part, the tag employs phase-based codeword translation to put data onto the carrier. Note that our encoding is single-symbol in nature, while prior arts use multiple symbols.

Decoding. At a single receiver, the AP processes the received signal as a normal WiFi packet, first preamble then payload. During payload processing, the handling of the productive data part is still the same as normal WiFi. But starting from the carrier part, it decodes tag data by XORing the last symbol of the productive data part and every symbol in the carrier part. Finally, both productive and tag data are recovered when the packet ends.

From the above description, we can conclude there are three main advantages for RapidRider+. First, the processing is much simpler as there is no need to coordinate with two receivers anymore. Second, aggregated productive and tag transmissions are made possible, and various tradeoffs between two kinds of data can be achieved by adjusting the lengths of two parts. In particular, we define aggregated transmission coefficient as $\gamma = \frac{l_p}{l}$, where l_p is the length of the productive data part, and l is the total length of the payload. Hence, for different application demands, we can tweak γ to achieve balances between the two. Third, RapidRider+ currently only works with restricted excitations, which means the excitor has to generate carriers as in Figure 4.8. This arrangement incurs additional restrictions for WiFi packets and thus may affect the overall WiFi throughput. Yet, such restrictions are traded for requiring only a single receiver.

4.3 Implementation

We prototype RapidRider using FPGAs, commodity radios, USRPs and ZedBoards. The details are as follows.

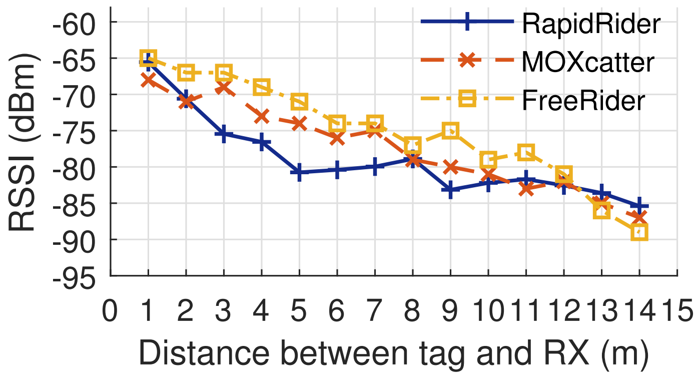
Backscatter tag. Our prototype is based on MOXcatter [89] and X-Tandem [90]. It mainly consists of a signal detector, an FPGA and an RF switch. The signal detector is constructed using an AD8313 and TLV3501. AD8313 is a multistage demodulation logarithmic amplifier and TLV3501 is a threshold voltage tuning circuit. The modulation and control logics are implemented by a XILINX Artix-7 XC7A35T. Also, we employ a mixed-mode clock manager (MMCM) to generate the same-frequency signals with different initial phases, which are used to control the RF-switch ADG902.

Ambient excitations. We use two types of excitations, commercial and SDR. The commercial one is a Dell laptop equipped with a Qualcomm Atheros AR938x as an ambient excitation. This Qualcomm NIC supports 802.11a/g/n protocol with 2 antennas. To facilitate the control of ambient signals, we use commercial software CommView [1], which provides access to various parameters, e.g., transmission rate and payload content. We also use ZedBoard with an AD9361 daughterboard as excitation to have an in-depth look at the intermediate processes.

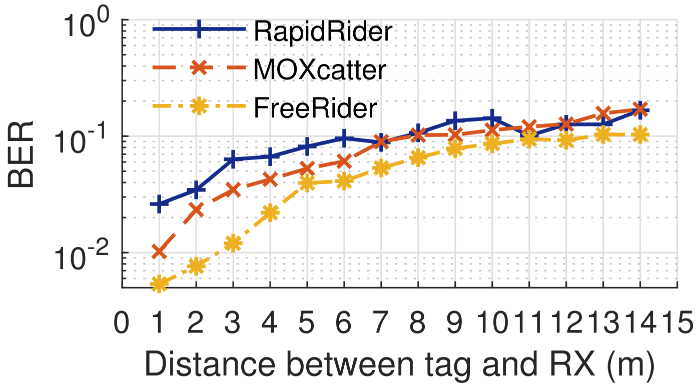
WiFi AP receiver. We have three types of receivers, commercial and two types of SDR. For commercial APs, we use the same setup as the ambient excitation. Meanwhile, a USRP N210 with an SBX40 daughter board and a USRP B210 are assembled as SDR APs, which have full access to the baseband signals. The USRPs are connected to an ASUS laptop running a modified open-source WiFi receiver *gr-ieee802-11* [19] on Ubuntu. When using ZedBoard SDR as excitation, we also use the same setup. Two ZedBoards are connected to the Dell laptops running the modified MATLAB sample program.

Deployment. We conduct comprehensive experiments in many different environments such as indoor and outdoor, line-of-sight (LoS) and non-line-of-sight (NLoS). As Figure 4.9 depicts, we first conduct indoor experiments on a second-floor platform of an office building. For LoS scenarios, we put the ambient excitation and backscatter tag together by the wall, and then gradually move the WiFi receiver away along the straight line. In NLoS situation, we move the ambient excitation and tag together to the other side of the wall, so that there is no direct path between WiFi receivers and the tag.

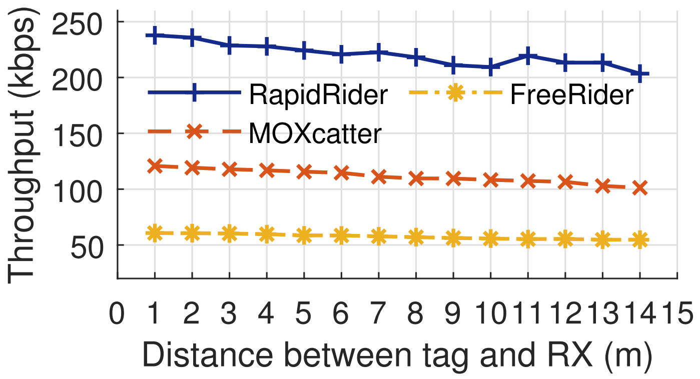
Competitions. We compare our design against two state-of-the-art systems, FreeRider [87] and MOXcatter [89]. FreeRider uses four OFDM symbols to encode one tag bit, and MOXcatter takes two symbols.



(a) RSSI



(b) BER

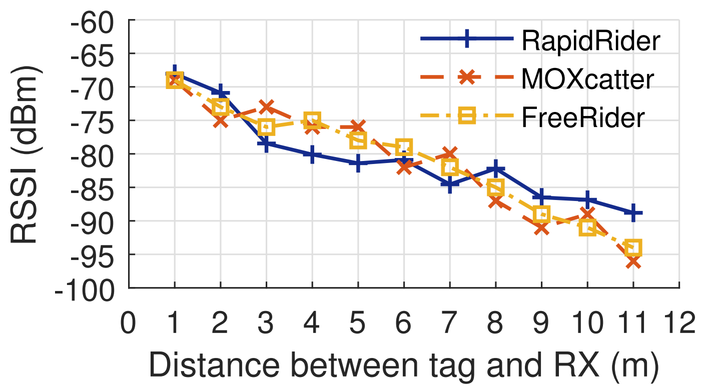


(c) Throughput

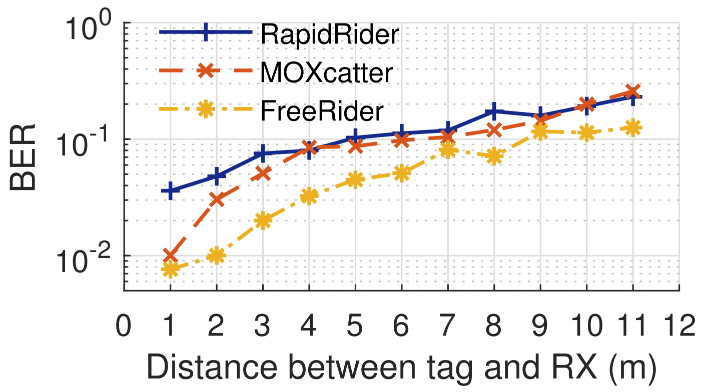
Figure 4.10: Backscatter RSSI, BER, and throughput across distances in LoS scenarios.

4.4 Evaluation

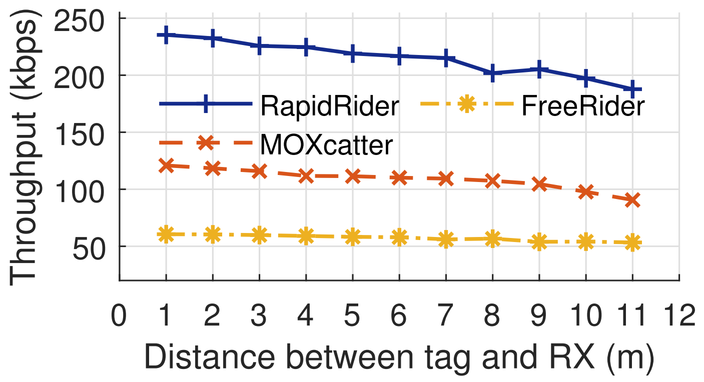
In this section, we fully evaluate RapidRider's performance and compare it against state-of-the-art works under various environments. We first measure the end-to-end performance and then conduct micro-benchmark experiments.



(a) RSSI



(b) BER



(c) Throughput

Figure 4.11: Backscatter RSSI, BER, and throughput across distances in NLoS scenarios.

4.4.1 End-to-End Performance

To evaluate the end-to-end performance, we measure RSSIs, BERs, and throughputs of backscattered signals for both indoor and outdoor environments.

Indoor experiments. For indoor environments, we conduct experiments in both LoS and NLoS scenarios. We deploy the backscatter tag 0.5 m away from the excitation. Then we move the receiver far away and measure the RSSI, BER, and throughput at different distances.

From Figure 4.10a and 4.11a, we can observe as the distance increases, the RSSI of the backscattered signal decreases rapidly, and the maximum communication range of RapidRider is 14 m for LoS and 11 m for NLoS. This is because the signal strength of backscattered signals under NLoS deployment is weaker than that with LoS. For the same distances, RSSIs of the backscattered signal in NLoS environments are 5 to 10 dBm lower than those in LoS scenarios.

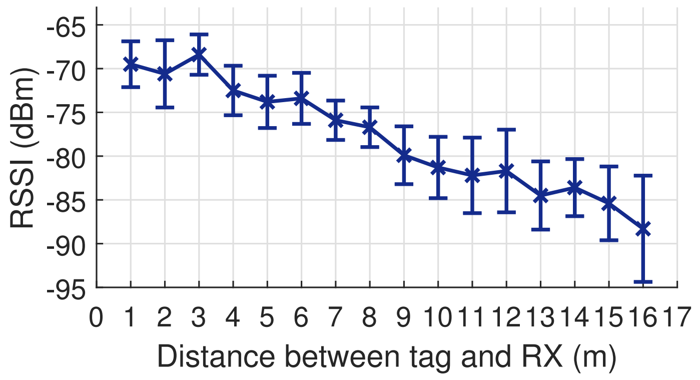
As shown in Figure 4.10b and 4.11b, BERs gradually increase with longer distances, and BERs in LoS are better than those in NLoS. When the distance between the receiver and tag is within 8 m, BERs for RapidRider in LoS environments are lower than 10%. However, in NLoS situations, when the receiver is just 5 m away, the BER rises to over 10%. Furthermore, in both LoS or NLoS scenarios, FreeRider's BERs are significantly lower than those of RapidRider and MOXcatter in all cases. It is mainly due to redundant coding.

Despite the advantages brought by redundant coding, FreeRider and MOXcatter are no match for RapidRider in terms of throughput. As depicted in Figure 4.10c and Figure 4.11c, the maximum throughputs of RapidRider are 237.8 kbps and 235.3 kbps in LoS and NLoS scenarios, respectively. In contrast, counterparts of MOXcatter and FreeRider are 120.8 kbps and 60.7 kbps for LoS cases. In other words, for maximum throughput, RapidRider is 1.97x and 3.92x better than MOXcatter and FreeRider. This is mainly attributed to that RapidRider is a single-symbol based system, while the other two are based on multiple symbols. The above results show that RapidRider significantly improves raw-data rates and achieves efficient OFDM backscatter.

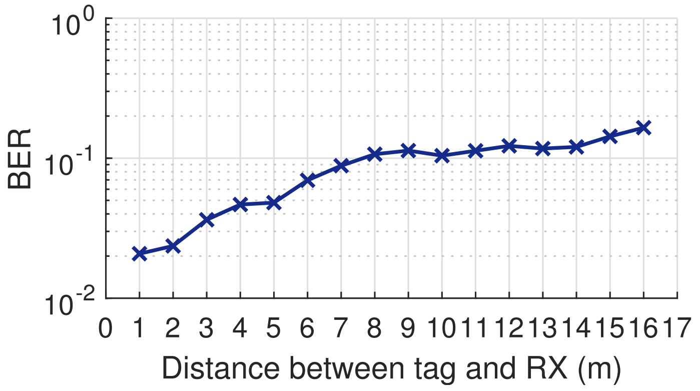
Outdoor experiments. In addition, we also examine the performance of RapidRider in outdoor environments. We conduct experiments in a parking lot on campus where there is no obstruction, i.e., all LoS cases. As shown in Figure 4.12a, we observe that due to the lack of obstacles for outdoors, RSSIs of the backscattered signals are significantly better than those of indoors. Accordingly, the maximum communication range expands to 16 m. Similarly, BERs also benefit from outdoors; even at the farthest distance - 16 m, the BER keeps under 20%. Better BERs and RSSIs lead to better throughput: the maximum throughput for outdoors is 239.1 kbps. When the tag and receiver are far away, the outdoor performance is much better than indoor performance. For instance, at a distance of 14 m, the throughput for outdoors reaches 214.7 kbps while the counterpart is 203.5 kbps for indoors.

4.4.2 Micro Benchmarks

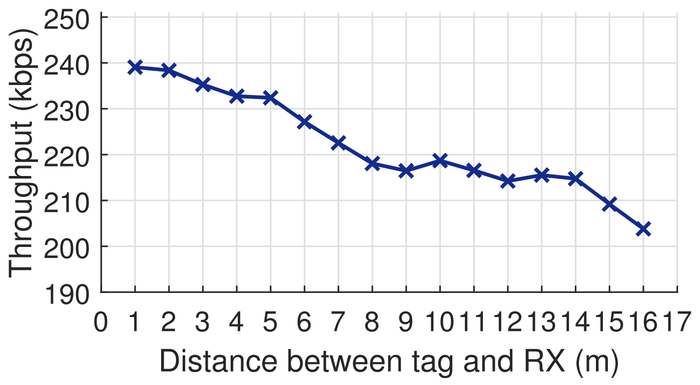
Next, we first measure the performance at different layers when using a different number of symbols for tag data modulation. Then, we evaluate RapidRider's single-symbol encoding performance and robustness to different excitation signals. Further, we compare and analyze



(a) RSSI



(b) BER



(c) Throughput

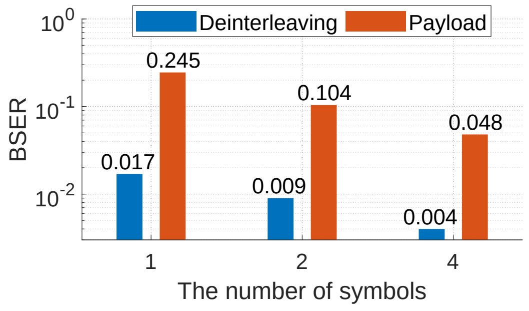
Figure 4.12: Backscatter RSSI, BER, and throughput across distances in outdoor deployment.

the translation methods and find the optimal threshold setting. Finally, we evaluate the system performance under different MCSs.

Impact of the number of symbols for modulation. We examine how symbol-wide and packet-wide operations behave when using different numbers of symbols and focus on

Symbol Layer BSER	1-symbol	2-symbol	4-symbol
Demodulation	0.43%	0.22%	0.11%
Deinterleaving	0.44%	0.22%	0.11%
Decoding	18.9%	6.7%	3.3%
Payload	25.2%	7.1%	3.5%

(a) BSER comparison for four layers by simulation experiments.



(b) BSER comparison for Deinterleaving and Payload level in real experiments.

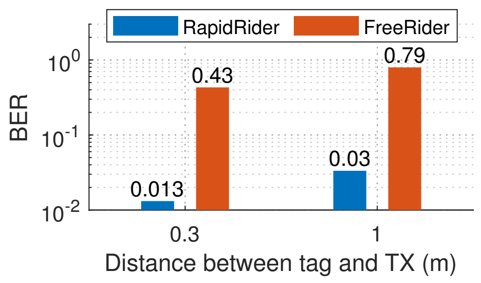
Figure 4.13: Perfomance comparison when using different numbers of symbols for tag data modulation.

the deinterleaving layer and the payload layer, which are used for tag data demodulation by RapidRider and FreeRider & MOXcatter, respectively. We fix the distance between the backscatter tag and receiver at 2 m and place the transmitter close to the tag. The number of symbols used to modulate a single tag data is set to 1, 2 and 4. We use the newly defined new metrics, BSER, to do evaluation. The results are shown in Figure 4.13.

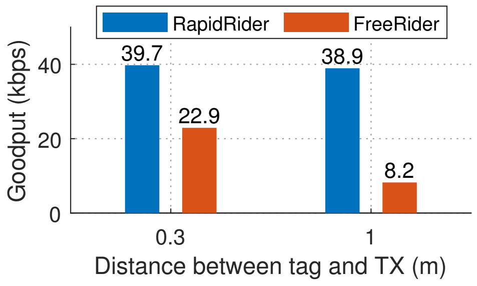
First, we conduct simulation experiments on MATLAB. From the operations of receiver, we select 4 layers' bitstreams, which are demodulation, deinterleaving, decoding and payload. The corresponding BSER calculation results are shown in Figure 4.13a. It's obvious that the BSER before the deinterleaving layer is much smaller than that after it since decoding and descrambling are packet-wide operations. For deinterleaving layer, as the number of symbols used for modulation decreases, the BSER only increases from 0.11% to 0.44%. In comparison, BSER increases from 3.3% to 25.2% at payload layer. In the case of 1-symbol, an interesting discovery is that there is a certain gap between the BSER of decoding and the payload layer, which is due to descrambling. The above results demonstrate the non-exchangeability of payload-wide operations and $\mathcal{C}$. Further, we conduct a real experiment to measure the BSER of the payload and deinterleaving layers. The results in Figure 4.13b validate the findings of the simulation experiments. The value of BSER for the payload layer is much higher than deinterleaving, which explains why FreeRider and MOXcatter can't achieve single-symbol modulation.

Impact of single symbol modulation. We examine how RapidRider and FreeRider behave for single-symbol modulation [2]. Besides throughput, goodput is another standard metric for evaluating communication and networking systems. Goodput is the application-layer throughput, which is affected by more factors than physical-layer throughput. For

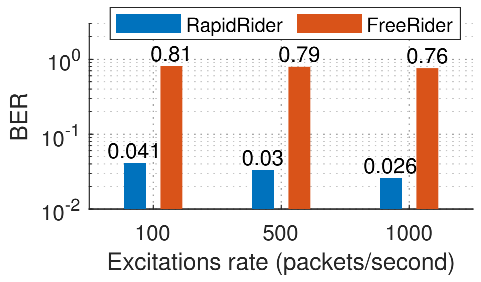
[2]MOXcatter is not included here because single-symbol MOXcatter is equivalent to single-symbol FreeRider.



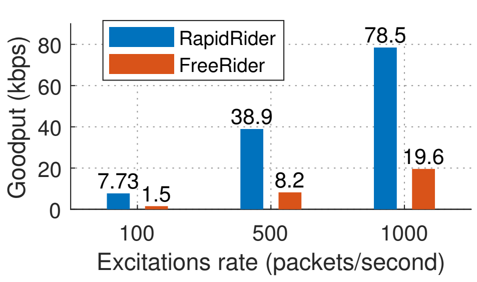
(a) BER comparison in different distance of TX to tag.



(b) Goodput comparison in different distance of TX to tag.



(c) BER comparison in different excitations rate.



(d) Goodput comparison in different excitations rate.

Figure 4.14: Performance comparison for RapidRider and single-symbol FreeRider.

example, goodput highly depends on excitation rates for backscatter systems. We fix the distance between the backscatter tag and receiver at 1 m and put the transmitter at 0.3 m and 1 m away from the tag. The excitation rates are set at 100 pkt/s, 500 pkt/s, and 1000 pkt/s, where each packet has 420 bytes, i.e., 150 OFDM symbols. The results are shown in Figure 4.14.

We first investigate the impact of distance on backscatter communication and fix the excitation rate at 500 pkt/s. As Figure 4.14a depicts, when the distance between the tag and transmitter is only 0.3 m, the BER of RapidRider is only 1.3%, while the BER of single-symbol FreeRider is 43%. When the tag is 1 m away from TX, the BER of RapidRider slightly increases to 3% while FreeRider's BER becomes 79%. Such high BERs for FreeRider indicate that single-symbol FreeRider does not work for backscatter communication. By contrast, RapidRider's BERs stay within reasonable ranges thanks to the deinterleaving-twins strategy.

Then, we check the impact of different excitation rates. According to Figure 4.14c and Figure 4.14d, at various excitation rates, BERs of RapidRider are 4%, 3%, and 2.6%, which means excitation rates do not affect BERs much. In addition, as the excitation rate increases, the goodput of RapidRider grows accordingly. In particular, when the excitation rate increases from 100 pkt/s to 1000 pkt/s, the corresponding goodput improves from 7.7

kbps to 78.5 kbps. Therefore, we learn that the excitation rate is one of the dominating factors for goodput.

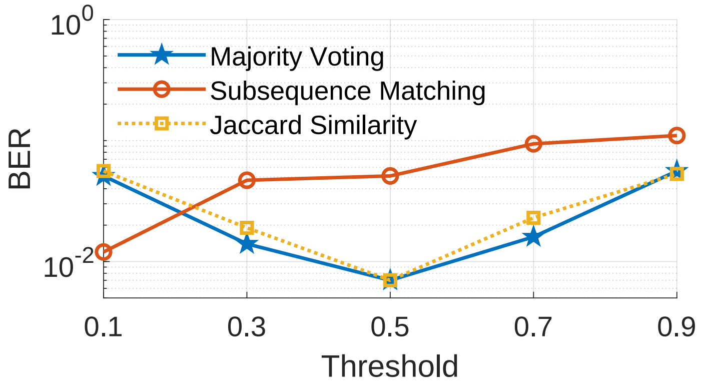
Impact of translation methods. We measure the different translation methods for RapidRider and FreeRider using single-symbol modulation. Moreover, we explore the optimal threshold settings for these methods. The excitation transmitter is placed close to the backscatter tag and the receiver is 2 m away from the tag. To eliminate the sensitivity of the decoding method to the tag data and make the results more reliable, we use three groups of random tag data in the experiment and take the averaged results. For Majority Voting and Subsequence Matching, we also average the results of two subcategories, which take '1' or '0' as the translation core, respectively.

First, we compare translation methods applicable to the case where the sequence pattern is consecutive '1'. The range of the threshold is set from 0.1 to 0.9 with a step of 0.2. Figure 4.15a and 4.15b shows the results for RapidRider and FreeRider. For RapidRider, Majority Voting and Jaccard Similarity are significantly better than Subsequence Matching in most cases. However, for single-modulation FreeRider, BER is much higher and there is no obvious difference between several methods, which is mainly due to the difficulty of implementing single-symbol modulation at the payload layer. Therefore, we focus on Figure 4.15a to find the optimal threshold setting. The optimal value of Majority Voting and Jaccard Similarity is around 0.5. For Subsequence Matching, the BER decreases as the threshold decreases within the current setting range. To find the optimal value, we carry out further experiments that reduce the value of the threshold. The result is that if the threshold is too small, BER will also increase, and find that the optimal threshold is around 0.1. Therefore, we learn that translation methods have a great influence on the BER and choosing the appropriate translation method is extremely important.

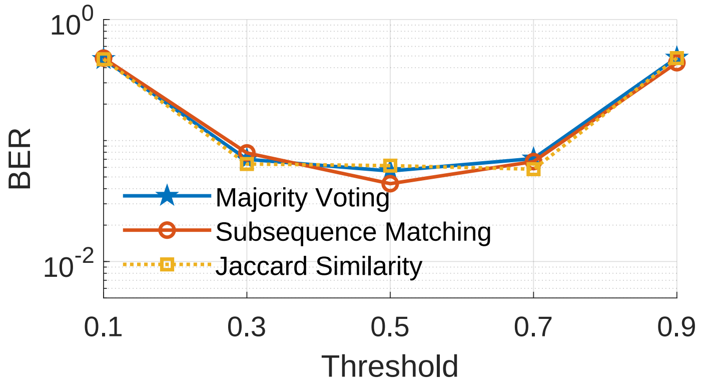
Then, we evaluate the two methods for the alternating sequence of '1' and '0' for RapidRider. The range of the threshold is set from 0.375 to 0.525 with a step of 0.25. According to Figure 4.15c, 0.425 is the best choice for Majority Voting and 0.5 for Difference Summation. In addition, we find that in the optimal case, the BERs of the two methods are similar. Therefore, both methods can be used for tag data translation. From the above results, it can be seen that Majority Voting is the most general method. It can be applied to many scenarios as long as the threshold is set appropriately.

Impact of MCS. We examine how RapidRider and single-symbol FreeRider behave at different MCSs. We fix the distance backscatter tag and receiver at 2 m and put the transmitter close to the tag. We use the Majority Voting method for comparison and set the threshold to the optimal value.

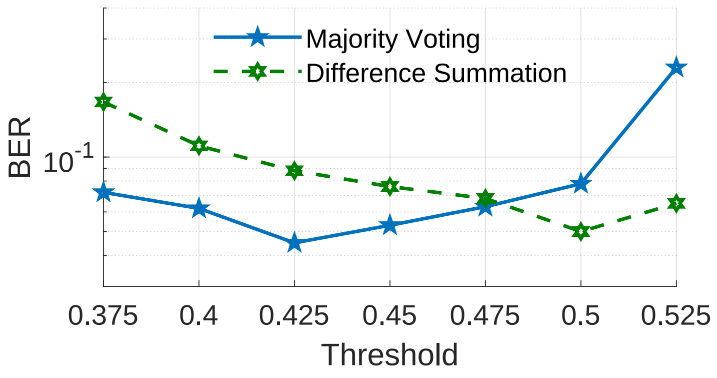
As Figure 4.16 depicts, the performance of RapidRider is much better than FreeRider no matter what MCS is. When MCS = 0, the BER of FreeRider is 5.6%, while RapidRider is only 0.7%. Further, when MCS increases to 3, the BER of FreeRider increases to 20.4% rapidly, which is unacceptable for data transmission. But the BER of RapidRder is still only



(a) BER comparison for RapidRider.



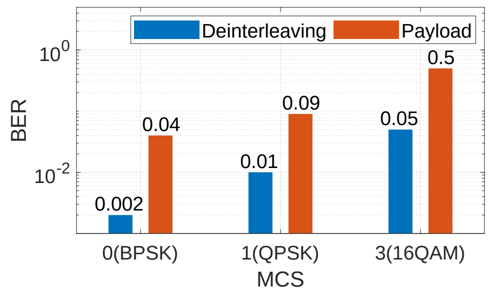
(b) BER comparison for FreeRider.



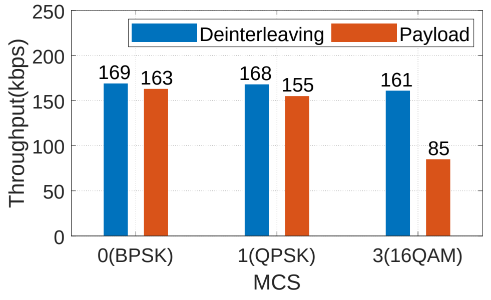
(c) BER comparison for RapidRider.

Figure 4.15: Performance comparison for four translation methods: Majority Voting, Subsequence Matching, Jaccard Similarity and Difference summation, which can be used in different cases. (a) and (b) shows the comparison of the first three and (c) shows the result of Majority Voting and Difference Summation.

around 4.5%. Figure 4.16 tells us that RapidRider trades around 4kbps tag data throughput for a 4x increase in carrier data. By comparison, FreeRider costs over 30kbps, which is 7.5x

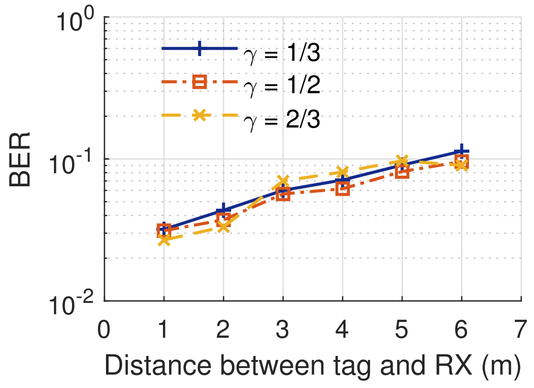


(a) BER comparison for RapidRider and FreeRider.

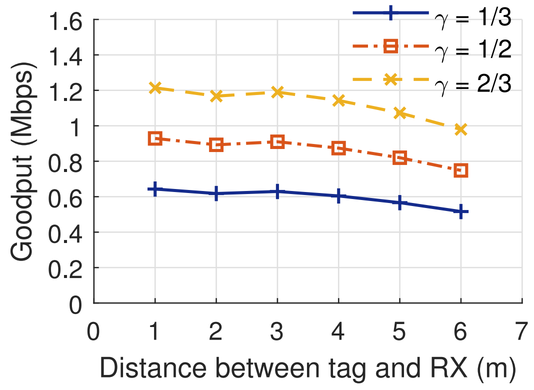


(b) Throughput comparison for RapidRider and FreeRider.

Figure 4.16: Performance comparison for different MCS.



(a) BER of RapidRider+.



(b) Aggregate goodput of RapidRider+.

Figure 4.17: Performance comparison of three typical modes for RapidRider+.

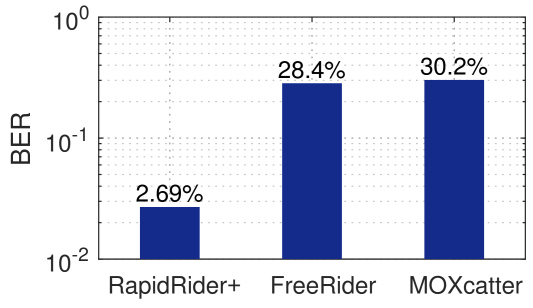
more than RapidRider. Again, the universality of our method is demonstrated. RapidRider can make full use of uncontrolled ambient wifi signals in the environment.

4.4.3 RapidRider+

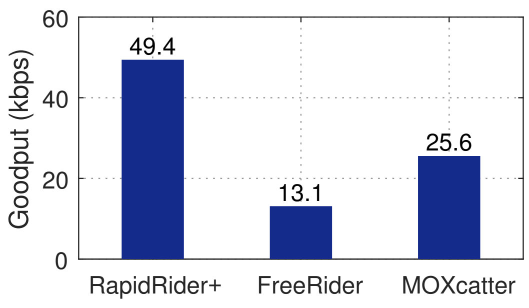
Next, we evaluate RapidRider+ using a single receiver.

Impact of aggregated transmission coefficient. We fix the distance between excitation transmitter and backscatter tag at 0.3 m, and the receiver is placed 1 m away from the tag. The excitation rate is 1000 pkt/s. We choose three typical values (1/3, 1/2, 2/3) for γ. The empirical results are shown in Figure 4.17. According to Figure 4.17a, BERs are very stable across different γ settings.

Regarding goodput, as shown in Figure 4.17b, when γ is 1/3, the aggregate goodput is 698.6 kbps, where the productive goodput is 609.4 kbps and tag goodput is 49.2 kbps. When $\gamma = 1/2$ and $\gamma = 2/3$, the aggregate goodputs are 950.9 kbps and 1,243.5 kbps, respectively.



(a) BER comparison with obstacles.



(b) Goodput comparison with obstacles.

Figure 4.18: Comparison of BER and tag data throughputs with FreeRider and MOXcatter when there are obstacles in the original channel but no obstacles in the backscatter channel.

These show that RapidRider+ can achieve decent goodputs for both productive and tag data transmissions. Also, tradeoffs can be made by simply choosing a proper γ.

Impact of obstacles. In addition to being able to transmit productive data, supporting a single receiver is another advantage of RapidRider+. Prior works require two receivers with one working on the original channel and the other receiving backscattered packets. Such design has an implicit assumption: the excitation signal should be correctly received and decoded. Unfortunately, it is not always the case. To investigate the impact of the original channel quality on tag data decoding, we conduct experiments with obstacles in the way from the excitation to the AP. In particular, the tag is placed 0.5 m away from the transmitter and receiver. The original channel is blocked by metal obstacles, while the backscatter channel has LoS. The excitation rate is 1000 pkt/s, and γ is 1/3. Since FreeRider and MOXcatter cannot transmit productive and tag data at the same time, we only compare the BER and goodput for tag data transmissions.

Figure 4.18a shows the results of BERs for RapidRider+, FreeRider, and MOXcatter. Obviously, the metal obstacles negatively impact the quality of the original channel and further degrade the decoding of tag data. Specifically, BERs of FreeRider and MOXcatter increase significantly to 28.4% and 30.2%, while RapidRider+'s BER is only 2.69%. This is mainly due to that RapidRider+ only requires one receiver whose channel has clear LoS, while FreeRider and MOXcatter have the original channel blocked. Moreover, RapidRider+ achieves better tag-data goodput than both FreeRider and MOXcatter, as shown in Figure 4.18b. Even the modulatable length of RapidRider+ is only 2/3 of FreeRider and MOXcatter, the goodput of RapidRider+ reaches 49.4 kbps. On the contrary, FreeRider and MOXcatter can only achieve 13.1kbps and 25.6kbps goodputs, respectively. Similar to BER comparison, the main factor is that the low-quality data from the original channel seriously affects tag data decoding. What is worse, if the original data packet is completely lost, no tag data can be deduced even with perfect backscatter channels. Hence, RapidRider+'s single-receiver solution is more robust in mobile wireless environments.

4.5 Discussion

Excitation constraints. In RapidRider+ design, to accommodate single-receiver scenarios, we have imposed some restrictions on excitations. So far, there is no available theory or experiment that could bring up single-receiver decoding with random excitations. One possible solution is to decode data by brute-force. Even though the complexity is super high, pruning heuristic algorithms based on statistics may help when high-performance supercomputing is accessible.

Decoding on other levels. According to our derivation in Section III.B, in fact, the successful decoding can be made on any level between FFT and deinterleaving. Hence, the choice of decoding level should meet real-world applications. Our choice of the deinterleaving level here is based on that we want the computation overhead of deinterleaving twins (one forward and one backward) is balanced.

Higher-order modulation. Higher backscatter throughput requires higher-order modulation. To enable higher-order modulation, e.g., 16-QAM, 64-QAM, the current codeword translation scheme has to be revamped. At the same time, the circuits of QAM modulation for backscatter need to be designed, which is different from traditional RF front-end using IQ mixers.

4.6 Conclusion

We have presented RapidRider, the first WiFi backscatter system that achieves 237.8 kbps throughput with uncontrolled ambient OFDM signals. The key enabler is the single-symbol decoding scheme that uses deinterleaving twins with a forward deinterleaver and a backward one. The design of this decoding scheme is based on our insight that not all OFDM WiFi operations are commutative with codeword translation. We also explore the MCS for excitation and discuss several translation methods, finding the optimal threshold setting for different methods. We prototype our design using off-the-shelf devices and customized tags. Extensive field studies show that RapidRider achieves up to 3.92x and 1.97x throughput gains over FreeRider and MOXcatter. We believe that RapidRider will benefit a range of backscatter applications as it provides a low-power, high-throughput backscatter communication by reusing existing ambient signals.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

This thesis explores fundamental factors that hinder the development of robust general-purpose backscatter with commodity radios. We analyze the root cause of these issues, and propose some solutions, including link layer design, commercial-compatible modulation techniques, and decoding methods, to significantly improve the compatibility with commodity radios, thus making backscatter communication into general-purpose battery-free data transfer for IoTs.

We present a Mobility-aware Rate Adaptation (MobiRate) method that uses Physical (PHY) information for backscatter networks. Our goal is to achieve high throughput while remaining compatible with standard and commercial RFID readers. We have observed that current backscatter networks that aim to provide a high-throughput interconnected platform are not adequately prepared. They are not fully compatible with ambient signals, commodity devices, and protocol stacks. Our aim is to deliver high throughput for mobile backscatter networks in a standard-compliant manner. Specifically, we focus on optimizing rate adaptation at the link layer for the widely used C1G2 standards [5]. Our design eliminates hardware dependency by introducing a throughput-based rate adaptation framework and extensively leveraging fine-grained mobility hints to optimize probing processes. We compare MobiRate with two state-of-the-art solutions, Blink and CARA, in a wide range of settings. Through extensive experiments, we demonstrate that MobiRate can first detect self-interference accurately, with an accuracy of over 90%, and then shift probing direction for better rate selection, resulting in an average of 1.21x throughput gains. We tested MobiRate with different scenarios, such as classrooms, lounges, and various velocities, under self-interference. Our results show that MobiRate can provide stable and effective countermeasures for self-interference detection and achieve better rate selection for higher throughput.

We have proposed a reliable BLE backscatter modulation design that works with a single commodity receiver. Providing a backscatter solution that is completely built by commercial radios is a common goal. Although the most recent work, FreeRider [87], realized this goal with only commodity BLE radios, it suffers from key reliability issues. FreeRider employs a two-step modulation to be compatible with BLE signals, which inevitably introduces unreliability due to self-interference or no solid signal in the target frequency. Our proposed design, RBLE, modulates the sensor data of the RBLE tag on BLE exciting signals and backscatters new BLE packets that any commodity BLE device can decode. The main contributions of RBLE include designing direct frequency shift modulation to enable bit-by-bit packet regeneration and leveraging dynamic channel reconfiguration to bypass interfered channels. RBLE achieves more than 17.3x and up to 78.8x goodput gains in Line-of-Sight (LoS) cases, and more than 17.1x and up to 66x goodput gains in Non-Line-of-Sight (NLoS) scenarios, compared to FreeRider. By the help of dynamic channel configuration, RBLE with channel hopping achieves 1.92x goodput gain over the one without such help in the presence of strong WiFi interference. We also implemented RBLE with only off-the-shelf phones, including iPhones and Android phones, and the experiments showed that RBLE is able to work with both data and advertising packets from smartphones as carriers. The maximal uplink range is 16 m for an iPhone as the receiver. In summary, our RBLE design is a reliable and efficient BLE backscatter modulation design that achieves high goodput gains in both LoS and NLoS scenarios, while being compatible with commodity BLE devices. It also leverages dynamic channel reconfiguration and is able to work with off-the-shelf smartphones as carriers.

In the last, we introduce RapidRider, a new WiFi backscatter design that utilizes uncontrolled OFDM WiFi signals as excitations and efficiently embeds tag data at the single-symbol rate. While backscatter communication with ambient signals is already achieved, it requires a controlled excitation signal. Some packet-level WiFi solutions have been proposed to work with uncontrolled ambient signals, but they suffer from low data rates (typically 1 kbps) since a packet can only carry one bit and are susceptible to environmental changes. We first explain why previous systems fail in single-symbol backscatter and then present our solution based on deinterleaved data. We also design a deinterleaving-twins decoding scheme, incorporating a forward deinterleaver and a backward deinterleaver, to enable the use of uncontrolled excitations. Additionally, we explore how the modulation and coding scheme (MCS) of the excitation affects tag demodulation, and design several translation methods for the first time. To break the two-receiver limit, we introduce a novel aggregated transmission mechanism that allows productive data and tag data to be sent on the same packet. The key insight is to use the pilot symbol as the reference symbol for tag data demodulation, making a single receiver sufficient for decoding both data streams. For single-symbol modulation, the BER of RapidRider could be as low as 1.3%, while that of FreeRider is 43% under the same setting, demonstrating that RapidRider is state-of-the-art for single-symbol OFDM backscatter. Thanks to RapidRider, for the first time, we can use the abundant uncontrolled WiFi signals around us freely for backscatter.

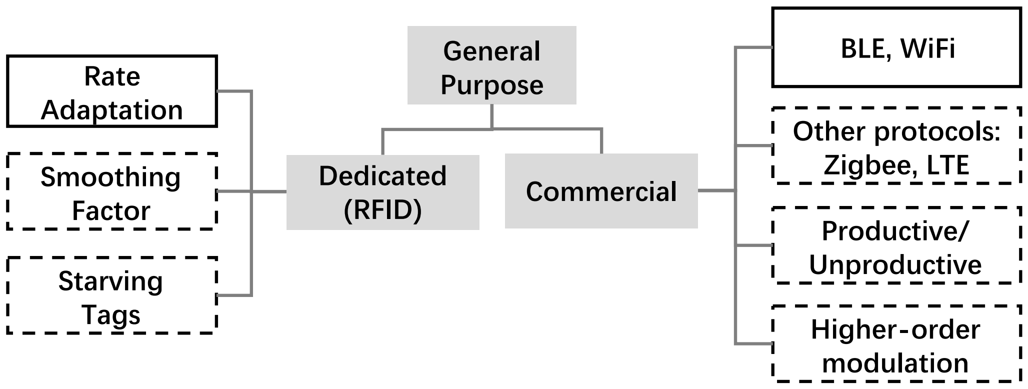


Figure 5.1: An overview of future work.

In summary, the proposed designs for backscatter communication, including MobiRate, RBLE, and RapidRider, aim to overcome the limitations of existing solutions by leveraging ambient signals for communication and achieving high-throughput and reliability with commodity radios. The proposed designs show promising results in various scenarios and can potentially enable new applications for backscatter communication in the future.

5.2 Future Work

5.2.1 Commercial

Productive and unproductive exciting signals. Choosing between an unproductive single tone or a productive data-carrying signal as the carrier involves trade-offs. For instance, FreeRider relies on the exciting signals' data sequence to decode the tag data. If the original channel's data sequence is corrupted, it becomes challenging to decode tag data, even if the backscattered channel's data is error-free. This productive-data dependence can significantly affect the bit error rate (BER) of the tag data when the original channel's quality becomes unstable. However, backscattering with productive exciting signals represents a significant step towards exploiting rich ambient signals, as it does not require controlling the exciting signal's payload content.

We have addressed the WiFi backscatter case by enabling the use of productive data-carrying signals as the carrier. Nevertheless, we are still seeking a compromise solution for BLE radios that can leverage productive exciting signals while ensuring the backscatter transmission's reliability. One possible approach is to properly manage the exciting signals' data content, maintaining a certain level of productive data goodput.

Choice of codeword scheme. The RBLE employs a codeword scheme that relies on a repetition code with multiple code rates. However, the error correction capability of such a code is limited. To improve it, we can opt for more complex codeword schemes, such as BCH codes, which offer higher code rates and stronger error correction capabilities. By using different code rates of BCH codes, we could achieve adaptive encoding.

Decoding on other levels. Based on our derivation in Section 4.2.2, successful decoding can be performed at any level between FFT and deinterleaving. Therefore, the selection of the decoding level should be based on the practical applications. Our choice of the deinterleaving level here is based on that we want the computation overhead of deinterleaving twins (one forward and one backward) is balanced.

Higher-order modulation. Achieving a higher backscatter throughput necessitates the adoption of higher-order modulation schemes, such as 16-QAM and 64-QAM. However, to enable such modulation schemes, the current codeword translation technique needs to be restructured. Additionally, the circuits of QAM modulation for backscatter need to be designed, which is different from traditional RF front-end using IQ mixers.

5.2.2 Dedicated

Starving Tags. Currently, our design has a potential issue of starving tags, meaning that some tags might not get interrogated. To address this problem, we can implement a throughput-based access protocol that ensures each participant attains equal throughput. In this type of design, the rate in the link layer is usually fixed (e.g., FM0) because maintaining the same throughput for multiple rates is a difficult task. However, in multi-rate scenarios, we can attempt to provide every node with equal access time rather than equal throughput. We aim to enhance the existing link layer design in the future to prevent any starving tags.

Smoothing Factor. The smoothing parameter is theoretically influenced by two primary factors: mobility and environment. In MobiRate, velocity is mainly utilized to indicate mobility, but future research may explore acceleration and orientation as well. On the other hand, modeling the environment entails considering RF propagation properties, frequency fading, polarization, and other factors.

We guess that deep reinforced learning could be a promising approach to optimizing the smoothing factor. By gathering more data traces and examining the interaction between η and actual rate feedback, we can deduce the appropriate value for η dynamically based on different environments and mobility status. Nonetheless, this technique may result in unwanted processing delays, so finding ways to make it more lightweight is a worthwhile area for further investigation.

Bibliography

[1] Commview for wifi. `http://www.tamos.com/products/commwifi//`.

[2] IEEE 802.11n-2009 standard, 2009. `http://standards.ieee.org/getieee802/download/802.11n-2009.pdf`.

[3] ATH9k. `https://wireless.wiki.kernel.org/en/users/drivers/ath9k`, 2017.

[4] BLE API on Windows. `https://docs.microsoft.com/en-us/windows/uwp/devices-sensors/bluetooth-low-energy-overview`, 2017.

[5] EPC C1G2 Standard. `http://www.gs1.org/epcrfid/epc-rfid-uhf-air-interface-protocol/2-0-1`, 2017.

[6] FreeRider implementation. `https://github.com/pengyuzhang/FreeRider`, 2017.

[7] Low Level Reader Protocol. `http://www.gs1.org/epcrfid/epc-rfid-llrp/1-1-0`, 2017.

[8] Minstrel. `https://wireless.wiki.kernel.org/en/developers/documentation/mac80211/ratecontrol/minstrel`, 2017.

[9] BLE API on Android. `https://developer.android.com/guide/topics/connectivity/bluetooth-le`, 2020.

[10] BLE API on macOS. `https://developer.apple.com/bluetooth`, 2020.

[11] BLE API on Ubuntu. `https://core.docs.ubuntu.com/en/stacks/bluetooth/bluez/docs/reference/dbus-api`, 2020.

[12] BLE Scanner. `https://apps.apple.com/us/app/ble-scanner-4-0/id1221763603`, 2020.

[13] LightBlue. `https://apps.apple.com/us/app/lightblue/id557428110`, 2020.

[14] nRF Connect. `https://apps.apple.com/us/app/nrf-connect/id1054362403`, 2020.

[15] OptiTrack. `https://www.optitrack.com`, 2022.

[16] Ali Abedi, Farzan Dehbashi, Mohammad Hossein Mazaheri, Omid Salehi-Abari, and Tim Brecht. Witag: Seamless wifi backscatter communication. *Proc. of ACM SIGCOMM*, 2020.

[17] Dinesh Bharadia, Kiran Raj Joshi, Manikanta Kotaru, and Sachin Katti. Backfi: High throughput wifi backscatter. In *Proc. of ACM SIGCOMM*, 2015.

[18] John Charles Bicket. *Bit-rate selection in wireless networks.* PhD thesis, Massachusetts Institute of Technology, 2005.

[19] Bastian Bloessl, Michele Segata, Christoph Sommer, and Falko Dressler. An ieee 802.11 a/g/p ofdm receiver for gnu radio. In *Proceedings of the second workshop on Software radio implementation forum.* ACM, 2013.

[20] Michael Buettner and David Wetherall. An empirical study of UHF RFID performance. In *Proc. of ACM MobiCom*, 2008.

[21] Zicheng Chi, Xin Liu, Wei Wang, Yao Yao, and Ting Zhu. Leveraging ambient lte traffic for ubiquitous passive communication. In *Proc. of ACM SIGCOMM*, 2020.

[22] Hsun-Wei Cho and Kang G Shin. Bluefi: bluetooth over wifi. In *Proc. of ACM SIGCOMM*, 2021.

[23] Manideep Dunna, Miao Meng, Po-Han Wang, Chi Zhang, Patrick P Mercier, and Dinesh Bharadia. Syncscatter: Enabling wifi like synchronization and range for wifi backscatter communication. In *Proc. of USENIX NSDI*, 2021.

[24] Joshua F Ensworth and Matthew S Reynolds. Every smart phone is a backscatter reader: Modulated backscatter compatibility with bluetooth 4.0 low energy (ble) devices. In *Proc. of IEEE RFID*, 2015.

[25] Joshua F Ensworth and Matthew S Reynolds. Ble-backscatter: Ultralow-power iot nodes compatible with bluetooth 4.0 low energy (ble) smartphones and tablets. *IEEE Transactions on Microwave Theory and Techniques*, 65(9):3360–3368, 2017.

[26] Xiaoyi Fan, Wei Gong, and Jiangchuan Liu. Tagfree activity identification with rfids. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2(1):1–23, 2018.

[27] Reza Ghaffarivardavagh, Sayed Saad Afzal, Osvy Rodriguez, and Fadel Adib. Ultra-wideband underwater backscatter via piezoelectric metamaterials. In *Proc. of ACM SIGCOMM*, 2020.

[28] Reza Ghaffarivardavagh, Sayed Saad Afzal, Osvy Rodriguez, and Fadel Adib. Underwater backscatter localization: Toward a battery-free underwater gps. In *Proc. of ACM HotNets*, 2020.

[29] Richard M Goldstein, Howard A Zebker, and Charles L Werner. Satellite radar interferometry: Two-dimensional phase unwrapping. *Radio science*, 23(4):713–720, 1988.

[30] Wei Gong, Si Chen, Jiangchuan Liu, and Zhi Wang. Mobirate: Mobility-aware rate adaptation using phy information for backscatter networks. In *Proc. of IEEE INFOCOM*, 2018.

[31] Wei Gong, Haoxiang Liu, Lei Chen, Kebin Liu, and Yunhao Liu. Fast composite counting in rfid systems. *IEEE/ACM Transactions on Networking*, 24(5):2756–2767, 2015.

[32] Wei Gong, Haoxiang Liu, Jiangchuan Liu, Xiaoyi Fan, Kebin Liu, Qiang Ma, and Xiaoyu Ji. Channel-aware rate adaptation for backscatter networks. *IEEE/ACM Transactions on Networking*, 26(2):751–764, 2018.

[33] Wei Gong, Haoxiang Liu, Kebin Liu, Qiang Ma, and Yunhao Liu. Exploiting Channel Diversity for Rate Adaptation in Backscatter Communication Networks. In *Proc. of IEEE INFOCOM*, 2016.

[34] Wei Gong and Jiangchuan Liu. Roarray: Towards more robust indoor localization using sparse recovery with commodity wifi. *IEEE Transactions on Mobile Computing*, 18(6):1380–1392, 2018.

[35] Wei Gong, Jiangchuan Liu, and Zhe Yang. Fast and reliable unknown tag detection in large-scale rfid systems. In *Proc. of ACM MOBIHOC*, 2016.

[36] Wei Gong, Kebin Liu, and Yunhao Liu. Directional diagnosis for wireless sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, 26(5):1290–1300, 2014.

[37] Wei Gong, Longzhi Yuan, Qiwei Wang, and Jia Zhao. Multiprotocol backscatter for personal iot sensors. In *Proc. of ACM CoNEXT*, 2020.

[38] Joshua D Griffin and Gregory D Durgin. Complete link budgets for backscatter-radio and rfid systems. *IEEE Antennas and Propagation Magazine*, 51(2):11–25, 2009.

[39] Jeremy Gummeson, Pengyu Zhang, and Deepak Ganesan. Flit: a bulk transmission protocol for RFID-scale sensors. In *Proc. of ACM MobiSys*, 2012.

[40] Xiuzhen Guo, Longfei Shangguan, Yuan He, Nan Jing, Jiacheng Zhang, Haotian Jiang, and Yunhao Liu. Saiyan: Design and implementation of a low-power demodulator for LoRa backscatter systems. In *Proc. of USENIX NSDI*, 2022.

[41] Daniel Halperin, Wenjun Hu, Anmol Sheth, and David Wetherall. Predictable 802.11 packet delivery from wireless channel measurements. In *Proc. of ACM SIGCOMM*, 2010.

[42] Mehrdad Hessar, Ali Najafi, and Shyamnath Gollakota. Netscatter: Enabling large-scale backscatter networks. In *Proc. of USENIX NSDI*, 2019.

[43] Pan Hu, Pengyu Zhang, and Deepak Ganesan. Laissez-faire: Fully AsymmetricBackscatter Communication. In *Proc. of ACM SIGCOMM*, 2015.

[44] Pan Hu, Pengyu Zhang, Mohammad Rostami, and Deepak Ganesan. Braidio: An integrated active-passive radio for mobile devices with asymmetric energy budgets. In *Proc. of ACM SIGCOMM*, 2016.

[45] Zhanxiang Huang and Wei Gong. Eascatter: Excitor-aware bluetooth backscatter. In *Proc. of IEEE IWQoS*, 2022.

[46] Vikram Iyer, Vamsi Talla, Bryce Kellogg, Shyamnath Gollakota, and Joshua Smith. Inter-technology backscatter: Towards internet connectivity for implanted devices. In *Proc. of ACM SIGCOMM*, 2016.

[47] Mohamad Katanbaf, Vivek Jain, and Joshua R Smith. Relacks: Reliable backscatter communication in indoor environments. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 4(2):1–24, 2020.

[48] Bryce Kellogg, Aaron Parks, Shyamnath Gollakota, Joshua R Smith, and David Wetherall. Wi-fi backscatter: Internet connectivity for rf-powered devices. In *Proc. of ACM SIGCOMM*, 2014.

[49] Bryce Kellogg, Vamsi Talla, Shyamnath Gollakota, and Joshua R Smith. Passive wi-fi: Bringing low power to wi-fi transmissions. In *Proc. of USENIX NSDI*, 2016.

[50] Songfan Li, Hui Zheng, Chong Zhang, Yihang Song, Shen Yang, Minghua Chen, Li Lu, and Mo Li. Passive DSSS: Empowering the downlink communication for backscatter systems. In *Proc. of USENIX NSDI*, 2022.

[51] Haoxiang Liu, Wei Gong, Xin Miao, Kebin Liu, and Wenbo He. Towards adaptive continuous scanning in large-scale rfid systems. In *Proc. of IEEE INFOCOM*, 2014.

[52] Kebin Liu, Qiang Ma, Wei Gong, Xin Miao, and Yunhao Liu. Self-diagnosis for detecting system failures in large-scale wireless sensor networks. *IEEE Transactions on Wireless Communications*, 13(10):5535–5545, 2014.

[53] Ruofeng Liu, Zhimeng Yin, Wenchao Jiang, and Tian He. Wibeacon: Expanding ble location-based services via wifi. In *Proc. of ACM MOBICOM*, 2021.

[54] Vincent Liu, Aaron Parks, Vamsi Talla, Shyamnath Gollakota, David Wetherall, and Joshua R Smith. Ambient backscatter: wireless communication out of thin air. In *Proc. of ACM SIGCOMM*, 2013.

[55] Xin Liu, Zicheng Chi, Wei Wang, Yao Yao, Pei Hao, and Ting Zhu. Verification and redesign of ofdm backscatter. In *Proc. of USENIX NSDI*, 2021.

[56] Mohammad Hossein Mazaheri, Alex Chen, and Omid Abari. Mmtag: A millimeter wave backscatter network. In *Proc. of ACM SIGCOMM*, 2021.

[57] Saman Naderiparizi, Mehrdad Hessar, Vamsi Talla, Shyamnath Gollakota, and Joshua R Smith. Towards battery-free hd video streaming. In *Proc. of USENIX NSDI*, 2018.

[58] Pavel V Nikitin, KVS Rao, and Roberto D Martinez. Differential rcs of rfid tag. *Electronics Letters*, 43(8):431–432, 2007.

[59] Jiajue Ou, Mo Li, and Yuanqing Zheng. Come and Be Served: Parallel Decoding for COTS RFID Tags. In *Proc. of ACM MobiCom*, 2015.

[60] Ioannis Pefkianakis, Yun Hu, Starsky HY Wong, Hao Yang, and Songwu Lu. MIMO rate adaptation in 802.11 n wireless networks. In *Proc. of ACM MOBICOM*, 2010.

[61] John G Proakis and Masoud Salehi. *Digital communications*, volume 4. McGraw-hill New York, 2001.

[62] Hariharan Rahul, Farinaz Edalat, Dina Katabi, and Charles G Sodini. Frequency-aware rate adaptation and MAC protocols. In *Proc. of ACM MobiCom*, 2009.

[63] Ahmed Rennane, Abanob Abdelnour, Darine Kaddour, Rachida Touhami, and Smail Tedjini. Design of passive uhf rfid sensor on flexible foil for sports balls pressure monitoring. *IET Microwaves, Antennas & Propagation*, 12(14):2154–2160, 2018.

[64] James Rosenthal, Alexandra Pike, and Matthew S Reynolds. A 1 mbps 158 pj/bit bluetooth low energy (ble) compatible backscatter communication uplink for wireless neural recording in an animal cage environment. In *Proc. of IEEE RFID*, 2019.

[65] James Rosenthal and Matthew S Reynolds. A 1.0-mb/s 198-pj/bit bluetooth low-energy compatible single sideband backscatter uplink for the neurodisc brain–computer interface. *IEEE Transactions on Microwave Theory and Techniques*, 67(10):4015–4022, 2019.

[66] Mohammad Rostami, Xingda Chen, Yuda Feng, Karthikeyan Sundaresan, and Deepak Ganesan. MIXIQ: re-thinking ultra-low power receiver design for next-generation on-body applications. In *Proc. of ACM MobiCom*, 2021.

[67] Steven W Smith et al. The scientist and engineer's guide to digital signal processing. 1997.

[68] Li Sun, Souvik Sen, and Dimitrios Koutsonikolas. Bringing mobility-awareness to WLANs using PHY layer information. In *Proc. of ACM CONEXT*, 2014.

[69] Vamsi Talla, Mehrdad Hessar, Bryce Kellogg, Ali Najafi, Joshua R Smith, and Shyamnath Gollakota. Lora backscatter: Enabling the vision of ubiquitous connectivity. In *Proc. of ACM IMWUT*, 2017.

[70] Vamsi Talla, Bryce Kellogg, Shyamnath Gollakota, and Joshua R Smith. Battery-free cellphone. In *Proc. of ACM IMWUT*, 2017.

[71] Ambuj Varshney, Oliver Harms, Carlos Pérez-Penichet, Christian Rohner, Frederik Hermans, and Thiemo Voigt. Lorea: A backscatter architecture that achieves a long communication range. In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems*, 2017.

[72] Deepak Vasisht, Guo Zhang, Omid Abari, Hsiao-Ming Lu, Jacob Flanz, and Dina Katabi. In-body backscatter communication and localization. In *Proc. of ACM SIGCOMM*, 2018.

[73] Anran Wang, Vikram Iyer, Vamsi Talla, Joshua R Smith, and Shyamnath Gollakota. fm backscatter: Enabling connected cities and smart fabrics. In *Proc. of USENIX NSDI*, 2017.

[74] Haoyu Wang, Si Chen, and Wei Gong. Mobility improves accuracy: Precise robot manipulation with cots rfid systems. In *Proc. of IEEE PerCom*, 2021.

[75] Haoyu Wang and Wei Gong. Rf-pen: Practical real-time rfid tracking in the air. *IEEE Transactions on Mobile Computing*, 20(11):3227–3238, 2020.

[76] Jue Wang, Haitham Hassanieh, Dina Katabi, and Piotr Indyk. Efficient and Reliable Low-Power Backscatter Networks. In *Proc. of ACM SIGCOMM*, 2012.

[77] Jue Wang, Deepak Vasisht, and Dina Katabi. RF-IDraw: Virtual Touch Screen in the Air Using RF Signals. In *Proc. of ACM SIGCOMM*, 2014.

[78] Zhaoyuan Xu and Wei Gong. Enabling zigbee backscatter communication in a crowded spectrum. In *Proc. of IEEE ICNP*, 2022.

[79] Lei Yang, Yekui Chen, Xiang-Yang Li, Chaowei Xiao, Mo Li, and Yunhao Liu. Tagoram: Real-time tracking of mobile RFID tags to high precision using COTS devices. In *Proc. of ACM MobiCom*, 2014.

[80] Yifan Yang, Longzhi Yuan, Jia Zhao, and Wei Gong. Content-agnostic backscatter from thin air. In *Proc. of ACM MOBISYS*, 2022.

[81] Zhice Yang, Qianyi Huang, and Qian Zhang. Nicscatter: Backscatter as a covert channel in mobile devices. In *Proc. of ACM MOBICOM*, 2017.

[82] Longzhi Yuan and Wei Gong. Subscatter: Sub-symbol wifi backscatter for high throughput. In *Proc. of IEEE ICNP*, 2022.

[83] Lonzhi Yuan, Can Xiong, Si Chen, and Wei Gong. Embracing self-powered wireless wearables for smart healthcare. In *Proc. of IEEE PerCom*, 2021.

[84] Jiansong Zhang, Kun Tan, Jun Zhao, Haitao Wu, and Yongguang Zhang. A practical SNR-guided rate adaptation. In *Proc. of IEEE INFOCOM*, 2008.

[85] Pengyu Zhang, Dinesh Bharadia, Kiran Joshi, and Sachin Katti. Hitchhike: Practical backscatter using commodity wifi. In *Proc. of ACM SenSys*, 2016.

[86] Pengyu Zhang, Jeremy Gummeson, and Deepak Ganesan. Blink: A high throughput link layer for backscatter communication. In *Proc. of ACM MobiSys*, 2012.

[87] Pengyu Zhang, Colleen Josephson, Dinesh Bharadia, and Sachin Katti. Freerider: Backscatter communication using commodity radios. In *Proc. of ACM CONEXT*, 2017.

[88] Pengyu Zhang, Mohammad Rostami, Pan Hu, and Deepak Ganesan. Enabling practical backscatter communication for on-body sensors. In *Proc. of ACM SIGCOMM*, 2016.

[89] Jia Zhao, Wei Gong, and Jiangchuan Liu. Spatial stream backscatter using commodity wifi. In *Proc. of ACM MOBISYS*, 2018.

[90] Jia Zhao, Wei Gong, and Jiangchuan Liu. X-tandem: Towards multi-hop backscatter communication with commodity wifi. In *Proc. of ACM MOBICOM*, 2018.

[91] Jia Zhao, Wei Gong, and Jiangchuan Liu. Towards scalable backscatter sensor mesh with decodable relay and distributed excitation. In *Proc. of ACM MobiSys*, 2020.