TransFi: Emulating Custom Wireless Physical Layer from Commodity WiFi

Ruirong Chen
University of Pittsburgh
USA
ruirongchen@pitt.edu

Wei Gao
University of Pittsburgh
USA
weigao@pitt.edu

ABSTRACT

New wireless physical-layer designs are the key to improving wireless network performance. Adopting these new designs, however, requires modifications on wireless hardware and is difficult on commodity devices. In this paper, we show that this hardware modification in many cases can be avoided by *TransFi*, a new software technique that enables custom wireless PHY functionality on commodity WiFi transmitters via fine-grained emulation. Our basic insight is that many custom wireless signals can be emulated by manipulating the MAC payloads of WiFi MIMO streams and mixing the transmitted signals from these streams on the air. To perform such emulation, TransFi considers the target signal as a mixture of QAM constellation points on the complex plane, and reversely computes the MAC payload of each MIMO stream from one selected QAM constellation point. We implemented TransFi on commodity WiFi devices to emulate three custom wireless PHYs with diverse characteristics. Experiment results show that TransFi's accuracy of emulation is >90% when transmitting emulated data payloads at 11.4 Mbps (46x faster than existing methods), and the decoding error at this data rate is <1% (10x lower than existing methods).

CCS CONCEPTS

• **Networks** → **Network protocols**.

KEYWORDS

Wireless Physical Layer, Signal Emulation, WiFi, Quadrature Amplitude Modulation.

ACM Reference Format:
Ruirong Chen and Wei Gao. 2022. TransFi: Emulating Custom Wireless Physical Layer from Commodity WiFi. In *The 20th Annual International Conference on Mobile Systems, Applications and Services (MobiSys '22), June 25–July 1, 2022, Portland, OR, USA.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3498361.3538946

1 INTRODUCTION

Emerging mobile applications continuously raise higher requirements on wireless networks, which motivate new designs of the wireless physical layer (PHY). For example, delay-sensitive applications, such as online gaming, build on fast detection and avoidance of wireless interference that need new PHY preambles to coordinate between wireless devices [2, 35, 46]. The Internet of Things (IoT) calls for more agile usage of wireless spectrum, in order to support more concurrent wireless transmissions [3, 5, 63].

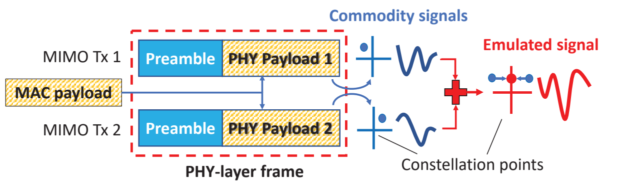


Figure 1: TransFi emulates a custom signal by mixing WiFi signals from multiple MIMO streams on the air

Adopting new PHY designs used to always involve modifications of wireless hardware, due to the incompatible formats of new wireless signals. For example, new PHY preambles for power saving [65], identification [43] and collision detection [46] cannot be produced by commodity wireless hardware. Similarly, new methods of spectrum utilization, such as OFDMA in 802.11ax [2, 56], cannot be correctly used by legacy wireless devices. Such hardware modifications, however, are difficult on most commodity wireless devices. This difficulty results in slow adoption of new wireless PHY techniques, and is also the major reason for the gap between lab prototypes and actual wireless systems in use.

To avoid such hardware modification, one approach is to selectively transmit a commodity wireless signal that best approximates to the target signal in the new PHY design. However, since each commodity wireless hardware can only produce a finite number of fixed PHY signal waveforms[1], this method is too coarse-grained to precisely approximate to the target signal that may arbitrarily appear in custom wireless PHY, and will result in large and uncontrollable approximation error when transmitting high-speed data frames. Instead, it can only be used to transmit network control or beacon frames with fixed contents for low-speed cross-technology communication (e.g., from WiFi to ZigBee [20, 33], Bluetooth [11, 38], LoRa [18, 37] and LTE [17]).

Instead, we envision that this constraint of commodity wireless hardware can be removed by *fine-grained emulation*, which mixes multiple commodity wireless signals with adaptively selected amplitudes and phases. For example, mixing two commodity signals with the same amplitude and phases at 45° and 315° creates a new signal with a phase at 0°. Further changing one signal's phase from 315°

[1]These signal waveforms are decided by the wireless PHY's modulation and coding methods. For example, number of these waveforms in WiFi depends on the QAM constellation diagram being used, and varies from 2 to 64 when the QAM constellation diagram changes from BPSK to 64-QAM.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
MobiSys '22, June 25–July 1, 2022, Portland, OR, USA
© 2022 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-9185-6/22/06…$15.00
https://doi.org/10.1145/3498361.3538946

to 225° produces a null signal with zero amplitude. On commodity WiFi, such selection of commodity wireless signals could be done by manipulating the MAC payloads of its MIMO streams[2], and the selected signals are mixed on the air when being transmitted in MIMO streams. In this way, the approximation error to the target signal from such mixture could be effectively minimized, with more MIMO streams or higher-order modulation in each stream.

Based on this insight, in this paper we present *TransFi*, a new software technique that enables custom wireless PHY functionalities on commodity WiFi transmitters. As shown in Figure 1, the TransFi software computes the commodity WiFi's MAC payload based on the target signal being emulated, and passes the computed MAC payloads to the WiFi PHY layer to produce the target signal. More specifically, it considers the target signal in each wireless symbol as a custom point on the complex plane, and selects a set of commodity QAM constellation points whose geometric mixture matches the custom point. Each selected QAM constellation point is then *reversely* computed to the MAC payload of one MIMO stream, by mimicking the WiFi data decoding process.

The major challenge of emulation is finding the optimal approximation to the target signal, from the large volume of possible selections of commodity QAM constellation points. In particular, even selections that provide the same theoretical approximation could practically result in different signals being received, due to unpredictable signal distortion during channel propagation. TransFi takes these practical factors into account, and constrains such selection within a limited scope to avoid excessive computation overhead.

The optimality of selection could be further affected by the data encoder at commodity WiFi transmitter, which makes some selected QAM constellation points fail to reversely result in legal MAC payloads. The reason is that WiFi data encoder appends parity bits to the MAC payload before passing it to PHY. The PHY payloads corresponding to some selected QAM constellation points, however, may contain illegal parity bits. To address this challenge, we append redundant bits to the PHY payload of each selected QAM constellation point, so that the expanded PHY payload always results in a legal MAC payload. In TransFi, such reverse computation of MAC payload is based on the Trellis diagram in WiFi data decoding [21, 39], and we minimize the computation and storage costs by adaptively dividing the Trellis diagram into small segments.

To our best knowledge, TransFi is the first to allow commodity WiFi to transmit high-speed data frames of custom wireless PHY with random payloads. Our detailed contributions are as follows:

- We analytically verified the possibility of emulating custom wireless signal using a combination of QAM constellation points, from WiFi MIMO streams.
- We managed to minimize the difference between the target signal being emulated and the actually emulated signal being received, by selecting the optimal set of QAM constellation points for emulation.
- We developed techniques that always produce legal MAC payloads from any selected set of QAM constellation points in emulation, and ensured the usability of these techniques on commodity WiFi devices by incorporating the characteristics of wireless PHY hardware into account.

[2]MIMO has been a standard technology in mainstream WiFi standards from 802.11n [1] to 802.11ax [3]. Old WiFi standards such as 802.11a/b/g, although still being supported, are obsolete and less used in practice [19]

TransFi can emulate a large collection of custom wireless PHY designs in both time and frequency domains, as long as they operate in the same frequency band as WiFi and do not use finer frequency-division multiplexing than WiFi[3]. In the time domain, TransFi can emulate new PHY preambles designed for different purposes such as collision detection [3] and avoidance [46], device coordination [65], and energy saving [40], by precisely emulating the required amplitude and phase of the time-domain signal. Data payloads will be emulated and transmitted after these emulated preambles. In the frequency domain, TransFi can emulate different ways of custom spectrum usage, including multi-access channels (e.g., OFDMA [28]) and spectrum adaptation [63], by converting the frequency-domain changes to the corresponding time-domain signal.

TransFi is used on commodity WiFi transmitters to emulate custom wireless signals, which may not pass the PHY signal detection of a commodity receiver. In these cases, a custom receiver that implements the Rx hardware functionality of the wireless PHY being emulated will be needed to correctly receive and decode the emulated signal, but such a custom receiver is not part of our TransFi design. In practice, many wireless systems, including most sensor network [6, 64] and IoT systems [48], are highly asymmetric, where a large amount of wireless transmitters send their locally produced data to the few receivers (a.k.a., data sinks). In these systems, while it is easier to deploy custom wireless PHY onto the few receivers via hardware upgrades, it will be difficult or even infeasible to upgrade the hardware of all transmitters. TransFi, then, will be the key enabler of new wireless PHY techniques in these systems.

We implemented TransFi on an Atheros AR9580 WiFi adapter and used it to emulate three custom wireless PHYs, including custom PHY preambles (CSMA/CN [46]), spectrum adaptation (FSA [63]) and spectrum multiplexing (OFDMA). Our experiment results have the following conclusions:

- TransFi is *accurate* and *effective*. TransFi's accuracy of emulating custom wireless signals is >90%. Based on this accuracy, the emulated wireless PHY can transmit data payloads at 11.4 Mbps, >46x faster than the best existing work [33]. Decoding errors of emulated data frames under this data rate is <1%, >10x lower than the best existing result [37].
- TransFi is *adaptive*. TransFi can well adapt to different system settings and wireless channel conditions. Even with severe channel conditions, it requires at most 2 dB of extra SNR to reliably decode the emulated signal, when compared to commodity wireless systems.
- TransFi is *lightweight*. TransFi involves the minimum computation and storage overhead. The run-time computing time of emulation is always shorter than the time gap between commodity PHY frames, hence allowing real-time transmission of emulated data frames.

2 BACKGROUND & MOTIVATION

To better understand the design of TransFi, we first introduce the basics of QAM modulation used in WiFi, and then motivate our

[3]Most WiFi systems build on orthogonal frequency-division multiplexing (OFDM), where finer multiplexing leads to more OFDM subcarriers. See detailed discussions about such scope of TransFi emulation in Section 12.

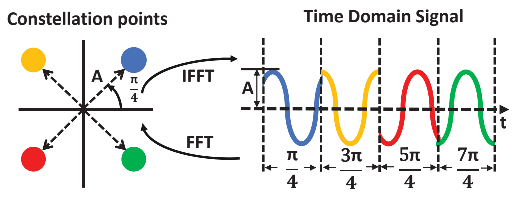


Figure 2: QAM modulation

design by demonstrating the technical difficulty of emulation due to WiFi's data encoding and decoding.

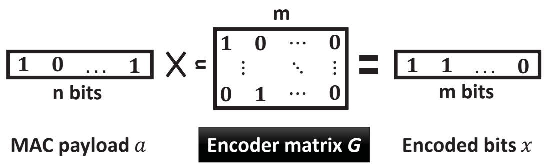
2.1 QAM Modulation

Quadrature amplitude modulation (QAM) is the technical foundation of WiFi. QAM modulates data bits to constellation points on the complex plane, which are converted into time-domain signals by FFT. For example, Figure 2 shows that a time-domain signal converted from QPSK has a constant amplitude but varying phases among $\pi/4, 3\pi/4, 5\pi/4$, and $7\pi/4$.

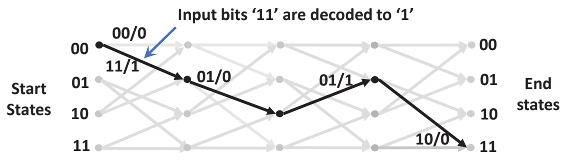
Figure 2 shows that any time-domain signal can be represented by its amplitude and phase over the carrier wave, corresponding to a point on the complex plane. The major challenge of producing new wireless signals, though, is that the desired point may not exist in the constellation diagram used by commodity WiFi. This challenge motivates our design of TransFi, which utilizes multiple MIMO streams to approximate to such a custom constellation point.

2.2 Wireless MIMO

A wireless MIMO transmitter uses multiple Tx radios to transfer synchronized data streams in the same spectrum [55], and such synchronization can be achieved at the nanosecond level [50] by sharing the same clock between Tx radios. TransFi utilizes this precise timing to ensure that MAC payloads in different MIMO streams are always aligned over time, which is vital to correctly mix the corresponding wireless signals on the air.



(a) Data encoding



(b) A Trellis diagram of a 1/2 decoder, which decodes input bits of '11010110' to output bits '1010'

Figure 3: Data encoding and decoding in WiFi

2.3 WiFi Data Encoding and Decoding

Data encoding is used in WiFi for error correction. For example in Figure 3(a), a convolutional encoder with a n/m code rate uses a $n \times m$ matrix to produce m encoded bits from n input bits as $\mathbf{x} = \mathbf{a} \cdot \mathrm{G}$, appending $m - n$ parity bits.

Encoded bits are decoded in WiFi using the Viterbi algorithm [21], which finds a viable path in a Trellis diagram for the given input. As shown in Figure 3(b), on a n/m decoder, the Trellis diagram for decoding a M-bit input contains $\lceil M/m \rceil$ steps, and each step decodes m encoded bits to n output bits. Each state in the Trellis diagram has m outgoing edges to the next step, and there are only 2^n legal inputs among 2^m possibilities. For example, in the first step shown in Figure 3(b), the start state '00' only allows legal inputs '00' and '11', which are decoded as '0' and '1', respectively.

According to Figure 3(b), it is possible that some input bits do not correspond to any path in the Trellis diagram. If these input bits correspond to TransFi's selected constellation points for emulation, they cannot be reversely computed to MAC payloads. This possibility motivates our TransFi design that adds redundancy to reverse computation of MAC payloads.

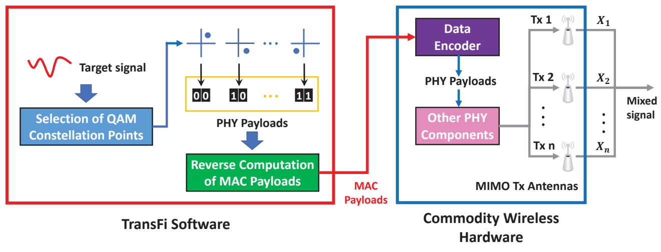


Figure 4: TransFi system overview

3 SYSTEM OVERVIEW

As a software technique, TransFi minimizes the error of emulating the target signal by selecting the optimal set of commodity QAM constellation points being used at the commodity WiFi transmitter. As illustrated in Figure 4, the TransFi software computes the MAC payload for each WiFi MIMO stream from the selected QAM constellation point, and uses the computed MAC payloads as the output to the commodity WiFi hardware, where the transmitted wireless signal on each MIMO Tx antenna is produced based on the corresponding MAC payload. In this way, our design of TransFi is generic and can be used to emulate a large collection of custom wireless signals that differ from commodity WiFi signals in both time and frequency domains.

3.1 Mixing Wireless Signals on the Air

In TransFi, the wireless signals transmitted by different MIMO streams are mixed on the air. For example, for a WiFi MIMO with 2 Tx antennas, the time-domain signals being transmitted from the two MIMO streams can be written as

$$S_1(t) = A_1 \cos(2\pi f t + \phi_1), \quad S_2(t) = A_2 \cos(2\pi f t + \phi_2),$$

where f is the carrier frequency, $\{A_1, A_2\}$ and $\{\phi_1, \phi_2\}$ are the signals' amplitude and phases, respectively. Then, when these two signals are mixed on the air and $A_1 = A_2$, the mixed signal will be

$$\begin{aligned} S(t) &= A\cos(2\pi f t + \phi_1) + A\cos(2\pi f t + \phi_2) \\ &= A\cos(\frac{\phi_1 - \phi_2}{2})\cos(2\pi f t + \frac{\phi_1 + \phi_2}{2}) \qquad (1) \\ &= \tilde{A}\cos(2\pi f t + \tilde{\phi}), \end{aligned}$$

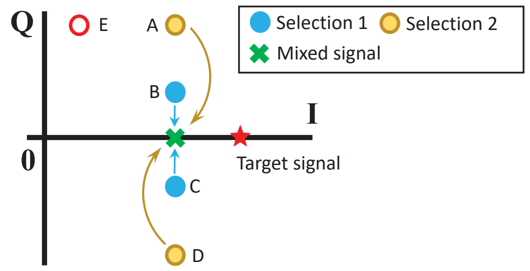


Figure 5: Selection of QAM constellation points. An example of using two 64-QAM constellation points to emulate a BPSK constellation point (e.g., for emulating custom preambles modulated by BPSK) is shown.

where $\tilde{A} = A\cos(\frac{\phi_1-\phi_2}{2})$ and $\tilde{\phi} = \frac{\phi_1+\phi_2}{2}$. If the constellation points' amplitudes are not equal (e.g., A and E in Figure 5), the mixed signal's amplitude and phase are similarly computed.

Figure 5 shows that this mixed signal corresponds to the geometric combination of the two QAM constellation points on the complex plane. Eq. (1) can be iteratively extended to more Tx antennas. For a given QAM modulation scheme that provides a collection of signal phases, the more Tx antennas being used, the more custom signals can be emulated. On the other hand, using higher-order QAM modulations also helps improve the granularity of emulation.

3.2 Selection of QAM Constellation Points

In TransFi, we seek the selection of commodity QAM constellation points that provides the optimal approximation to the target signal, but Figure 5 shows that multiple selections that provide the same theoretical approximation may coexist. To address this ambiguity, we demonstrated from both analytical and experimental perspectives that these selections result in different signal distortions in practice due to channel propagation, and further develop algorithms that decide the best selection with the minimum computation complexity. These details can be found in Section 4.

3.3 Reverse Computation of MAC Payload

As shown in Figure 4, TransFi outputs MAC payloads to different MIMO streams of the commodity WiFi transmitter. Hence, after having selected the QAM constellation points for emulation, TransFi decides the MAC payload for each MIMO stream, to ensure that the actually transmitted signals in WiFi MIMO streams match the selected constellation points. The MAC payload of each MIMO stream is decided via reverse computation from the corresponding PHY payload, which uniquely maps to the constellation point given a specific QAM modulation scheme. For example, when the 4 QPSK constellation points in Figure 2 are being used for emulation, the corresponding PHY payloads are $\{00, 01, 10, 11\}$, respectively.

Such reverse computation of MAC payload in TransFi mimics the data decoding process in commodity WiFi. Being different from existing work that is limited to offline computation for pre-defined preambles from a pre-computed inverse of encoder matrix [24, 33, 37], TransFi aims to achieve real-time computation of MAC payloads for arbitrary PHY payloads, e.g., corresponding to random data frames being transmitted. The major challenge, as described in Section 2.2, is that some of the selected PHY payloads may not

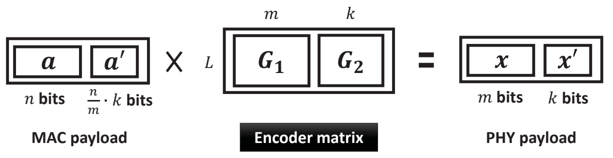


Figure 6: Reverse computation of MAC payload: the encoder's perspective

correspond to a legal MAC payload in commodity WiFi. To address this challenge, given a code rate (n/m) used in commodity WiFi, our approach is to compute and append k redundant bits $(\mathbf{x}')$ to the original m bits of PHY payload $(\mathbf{x})$, and the MAC payload to be reversely computed will be correspondingly expanded from n bits $(\mathbf{a})$ to $L = n + \frac{n}{m} \cdot k$ bits $([\mathbf{a}, \mathbf{a}'])$. In this way, when k satisfies that $L \geq m$, from the encoder's perspective we have

$$\mathbf{x} = [\mathbf{a}, \mathbf{a}'] \cdot \mathbf{G}_1, \tag{2}$$

and $\mathbf{G}_1$ is a fully ranked matrix. Hence, Eq. (2) ensures that a legal MAC payload $[\mathbf{a}, \mathbf{a}']$ always exists for a given $\mathbf{x}$. In practice, k will be selected as the smallest integer that results an integer value of L and satisfies $L \geq m$. For example, with a $2/3$ code rate, we have $k = 3$ for every 3 bits of PHY payload.

With the $\mathbf{x}'$ being decided, the MAC payload $[\mathbf{a}, \mathbf{a}']$ can be computed by applying $[\mathbf{x}, \mathbf{x}']$ as the input to a n/m WiFi decoder. The difficulty, however, is that for a given $\mathbf{x}$, not any $\mathbf{x}'$ can ensure that the decoder outputs a legal MAC payload. An intuitive solution to deciding $\mathbf{x}'$ is to apply each possible $\mathbf{x}'$ to the decoder until a legal MAC payload is produced, but is computationally expensive in commodity WiFi where encoders and decoders operate an entire frame at once. For example, the minimum payload of an 802.11n MAC frame contains 26 bits that correspond to 52 bits of PHY payload with a 1/2 code rate, and $\mathbf{x}'$ hence contains 52 bits at least.

Instead, we cache all the valid paths in the decoder's Trellis diagram for $m + k$ input bits, and only match the actual $\mathbf{x}$ to one valid path at run-time. To minimize the amount of storage overhead, we divide the Trellis diagram into small segments, and develop algorithms to identify and address the possible inconsistency across these segments during run-time matchmaking. These details are in Section 5.

Both $\mathbf{x}$ and $\mathbf{x}'$ will be transmitted. Since commodity WiFi builds on OFDM, we will ensure that $\mathbf{x}$ and $\mathbf{x}'$ are transmitted in two different but continuous sets of OFDM subcarriers, and match $\mathbf{x}$ to the band operated by the receiver, which can easily remove the out-of-band $\mathbf{x}'$ at its RF frontend. The Tx's channel bandwidth needs to be wider than the Rx bandwidth, and their ratio is determined by the WiFi code rate as $1/coderate$. With higher code rates being used (e.g., 5/6 in 802.11n), the channel bandwidth of the emulated wireless PHY can reach >80% of the WiFi channel bandwidth.

4 SELECTING CONSTELLATION POINTS

When multiple selections of QAM constellation points provide the same theoretical approximation to the target signal, constellation points with larger amplitudes will produce more distortion in the emulated signal during channel propagation. Hence, QAM constellation points with the smallest amplitudes among the available choices will be selected.

4.1 The Impact of Signal Amplitudes

When a selection of QAM constellation points is used for emulation, the received signal can be written as

$$\begin{aligned} Y &= H_1X_1 + H_2X_2 + \ldots + H_nX_n \\ &= H_1(\tilde{X} - d_1) + H_2(\tilde{X} - d_2) + \ldots + H_n(\tilde{X} - d_n) \\ &= (H_1 + H_2 + \ldots H_n)\tilde{X} - (H_1d_1 + H_2d_2 + \ldots + H_nd_n), \end{aligned} \tag{3}$$

where X_i is the time-domain signal corresponding to the i-th QAM constellation point, $\tilde{X} = \sum_{i=1}^{n} X_i$ is the mixed signal, $d_i = \tilde{X} - X_i$, and H_i is the response of the channel between the i-th MIMO Tx antenna and the receiver. In this case, $\{d_i\}$ with larger amplitudes, which correspond to longer distances between the selected QAM constellation points and target signal on the complex plane, amplify the impact of channel propagation as $H_1d_1 + H_2d_2 + \ldots + H_nd_n$ in Eq. (3). This impact, in turn, leads to larger distortion in the received signal Y, measured as the difference between the target signal and the emulated signal being received.

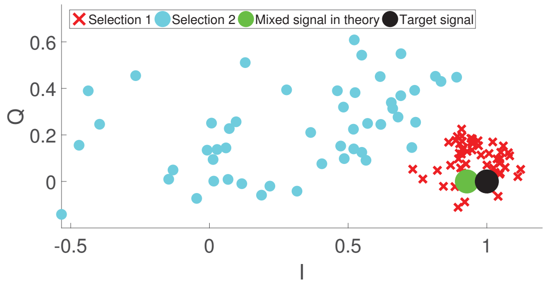


Figure 7: The received signals from different selections of QAM constellation points being used

To verify such distortion from a larger amplitude in emulation, we use the two selections of 64-QAM constellation points shown in Figure 5 to emulate a BPSK constellation point from commodity WiFi with 12dB SNR. As shown in Figure 7, the received signal produced by Selection 2 contains much larger distortion compared to that produced by Selection 1. Statistical analysis over 8,000 collected data samples further demonstrates that the average distortion in Selection 2 is 0.6534 on the complex plane, while such distortion in Selection 1 is only 0.1368 on average.

4.2 Iterative Selection

Results in Section 4.1 suggest that, the selection of QAM constellation points that results in the minimum amplitudes of $\{d_i\}$ should be used for emulation. For example in Figure 5, Selection 2 is a better choice than Selection 1. An intuitive approach is to compute the amplitudes of $\{d_i\}$ for all possible selections, but is computationally expensive in practice.

Instead, TransFi first decides a range for such selection, and then iteratively selects each constellation point based on the previously selected points within this range. As shown in Figure 8 which selects among 64-QAM constellation points for emulation, the range of selection is a square on the complex plane that centers at the target signal and has a side length of $2d_{\min}$, where $d_{\min}$ is the minimum distance between two commodity QAM constellation points being used. This range ensures to include the 4 constellation points that are the closest to the target signal on the complex plane.

Algorithm 1 Iterative selection of constellation points

```
Input: $T$: Target signal, $N$: No. of MIMO streams, $C$: Set of QAM
    constellation points in range of selection
Output: $\{P_i\}$: The selected $N$ constellation points
1: $P_1 \leftarrow \arg\min_{c \in C} |T - c|$
2: $n \leftarrow 2$
3: while $n \leq N$ do
4:     $P'_n \leftarrow T - \sum_{i=1}^{n-1} P_i$        //Ideal position of $P_n$
5:     $P_n \leftarrow \arg\min_{c \in C} |P'_n - c|$
6:     $n \leftarrow n + 1$
7: end while
```

Our approach to selecting constellation points within this range is described in Algorithm 1[4] and illustrated by the example in Figure 8. The first constellation point is selected as the closest one to the target signal (D in Figure 8). Afterwards, in the i-th round of selection, TransFi first decides the "ideal" position of the i-th constellation point, which can produce a mixed signal with the $i-1$ selected points to exactly match the target signal. The i-th constellation point, then, is selected as the closest one to the ideal position. In this way, such iterative selection can be applied to any number of MIMO streams being used. For example in Figure 8, the 2nd, 3rd and 4th selections will be A, D and B, respectively.

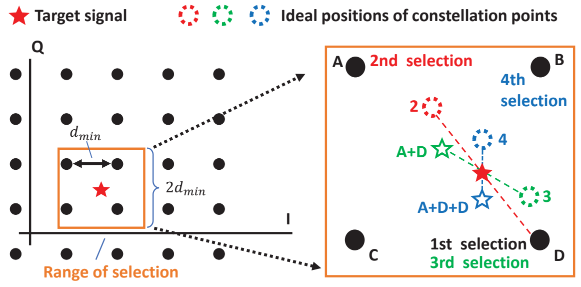


Figure 8: Selection of 64-QAM constellation points

5 COMPUTATION OF MAC PAYLOAD

In this section, we describe the technical details of reversely computing the MAC payloads in the TransFi software.

5.1 The Basic Approach

As described in Section 3.3, our primary approach to minimizing the run-time overhead of such reverse computation is to cache all the viable paths of the Trellis diagram. For example, for the Trellis diagram in Figure 3(b), the start state 00 corresponds to 16 viable paths listed in Figure 9. Thus, for any 4-bit PHY payload **x** that starts with '00' or '11', it matches an entry in the cached table and the corresponding MAC payload can be found[5]. In practice, this table could be cached as a binary tree to enforce a fast binary search.

The key drawback, however, is that large storage is needed for caching. For example, for a M-bit input and a 1/2 code rate, there

[4] All the additions in Algorithm 1 indicate geometric combinations of the corresponding constellation points on the complex plane.
[5] For other payload values of **x**, paths from other start states can be similarly searched. Eq. (2) then ensures that a matching path must be available.

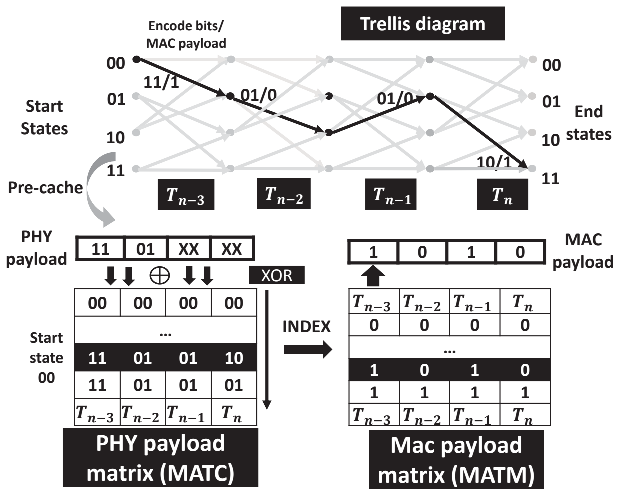


Figure 9: Finding the MAC payload from the cached paths in the Trellis diagram shown in Figure 3(b). Only the paths starting from the state '00' are shown.

will be $k \cdot 2^{M/2}$ viable paths in the Trellis diagram with k start states. When the MAC payload size is 40 bits in a data frame[6], more than 2^{40} diagram paths will be cached and the storage space needed could be $> 1,000$ GB.

5.2 Reduction of Needs on Storage Space

To reduce the storage need, we divide the Trellis diagram into segments and cache the paths in each segment, so that the number of cached paths reduces from $k \cdot 2^{M/2}$ to $k \cdot n \cdot 2^{M/2n}$ with n segments. For example, for a 40-bit MAC payload, dividing the Trellis diagram into 4 segments reduces the storage space to < 100 KB.

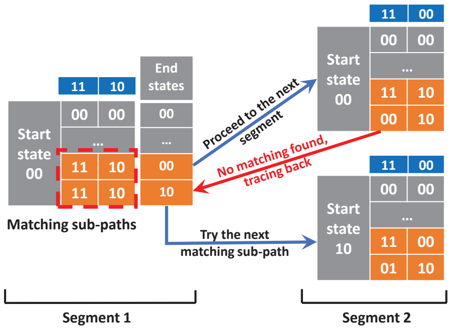

Start state 00
End states
Matching sub-paths
Proceed to the next segment
No matching found, tracing back
Try the next matching sub-path
Start state 00
Start state 10
Segment 1
Segment 2

Figure 10: Possible backtracing across different Trellis diagram segments. An example of dividing a Trellis diagram for 8-bit inputs into 2 segments is shown.

The start and end states of paths will also be cached. At run-time, TransFi divides a M-bit input into n parts, and seeks for a sub-path that matches the i-th part of input from the i-th Trellis diagram segment. These matching sub-paths are then concatenated according to their start and end states.

[6]Most WiFi data frames contain larger payloads. For example, the payload size of a 802.11n MAC frame ranges between 26 and 524,288 bits.

However, backtracing may happen across multiple segments. For example in Figure 10 with a given input '11101100', it will be divided into '1110' and '1100' to search for a matching sub-path in the two Trellis diagram segments, respectively. Note that multiple matching sub-paths may exist in a segment. Whenever a matching sub-path is found for '1110' in segment 1, it will proceed to segment 2 according to the end state of this sub-path, which will also be the start state in segment 2. However, it is possible that a matching sub-path with this start state cannot be found in segment 2. In this case, the matching procedure will backtrace to segment 1 and try to use the next matching sub-path in segment 1.

Our experiments show that such backtracing happens in 25% of data decoding cases, slightly increasing the run-time computing time. The more Trellis diagram is divided, the smaller each segment will be and hence more backtracing will happen. We will investigate such tradeoff in Section 9.5.

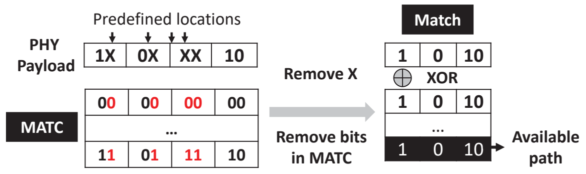


Figure 11: Bit shuffling due to interleaver

5.3 Impact of WiFi Interleaver

In commodity WiFi, the encoded bits will also be processed by an interleaver before they are sent to QAM modulation. Correspondingly, in the PHY payload being used to reversely compute the MAC payload, the bits in $\mathbf{x}$ and $\mathbf{x}'$ are shuffled as shown in Figure 11. However, since such shuffling pattern is fixed and pre-known, we can perform similar bit shuffling in the cached tables of Trellis diagram paths, to ensure that MAC payloads can always be correctly computed and $\mathbf{x}$ and $\mathbf{x}'$ are transmitted over two different but continuous sets of OFDM subcarriers. For example, for a 4-bit $\mathbf{x}$ and a 4-bit $\mathbf{x}'$, after shuffling, the bit positions of $\mathbf{x}'$ in the expanded PHY payload will be always 2, 4, 5 and 6.

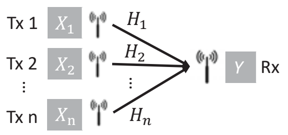
6 CHANNEL ESTIMATION

Correct channel estimation is the key to decoding the emulated data payloads at the receiver. As multiple wireless signals are mixed in the air, TransFi's communication channel is illustrated in Figure 12(a), as the mixture of sub-channels from individual Tx antennas to the Rx. When the transmitted preambles in MIMO Tx streams are $X_1, X_2, ..., X_n$, the linear estimator of this mixed channel is

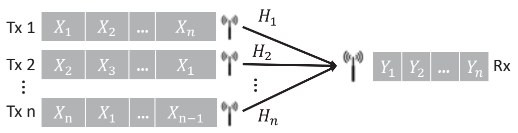
$$H_{linear} = \frac{Y}{X_1 + X_2 + ... + X_n} = \frac{H_1X_1 + H_2X_2 + ... + H_nX_n}{X_1 + X_2 + ... + X_n}, \tag{4}$$

where $H_1, H_2, ..., H_n$ indicate the channel response of individual sub-channels. This channel estimator, however, is variant with respect to the specific preambles being used and hence cannot reflect the actual condition of this mixed channel.

Instead, in TransFi we use $H = H_1 + H_2 + ... + H_n$ as the channel estimator, because $H_{linear} = H$ when $X_1 = X_2 = ... = X_n$. The major difficulty, however, is that wireless signals transmitted from different MIMO Tx streams can never be exactly the same, due to different cyclic shifts in these streams. Our solution, as shown



(a) The mixed channel



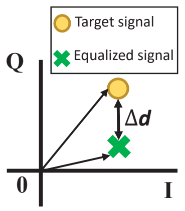
(b) Our method of channel estimation

Figure 12: Channel estimation in TransFi

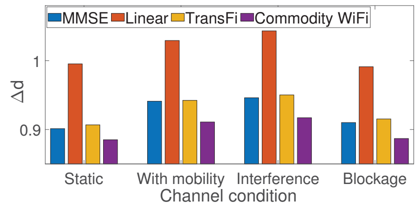
in Figure 12(b), is to transmit a series of n preambles in a shifted manner[7], and the received signal can then be written as

$$\begin{pmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{pmatrix} = \begin{pmatrix} H_1 & H_2 & \dots & H_n \\ H_n & H_1 & \dots & H_{n-1} \\ \vdots & \vdots & \ddots & \vdots \\ H_2 & H_3 & \dots & H_1 \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \\ \vdots \\ X_n \end{pmatrix}, \tag{5}$$

from which we can compute $H_1, H_2, ..., H_n$ from the pre-known $\{X_i\}$ and the received $\{Y_i\}$. Such computation only involves one-time matrix multiplication and is lightweight at the receiver. Also, since the channel estimation is only done once per transmitted frame, transmitting n preambles incurs negligible timing overhead.



(a) Measurement metric



(b) Comparison of errors

Figure 13: Measurement of channel estimation error

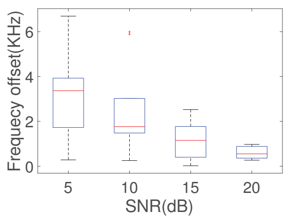
To evaluate this channel estimator, as shown in Figure 13(a), we measure the error of channel estimation as the distance Δd on the complex plane between the expected signal and the received signal after equalization. Our channel estimator, then, is compared with the commodity linear channel estimator described in Eq. (4), the MMSE estimator[8] and the commodity WiFi channel estimator, in 4 different scenarios: 1) a static transmitter; 2) a mobile transmitter; 3) with external interference; 4) the wireless signal propagates through a concrete wall. Results in Figure 13(b) show that the error of our channel estimator in these scenarios is only 0.6% higher than MMSE and 3.2% higher than that in commodity WiFi.

6.1 Other Channel Factors

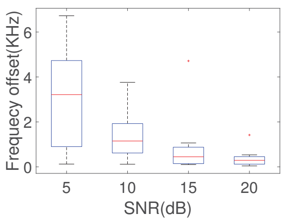
The frequency, timing and phase of the emulated signal could be affected by the wireless channel.

[7]In practice, we use these preambles as the ones being specified in the custom wireless PHY being emulated.
[8]The MMSE estimator [54, 61] achieves the minimum channel estimation error, but is rarely used in practice due to its high computational complexity.

Frequency shift. To compute and compensate the possible frequency shift of the emulated signal, we follow the same method being used in commodity WiFi. As shown in Figure 14, TransFi's frequency shift after such compensation is at the same level with that in commodity WiFi, and is far below 200 kHz that is required in the 802.11 standard [1].



(a) TransFi



(b) Commodity WiFi

Figure 14: The frequency offset after compensation

Frequency selective fading. Such heterogeneous fading is only significant if the transmitted signal bandwidth is larger than the channel's coherence bandwidth. Since commodity WiFi builds on OFDM, it is known to be resistant to frequency selective fading [14, 29, 49]. For example, the subcarrier bandwidth in WiFi is 312.5 kHz and WiFi's coherence bandwidth is >3.68 MHz [42], and each OFDM subcarrier in WiFi can hence be considered as a flat channel. Since TransFi individually emulates the target signal in each subcarrier, it will experience little impact from frequency selective fading.

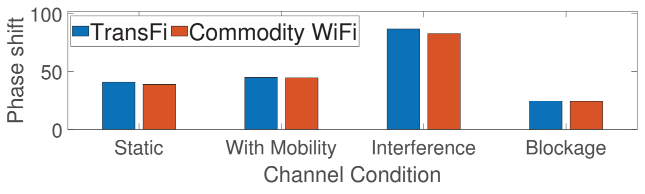


Figure 15: The amount of phase shift after correction

Phase shift. The phase shift in the received signal is a combined effect of channel propagation, signal distortion and noise. To correct such phase shift, TransFi emulates the standard pilot pattern in commodity WiFi [1, 9, 12], and computes the phase difference in the received pilots. As shown in Figure 15 over the four scenarios described in Section 6, after such correction, the remaining amount of phase shift in TransFi's emulated signal is similar to that in commodity WiFi.

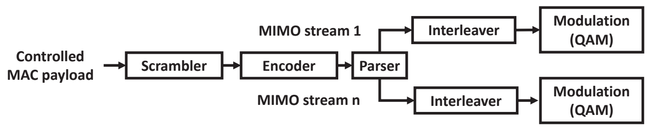


Figure 16: PHY layer of commodity WiFi

7 IMPLEMENTATION

To implement TransFi on the Qualcomm Atheros AR9580 WiFi adapter, we address the following challenges brought by different PHY components in WiFi Tx hardware, with respect to the commodity WiFi PHY as shown in Figure 16.

Scrambler seed. In commodity WiFi, a scrambler shuffles the input MAC payload based on an integer scrambler seed, to reduce inter-carrier interference. On Atheros AR9580, the scrambler seed is randomly initiated every time when the hardware powers up, and then the chipset circularly increments the seed by 1 in [1, 127] every time when a frame is transmitted. However, the value of scrambler seed embeds in wireless PHY hardware and is not directly visible to the transmitter's MAC layer.

To correctly retrieve the value of scrambler seed, when the TransFi transmitter powers up, the TransFi software sends a probing frame to the receiver, which extracts the initial scrambler seed value[9] and sends this value back to the transmitter. This value, then, will be used to decide the scrambler seed in subsequent data frames based on the increments.

Service bits and frame configuration header. The commodity WiFi adapter injects 16 service bits and 288 bits of frame configuration header into the first symbol of each PHY frame. To avoid the impact of these bits, TransFi starts emulation from the second symbol of each frame, by adding dummy bits to the beginning of MAC payload for emulation.

Cyclic prefix. The Cyclic Prefix (CP) in commodity WiFi is a time-domain signal that lasts 0.8us and appended to both the beginning and end of each data symbol. TransFi also takes these CPs into account when deciding the timing of the signal being emulated, to ensure precise emulation.

8 EVALUATION METHODOLOGY

In our evaluation, as shown in Figure 17(a), we execute the TransFi software on a Dell Precision 7820 workstation with a Netely N450B WiFi adapter. Each MIMO stream transmits 802.11n data frames with 5400 bytes of data payloads, using 64-QAM (5/6) over a 40 MHz channel. The target signal's bandwidth is set to 20 MHz, and is emulated by TransFi in subcarriers 74-127 of the 40 MHz channel. To ensure statistical convergence, results from each experiment are averaged over 2,540 emulated frames.

For the purpose of experimental evaluation and analysis, we use a WARP v3 SDR [13] to implement the Rx hardware functionality of selected custom wireless PHYs, but our evaluation results can be equivalently applied to manufactured receivers of these custom wireless PHYs without modifying their hardware[10].

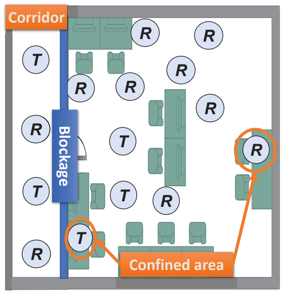
Custom Wireless PHYs being Emulated: The modifications of custom wireless PHY designs on the transmitted wireless signal can happen in both the time domain and frequency domain, both of which are evaluated in our experiments.

In the time domain, most modifications result in custom PHY preambles, and TransFi can emulate different preambles by approximating the mixed signal from multiple MIMO streams to the target signal's amplitude and phase. In particular, we choose the *CSMA/CN* preamble for WiFi collision detection [46], where frame collision is detected when the correlation result over this custom preamble is low, as the target signal to evaluate TransFi's performance of emulation, and details of such evaluation are in Section 9. In addition, we also emulate data payloads being transmitted after the emulated custom preambles, and experiment results on such data transmission demonstrate TransFi's performance of supporting practical data communication in custom high-speed wireless networks[11].

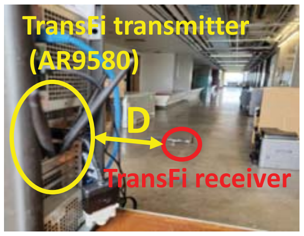
[9]This scrambler seed value embeds in the preamble of the transmitted frame and can hence be extracted by software at the receiver.

[10]Note that, many selected custom wireless PHY designs for emulation are lab prototypes and have not been commercialized.

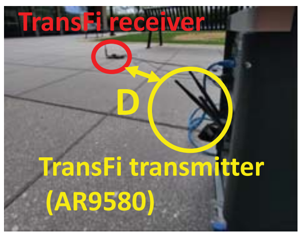
(a) Experimental system setup



(b) Office scenario (R:Rx T:Tx)



(c) Corridor scenario



(d) Outdoor scenario

Figure 17: System setup and scenarios for evaluation

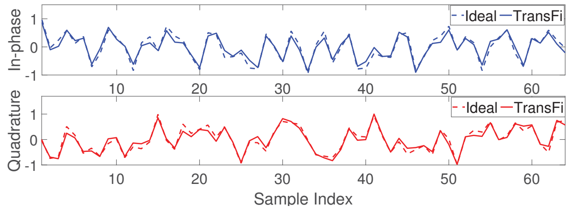
In the frequency domain, we evaluate TransFi's capability of adopting new PHY methods of custom spectrum usage, by converting the corresponding frequency-domain target signal to the time domain. These PHY methods include: 1) *Fine-grained spectrum adaptation (FSA)* [63] that allows multiple transmitters to concurrently transmit with narrower bandwidths in the same channel; and 2) *OFDMA* [23, 47, 58] that assigns subsets of subcarriers to multiple clients and allow them to concurrently transmit in the same channel. Details of these evaluations are in Section 10.

Evaluation scenarios: We evaluate TransFi in both indoor and outdoor scenarios. First, TransFi is evaluated in a $10m \times 10m$ office shown in Figure 17(b). We place the wireless transmitters and receivers at different positions, to produce different channel characteristics: 1) regular *Indoor* conditions; 2) a *Confined area* with metal objects next to wireless transceivers; 3) the Tx and Rx are placed on two sides of *Blockage* that is either a wood door, a concrete wall or a metal brick; 4) an environment with intermittent *Interference* from another WiFi device that transmits over 50% of time.

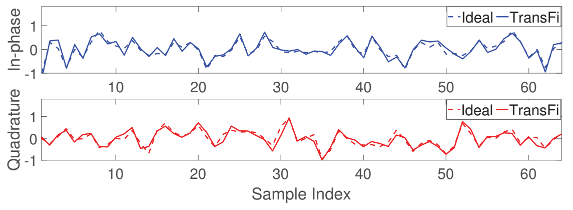
Second, we also evaluate TransFi in other scenarios with different channel conditions and longer communication distances. In the *Corridor* scenario shown in Figure 17(c), the communication distance (D) ranges between 5m and 25m. The maximum communication distance in the *Outdoor* scenario shown in Figure 17(d) is further enlarged to 50m.

Comparisons: In these evaluations, TransFi will be compared with the existing emulation techniques listed below. Since these existing techniques are limited to emulating the target signal by using only one data stream at the transmitter, our evaluation results will

[11]In comparison, most of existing coarse-grained emulation techniques are limited to low-speed wireless networks such as ZigBee [20, 33] and Bluetooth [11, 38], and are incapable of supporting high-speed data transmission at Mbps.



(a) Emulated LTF



(b) Emulated L-SIG

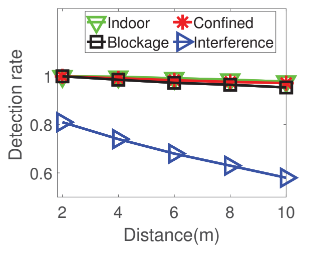
Figure 18: Emulated time-domain signal

show that they are incapable of precisely approximating to the target signal in custom wireless PHY and result in low network performance.

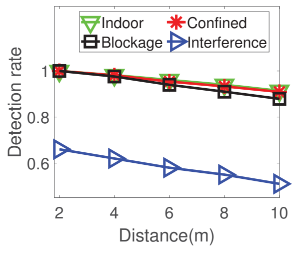
- *WeBee* [33]: It emulates a ZigBee signal by controlling a portion of WiFi signal from a single Tx, whose spectrum overlaps with the ZigBee spectrum.
- *SDR-Lite* [24]: It performs coarse-grained emulation on the WiFi OFDM preambles over a 20 MHz channel, with a single 40 MHz WiFi Tx.
- *XFi* [37]: It enables cross-technology communication (CTC) from ZigBee/LoRa to WiFi by reconstructing the WiFi receiver's decoded data to ZigBee/LoRa signal.

9 EVALUATION ON TIME-DOMAIN EMULATION

To produce a WiFi PHY frame that contains the custom CSMA/CN preamble[46], TransFi first emulates all the standard WiFi preambles, and appends the emulated CSMA/CN preamble afterwards. Besides, when multiple MIMO streams in commodity WiFi mix on the air, the structure of commodity WiFi frames is completely broken up and the original data payloads in commodity WiFi frames will be undecodeable after the mixture. In this case, to make complete WiFi PHY frames, TransFi also emulates WiFi data payloads following the emulated CSMA/CN preamble, and we also evaluate the communication performance of these emulated data payloads.



(a) LTF

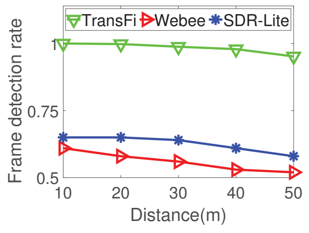


(b) L-SIG

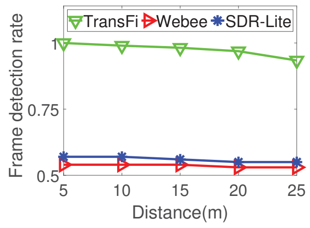
Figure 19: Detection rates on emulated preambles

9.1 Emulating Standard WiFi Preambles

We examine TransFi's performance on emulating Long Training Sequence (LTF) and Legacy SIGNAL Field (L-SIG) preambles, which are standard preambles in 802.11n. Two MIMO Tx streams are used with a 20 dB Tx/Rx gain. Figure 18 shows that the emulated time-domain signals closely match those in commodity WiFi under the same channel condition.



(a) Outdoor



(b) Corridor

Figure 20: FDRs using emulated LTF preambles

We further examined the receiver's detection rate on emulated preambles based on autocorrelation, and follow commodity WiFi to set the autocorrelation threshold as 0.8 [4, 15, 52]. Figure 19 shows that, when the communication distance varies from 2m to 10m in different indoor settings, the detection rates on both emulated preambles retain >90%. Even with strong interference, this detection rate can be >60%.

The LTF preambles are used in commodity WiFi for frame detection. Hence, we evaluated the frame detection rate (FDR) using our emulated LTF preambles and compared such rate with that in WeBee and SDR-Lite, in both outdoor and corridor environments with a Tx/Rx gain of 20dB. Results in Figure 20 show that in corridor and outdoor scenarios, even when the communication distance increases to 50m, TransFi's emulated preambles can still retain the FDR to be almost 100%. In contrast, such detection rates of WeBee and SDR-Lite are limited at 50%-65%, due to their coarse-grained emulation that is incapable of precisely approximating to the target signal with a single data stream at the transmitter.

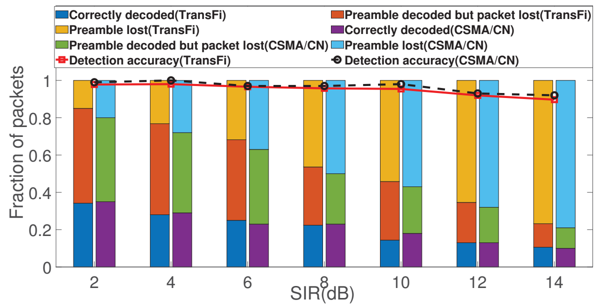


Figure 21: Performance of emulating the custom CSMA/CN preamble under different SIR

9.2 Emulating the CSMA/CN Preamble

We apply the emulated CSMA/CN preamble for collision detection, and then compare the successful collision detection rate, packet detection rate and data decoding rate with their counterparts reported in [46] under the same condition. As shown in Figure 21, when the signal-to-interference ratio (SIR) varies between 2 dB and 14 dB, the performance of collision detection using emulated preambles closely approximates to that with custom hardware in [46]. In all cases, TransFi achieves >95% accuracy for collision detection.

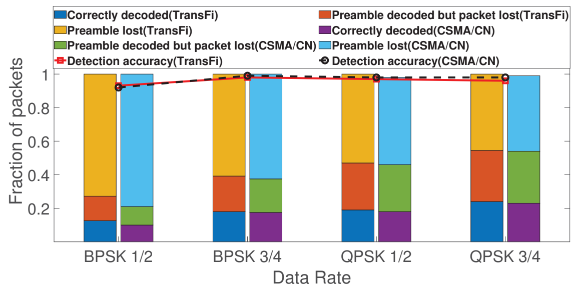


Figure 22: Performance of emulating the custom CSMA/CN preamble with different bit rates

We further investigate such emulation performance with different bit rates used in modulating the CSMA/CN preamble [46]. As shown in Figure 22, the performance difference between the emulated preamble and the preamble produced by custom hardware is always within 5%.

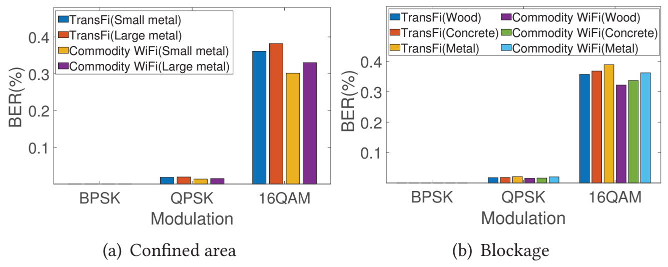


Figure 23: Data decoding errors under different channel conditions with 8dB SNR

9.3 Emulating WiFi Data Payloads

To produce complete WiFi PHY frames, the data payloads should be emulated following the emulated CSMA/CN preamble. We emulate data payloads that are modulated by BPSK, QPSK, and 16-QAM with 1/2 code rate. Results in Table 1 show that TransFi requires at most 1.1 to 1.6 dB of extra SNR to achieve 1% Bit Error Rate (BER) for data decoding at the receiver, compared to commodity WiFi.

SNR(dB)	BPSK(1/2)	QPSK(1/2)	16-QAM(1/2)
4	1.33(0.56)	4.01(2.98)	49.36(48.89)
8	0.12(0)	1.16(0.52)	39.63(32.01)
12	0(0)	0(0)	8.03(5.52)
16	0(0)	0(0)	1.033(0.28)

Table 1: BER of emulated data payloads using 2 Tx. Numbers in brackets are BER of commodity WiFi.

We further evaluated such BER over different environment settings as shown in Fig 17(b), with an 8dB fixed SNR. Figure 23 shows that, even under severe channel conditions (e.g., signal propagation path is blocked by a concrete wall), the BER for TransFi is at most 9% higher than commodity WiFi.

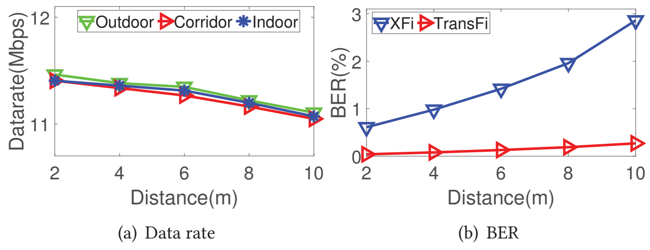


Figure 24: The data rate and BER when transmitting TransFi's emulated data payloads

Based on such low BER, Figure 24(a) shows that the emulated data payloads in TransFi can be reliably transmitted under different channel conditions. It achieves a maximum data rate of 11.4 Mbps when emulating data payloads modulated by 16-QAM with 1/2 code rate and 20dB SNR, 46x faster than that in WeBee [33]. Under this setup, Figure 24(b) shows that TransFi's BER is < 0.3% indoor in all communication distances, 10x lower than that in XFi [37], which produces much more bit errors due to its coarse-grained emulation.

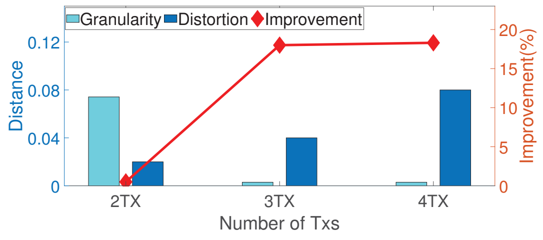


Figure 25: Emulation accuracy with more Txs

9.4 Emulation with More Tx Streams

In theory, using more MIMO Tx streams in emulation improves the its granularity. On the other hand, higher distortion may be also brought to the emulated signal due to MIMO spatial diversity. In our experiments, we measure both such emulation granularity and signal distortion, by measuring the signal distortion as the average distance between the target signal and the emulated signal on the complex plane. As shown in Figure 25, the emulation granularity increases when the number of Tx streams increases from 2 to 3, but remains nearly the same for 4 Tx streams. In contrast, higher distortion is observed when more Tx streams are used.

SNR(dB)	BPSK(1/2)	QPSK(1/2)	16-QAM(1/2)
4	1.15(0.56)	3.73(2.98)	48.06(48.89)
8	0.09(0)	0.98(0.52)	36.61(32.01)
12	0(0)	0(0)	7.41(5.52)
16	0(0)	0(0)	0.92(0.28)

Table 2: BER of emulated data payloads using 3 Tx. Numbers in brackets are BER of commodity WiFi.

As shown in Table 2, when 3 Tx streams are used, the extra SNR required to achieve 1% BER can be constrained within 0.9 dB, leading to a 18.2% improvement of emulation performance. However, Table 3 shows that using 4 Tx streams bring very little further improvement in performance. These results demonstrate that, in most practical cases, using 3 MIMO Tx streams would be sufficient to maximize the performance of emulation without introducing excessive signal distortion or computational overhead.

SNR(dB)	BPSK(1/2)	QPSK(1/2)	16-QAM(1/2)
4	1.14(0.56)	3.71(2.98)	48.36(48.89)
8	0.09(0)	0.97(0.52)	36.22(32.01)
12	0(0)	0(0)	7.48(5.52)
16	0(0)	0(0)	0.94(0.28)

Table 3: BER of emulated data payloads using 4 Tx. Numbers in brackets are BER of commodity WiFi.

9.5 Computation and Storage Overhead

We evaluate the computation and storage overhead in TransFi, with different sizes of Trellis diagram segments being used. Results in Figure 26 show that, when computing a MAC payload of 12,960 bits (10 symbols), dividing the Trellis diagram to segments of 30 bits could constrain the storage space used within 1.2 MB, and further reducing the segment size to 10 bits results in storage usage of 30.976 KB. Such reduction of segment size results in lower computation time, but further reducing the segment size increases the computing time due to frequent backtracing as described in Section 5.2.

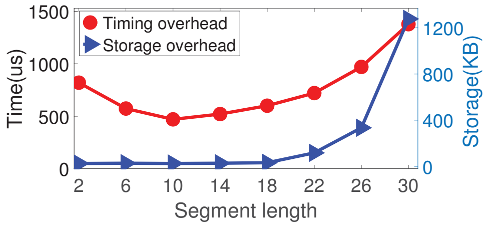


Figure 26: Computation and storage overhead with different sizes of Trellis diagram segments

In Figure 27, we compared the run-time computing time of emulation with the time gap between commodity PHY frames with the corresponding data payload sizes. In all different sizes of data payloads, TransFi's run-time computing time remains shorter than the time gap between WiFi PHY frames. These results demonstrate that TransFi can reliably support real-time communication in practice.

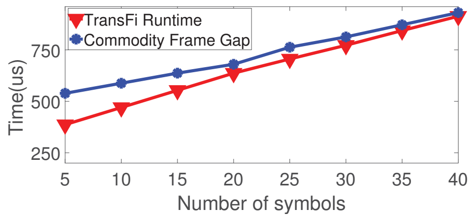


Figure 27: TransFi's run-time computing time compared to time intervals between WiFi PHY frames

10 EVALUATION ON FREQUENCY-DOMAIN EMULATION

To evaluate TransFi's performance on frequency-domain emulation, we examine TransFi's accuracy in emulating fine-grained and flexible spectrum usages, including FSA [63] and OFDMA [23, 47, 58]. In all experiments, we use two MIMO Tx streams for emulation.

10.1 Emulating Narrowband Signals in [63]

To examine TransFi's performance on emulating the narrowband signals being proposed in FSA [63], we use TransFi to migrate the time-domain signal up-sampling from the PHY layer to the MAC layer, by manipulating the MAC payload correspondingly. More specifically, narrowband signals that occupy 5 MHz, 10 MHz, and 15 MHz bandwidth in a 20 MHz channel are emulated and examined.

As shown in Figure 28(a), TransFi achieves above 94% in detecting the emulated signals under different channel SNRs, which is nearly the same as the performance achieved in [63] over custom hardware. Moreover, Figure 28(b) shows that no matter how the bandwidth of emulated signal varies, the detection rate is always above 95%.

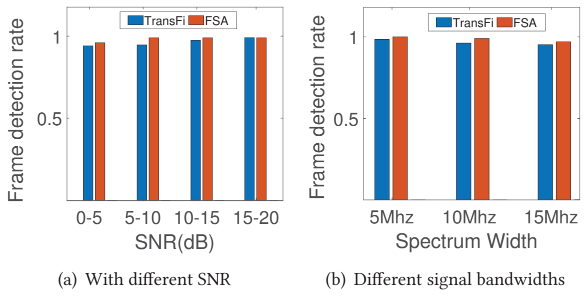


Figure 28: Accuracy of emulation for [63]

10.2 Emulating OFDMA Signals

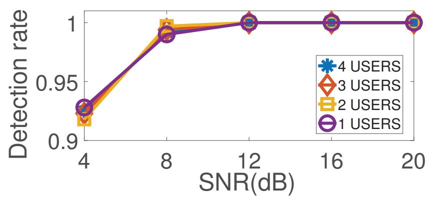
To examine TransFi's performance on emulating OFDMA, we use TransFi to emulate a wireless access point that operates OFDMA and concurrently transmits data to multiple users in the same channel. We distribute the 45 available data subcarriers defined in 802.11n to a maximum of 4 users, and emulate OFDMA frames for each user.

SNR	User 1	User 2	User 3	User 4
2	7.22(6.01)	6.49(5.11)	7.36(6.59)	6.56(5.32)
4	4.48(3.51)	3.18(2.19)	4.83(3.74)	4.24(3.30)
6	2.35(1.64)	1.73(1.02)	2.38(1.63)	1.70(1.09)
8	1.06(0.68)	0.75(0.37)	1.08(0.69)	0.80(0.35)
10	0.52(0.32)	0.22(0.09)	0.64(0.31)	0.21(0.09)
12	0.15(0.02)	0.06(0)	0.18(0.04)	0.08(0)

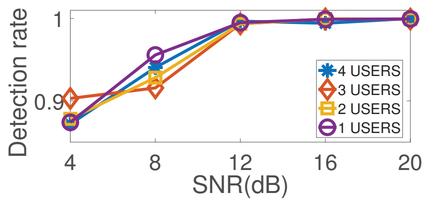
Table 4: Emulating 4 OFDMA users with different numbers of assigned subcarriers. Numbers in parentheses are BER in commodity OFDMA systems under the same condition.

First, we distribute these data subcarriers evenly. As shown in Figure 29(a), no matter how many users are involved, the emulated OFDMA frames can be detected with >92% accuracy. We further evaluate the chance of correct subcarrier allocation, by decoding the emulated RA preamble at each user. Figure 29(b) shows that more than 89% of emulated RA preambles can be correctly decoded. Finally, we compared the BER of decoding data payloads from emulated OFDMA frames, and Figure 29(c) shows that only extra 1.1dB, 1.6dB, and 2.8dB of channel SNR are needed for correct decoding with BPSK, QPSK and 16-QAM modulation.

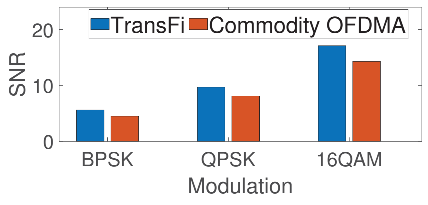
Further, we examine TransFi's performance when each user is assigned with a different number of subcarriers: 5, 17, 4 and 19 data subcarriers are assigned to 4 users, respectively. As shown in Table



(a) OFDMA frame detection accuracy



(b) Spectrum resources correctly allocated rate



(c) Channel SNR required to achieve 1% BER

Figure 29: Reception and decoding the emulated OFDMA data frames

4, at most 1.7 dB of extra SNR is needed for emulated data payloads to achieve the same BER as that in commodity OFDMA systems.

11 RELATED WORK

Software Defined Radios (SDR). SDRs allow flexible reconfiguration of the wireless hardware using software methods. For example, dynamic spectrum access can be implemented on SDRs by flexibly assigning different priorities of spectrum access to users [66]. SDRs can also be used to develop specialized gateways [30, 31, 62] or enable new methods of spectrum use [7, 8, 10, 25, 41]. However, most of the existing SDR designs and implementations require specialized hardware (e.g., FPGA chips) and hence cannot be adopted by commodity wireless devices. Instead, TransFi allows commodity wireless devices to adopt new methods of flexible spectrum usage, and hence have better applicability than SDR in practical systems.

Channel and source coding. Channel coding is widely used in wireless communication systems, but most existing coding methods focus on improving the link throughput [22] or correcting transmission errors [27, 51, 53, 57]. TransFi, although can be generally considered as a special type of source coding [16], has a completely different but more generic objective to produce custom wireless signals through appropriate encoding of commodity signals.

Packet emulation and injection. The design of reverse MAC payload computation in TransFi is partially motivated by the existing work on packet emulation, which has been used in CTC to approximate wireless signals among WiFi [20, 33, 34, 59], ZigBee [26, 60], Bluetooth [32, 38], LoRa [18, 37] and LTE [17, 36]. However, most of the packets being emulated are limited to network control (e.g., CSMA frames for collision avoidance [33]) or beacon frames (e.g., Bluetooth beacons [38]), which have fixed contents and are transmitted with low-order modulation (e.g., BPSK [34]) and low speed (e.g., kilobytes per second [37]). These existing techniques, hence, have limited expandability to custom wireless PHY or capability of achieving high-speed transmission of emulated data payloads, due to their coarse-grained emulation. TransFi, in contrast, is able to fundamentally improve such granularity of emulation.

Other existing work allows direct injection of arbitrary I/Q samples to WiFi PHY [44, 45]. However, they build on reverse engineering over specific WiFi chip models with known internal structures, and cannot be generically applied to commodity wireless devices. The payload injection, meanwhile, is usually expensive and hence limited to small control packets rather than random data payloads.

12 DISCUSSIONS

Scope of TransFi emulation. In theory, since any wireless signal can be mapped to the complex plane with the specific amplitude and phase, it can be emulated by TransFi within the WiFi frequency band, and its accuracy is jointly determined by the number of WiFi MIMO streams, modulation and the wireless channel condition.

One limitation is the granularity of frequency-division multiplexing, as finer multiplexing divides the frequency band into more but smaller sub-bands. For example, OFDM subcarrier spacing in 4G LTE and 5G can be as small as 15kHz, resulting in 20x finer multiplexing than that in 802.11n. Signals in extra subcarriers do not correspond to any producible wireless signals in commodity WiFi, and are hence difficult to be emulated. However, finer multiplexing requires a better but more expensive baseband processor and RF frontend. Hence, it is currently limited to being used in cellular networks. Further expanding TransFi to cellular networks will be our future work.

Reduction of spectrum efficiency. Since TransFi adds redundant data bits into MIMO data streams and mixes the transmitted wireless signals in multiple MIMO streams on the air, it emulates custom wireless PHY at the cost of reduced efficiency of spectrum utilization. However, as discussed in Section 3.3, such reduction could be effectively constrained within 20% by using higher code rates in commodity WiFi. On the other hand, the benefits of enabling custom wireless PHY on commodity devices could outweigh such reduction in many scenarios. For example, in highly congested wireless channels, adopting new PHY preambles [3, 35, 46] could help avoid interference and make crucial performance improvements in delay-sensitive applications.

Applicability to different WiFi hardware. Although our implementation is on a specific WiFi hardware model (Atheros AR9580), it can be applied to other WiFi hardware as long as they follow standard WiFi PHY design shown in Figure 16. The scrambler seed implementation could vary among different hardware, but most designs are simpler than the Atheros WiFi adapter. For example, a fixed scrambler seed is used in the RealTek WiFi hardware.

13 CONCLUSION

In this paper, we present TransFi to adopt custom wireless PHY designs on commodity devices without hardware modification. By mixing multiple MIMO streams on the air, TransFi can achieve real-time emulation of custom PHY signals from commodity WiFi transmitters with >90% emulation accuracy in different system settings and channel conditions.

ACKNOWLEDGMENTS

We thank our shepherd, Kate Ching-Ju Lin, and the anonymous reviewers for their comments and feedback. This work was supported in part by the National Science Foundation (NSF) under grant number CNS-1812407, CNS-2029520 and IIS-1956002.

REFERENCES

[1] Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. *IEEE Standard 802.11-2012*, March 2012.

[2] M. S. Afaqui, E. Garcia-Villegas, and E. Lopez-Aguilera. Ieee 802.11 ax: Challenges and requirements for future high efficiency wifi. *IEEE Wireless Communications*, 24(3):130–137, 2016.

[3] B. Bellalta. Ieee 802.11 ax: High-efficiency wlans. *IEEE Wireless Communications*, 23(1):38–46, 2016.

[4] B. Bloessl, M. Segata, C. Sommer, and F. Dressler. An IEEE 802.11a/g/p OFDM Receiver for GNU Radio. In *The 2nd ACM SIGCOMM Workshop of Software Radio Implementation Forum (SRIF)*, pages 9–16, Hong Kong, China, August 2013. ACM.

[5] T. E. Bogale and L. B. Le. Massive mimo and mmwave for 5g wireless hetnet: Potential benefits and challenges. *IEEE Vehicular Technology Magazine*, 11(1):64–75, 2016.

[6] A. Capponi, C. Fiandrino, B. Kantarci, L. Foschini, D. Kliazovich, and P. Bouvry. A survey on mobile crowdsensing systems: Challenges, solutions, and opportunities. *IEEE communications surveys & tutorials*, 21(3):2419–2465, 2019.

[7] K. Chebrolu and A. Dhekne. ESense: communication through energy sensing. In *Proceedings of ACM MobiCom*. ACM, 2009.

[8] R. Chen and W. Gao. Enabling cross-technology coexistence for extremely weak wireless devices. In *IEEE INFOCOM*, 2019.

[9] H. Cheon and D. Hong. Effect of channel estimation error in ofdm-based wlan. *IEEE Communications Letters*, 6(5):190–192, 2002.

[10] Z. Chi, Y. Li, H. Sun, Y. Yao, Z. Lu, and T. Zhu. B2w2: N-way concurrent communication for IoT devices. In *Proceedings of ACM SenSys*, 2016.

[11] H.-W. Cho and K. G. Shin. Bluefi: bluetooth over wifi. In *Proceedings of ACM SIGCOMM*, pages 475–487, 2021.

[12] S. Coleri, M. Ergen, A. Puri, and A. Bahai. Channel estimation techniques based on pilot arrangement in ofdm systems. *IEEE Transactions on broadcasting*, 48(3):223–229, 2002.

[13] M. Communications. 802.11 reference design for warp v3. In *warpproject.org/trac/wiki/802.11*, 2013.

[14] E. Dahlman, S. Parkvall, and J. Skold. *4G: LTE/LTE-advanced for mobile broadband*. Academic press, 2013.

[15] P. Eldad and S. Robert. *Next Generation Wireless LANs Throughput robustness and reliability in 802.11n*. Cambridge University Press, 2008.

[16] J. Fowler and B. Pesquet-Popescu. An overview on wavelets in source coding, communications, and networks. *EURASIP Journal on Image and Video Processing*, 2007:1–27, 2007.

[17] P. Gawłowicz, A. Zubow, S. Bayhan, and A. Wolisz. Ofdmfi: Enabling cross-technology communication between LTE-U/LAA and wifi. *arXiv preprint arXiv:1912.04093*, 2019.

[18] P. Gawlowicz, A. Zubow, and F. Dressler. Wi-lo: Emulating LoRa using COTS wifi. *arXiv preprint arXiv:2105.04998*, 2021.

[19] J. Gold. 2.4ghz is a headache for wi-fi users, and it is here to stay. Network World. https://www.networkworld.com/article/3192984/2-4ghz-is-a-headache-for-wi-fi-users-and-it-s-here-to-stay.html.

[20] X. Guo, Y. He, J. Zhang, and H. Jiang. Wide: physical-level CTC via digital emulation. In *2019 18th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, pages 49–60. IEEE, 2019.

[21] J. Hagenauer and P. Hoeher. A viterbi algorithm with soft-decision outputs and its applications. In *1989 IEEE Global Telecommunications Conference and Exhibition'Communications Technology for the 1990s and Beyond'*, pages 1680–1686. IEEE, 1989.

[22] B. Hochwald and K. Zeger. Tradeoff between source and channel coding. *IEEE Transactions on Information Theory*, 43(5):1412–1424, 1997.

[23] H. Holma and A. Toskala. *LTE for UMTS: OFDMA and SC-FDMA based radio access*. John Wiley & Sons, 2009.

[24] W. Jeong, J. Jung, Y. Wang, S. Wang, S. Yang, Q. Yan, Y. Yi, and S. M. Kim. Sdr receiver using commodity wifi via physical-layer signal reconstruction. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*, pages 1–14, 2020.

[25] W. Jiang, Z. Yin, S. M. Kim, and T. He. Side channel communication over wireless traffic: A ctc design. In *Proceedings of the 14th ACM Conference on Embedded Network Sensor Systems*, pages 346–347, 2016.

[26] W. Jiang, Z. Yin, R. Liu, Z. Li, S. M. Kim, and T. He. Bluebee: a 10,000x faster cross-technology communication via phy emulation. In *Proceedings of ACM SenSys*. ACM, 2017.

[27] R. Johannesson and K. S. Zigangirov. *Fundamentals of convolutional coding*. John Wiley & Sons, 2015.

[28] D. Kivanc, G. Li, and H. Liu. Computationally efficient bandwidth allocation and power control for ofdma. *IEEE transactions on wireless communications*, 2(6):1150–1158, 2003.

[29] K. F. Lee and D. B. Williams. A space-time coded transmitter diversity technique for frequency selective fading channels. In *Proceedings of the 2000 IEEE Sensor Array and Multichannel Signal Processing Workshop. SAM 2000 (Cat. No. 00EX410)*, pages 149–152. IEEE, 2000.

[30] Y. Li, Z. Chi, X. Liu, and T. Zhu. Chiron: Concurrent high throughput communication for iot devices. In *Proceedings of ACM MobiSys*, 2018.

[31] Y. Li, Z. Chi, X. Liu, and T. Zhu. Passive-zigbee: Enabling zigbee communication in iot networks with 1000x+ less power consumption. In *Proceedings of the 16th ACM Conference on Embedded Networked Sensor Systems*, pages 159–171, 2018.

[32] Z. Li and Y. Chen. Bluefi: Physical-layer cross-technology communication from bluetooth to wifi. In *2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS)*, pages 399–409. IEEE, 2020.

[33] Z. Li and T. He. Webee: Physical-layer cross-technology communication via emulation. In *Proceedings of ACM MobiCom*. ACM, 2017.

[34] Z. Li and T. He. Longbee: Enabling long-range cross-technology communication. In *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*, pages 162–170. IEEE, 2018.

[35] X. Lin, A. Adhikary, and Y.-P. E. Wang. Random access preamble design and detection for 3gpp narrowband iot systems. *IEEE Wireless Communications Letters*, 5(6):640–643, 2016.

[36] R. Liu, Z. Yin, W. Jiang, and T. He. Lte2b: Time-domain cross-technology emulation under lte constraints. In *Proceedings of the 17th Conference on Embedded Networked Sensor Systems*, pages 179–191, 2019.

[37] R. Liu, Z. Yin, W. Jiang, and T. He. Xfi: Cross-technology iot data collection via commodity wifi. In *2020 IEEE 28th International Conference on Network Protocols (ICNP)*, pages 1–11. IEEE, 2020.

[38] R. Liu, Z. Yin, W. Jiang, and T. He. Wibeacon: expanding BLE location-based services via wifi. In *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*, pages 83–96, 2021.

[39] H.-L. Lou. Implementing the viterbi algorithm. *IEEE Signal processing magazine*, 12(5):42–52, 1995.

[40] F. Lu, P. Ling, G. M. Voelker, and A. C. Snoeren. Enfold: downclocking ofdm in wifi. In *Proceedings of the 20th annual international conference on Mobile computing and networking*, pages 129–140, 2014.

[41] H. Lu, R. Chen, and W. Gao. Easypass: combating iot delay with multiple access wireless side channels. In *Proceedings of the 15th International Conference on Emerging Networking Experiments And Technologies*, pages 186–199, 2019.

[42] H. MacLeod, C. Loadman, and Z. Chen. Experimental studies of the 2.4-ghz ism wireless indoor channel. In *3rd Annual Communication Networks and Services Research Conference (CNSR'05)*, pages 63–68. IEEE, 2005.

[43] H. Rahul, H. Hassanieh, and D. Katabi. Sourcesync: a distributed wireless architecture for exploiting sender diversity. *ACM SIGCOMM Computer Communication Review*, 41(4):171–182, 2011.

[44] M. Schulz, J. Link, F. Gringoli, and M. Hollick. Shadow wi-fi: Teaching smartphones to transmit raw signals and to extract channel state information to implement practical covert channels over wi-fi. In *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services*, pages 256–268, 2018.

[45] M. Schulz, D. Wegemer, and M. Hollick. Nexmon: Build your own wi-fi testbeds with low-level mac and phy-access using firmware patches on off-the-shelf mobile devices. In *Proceedings of the 11th Workshop on Wireless Network Testbeds, Experimental evaluation & CHaracterization*, pages 59–66, 2017.

[46] S. Sen, R. R. Choudhury, and S. Nelakuditi. CSMA/CN: Carrier sense multiple access with collision notification. *IEEE/ACM Transactions on Networking (ToN)*, 20(2):544–556, 2012.

[47] K. Seong, M. Mohseni, and J. M. Cioffi. Optimal resource allocation for ofdma downlink systems. In *2006 IEEE international symposium on information theory*, pages 1394–1398. IEEE, 2006.

[48] C. Sobin. A survey on architecture, protocols and challenges in iot. *Wireless Personal Communications*, 112(3):1383–1429, 2020.

[49] H. Steendam and M. Moeneclaey. Analysis and optimization of the performance of ofdm on frequency-selective time-selective fading channels. *IEEE Transactions on Communications*, 47(12):1811–1819, 1999.

[50] G. L. Stuber, J. R. Barry, S. W. Mclaughlin, Y. Li, M. A. Ingram, and T. G. Pratt. Broadband mimo-ofdm wireless communications. *Proceedings of the IEEE*, 92(2):271–294, 2004.

[51] V. Tarokh, H. Jafarkhani, and A. R. Calderbank. Space-time block coding for wireless communications: Performance results. *IEEE Journal on selected areas in communications*, 17(3):451–460, 1999.

[52] J. Terry and J. Heiskala. *OFDM wireless LANs: A theoretical and practical guide*. Sams publishing, 2002.

[53] J. Thorpe. Low-density parity-check (ldpc) codes constructed from protographs. *IPN progress report*, 42(154):42–154, 2003.

[54] J.-J. Van De Beek, O. Edfors, M. Sandell, S. K. Wilson, and P. O. Borjesson. On channel estimation in ofdm systems. In *1995 IEEE 45th Vehicular Technology Conference. Countdown to the Wireless Twenty-First Century*, volume 2, pages 815–819. IEEE, 1995.

[55] R. Van Nee, V. Jones, G. Awater, A. Van Zelst, J. Gardner, and G. Steele. The 802.11n MIMO-OFDM standard for wireless LAN and beyond. *Wireless Personal Communications*, 37(3-4):445–453, 2006.

[56] D. Vassis, G. Kormentzas, A. Rouskas, and I. Maglogiannis. The ieee 802.11 g standard for high data rate wlans. *IEEE network*, 19(3):21–26, 2005.

[57] B. Vucetic and J. Yuan. *Turbo codes: principles and applications*, volume 559. Springer Science & Business Media, 2012.
[58] K. Wang and K. Psounis. Scheduling and resource allocation in 802.11 ax. In *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*, pages 279–287. IEEE, 2018.
[59] S. Wang, W. Jeong, J. Jung, and S. M. Kim. X-mimo: cross-technology multi-user mimo. In *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*, pages 218–231, 2020.
[60] S. Wang, S. M. Kim, and T. He. Symbol-level cross-technology communication via payload encoding. In *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*, pages 500–510. IEEE, 2018.
[61] S. Wu and Y. Bar-Ness. A phase noise suppression algorithm for ofdm-based wlans. *IEEE Communications Letters*, 6(12):535–537, 2002.
[62] Y. Yubo, Y. Panlong, L. Xiangyang, T. Yue, Z. Lan, and Y. Lizhao. Zimo: Building cross-technology mimo to harmonize zigbee smog with wifi flash without intervention. In *In Proceedings of ACM MobiCom*, 2013.
[63] S. Yun, D. Kim, and L. Qiu. Fine-grained spectrum adaptation in wifi networks. In *Proceedings of the 19th annual international conference on Mobile computing & networking*, pages 327–338, 2013.
[64] F. Zafari, A. Gkelias, and K. K. Leung. A survey of indoor localization systems and technologies. *IEEE Communications Surveys & Tutorials*, 21(3):2568–2599, 2019.
[65] X. Zhang and K. G. Shin. E-MiLi: energy-minimizing idle listening in wireless networks. In *ACM MobiCom*, 2011.
[66] Q. Zhao and B. M. Sadler. A survey of dynamic spectrum access. *IEEE signal processing magazine*, 24(3):79–89, 2007.