SWiFi: An Open Source SDR for Wi-Fi Networks High Order Modulation Analysis

Triet D. Vo-Huu
vohuudtr@ccs.neu.edu

Tien D. Vo-Huu
tienvh@ccs.neu.edu

Guevara Noubir
noubir@ccs.neu.edu

College of Computer and Information Science
Northeastern University
Boston, MA 02115

ABSTRACT

We introduce SWiFi, a complete Open Source software defined radio stack for Wi-Fi networks analysis and experimentation. Our implementation can decode packets for all Modulation Coding Schemes reaching 54Mbps (64-QAM with 3/4 coding rate). SWiFi runs on the popular Ettus/NI N210 but also on the low-cost and small form-factor HackRF. Beyond a careful and comprehensive implementation of the standard, SWiFi incorporates novel frequency offset and frequency domain equalization techniques to overcome the limitations of the SDR RF Front End. We rigorously evaluate the performance of the SWiFi receiver demonstrating a performance that closely matches or outperforms several commercial Wi-Fi cards with the latest IEEE802.11 chipsets. We also demonstrate the potential of SWiFi to support research on Wi-Fi networks and devices characterization spanning the MAC layers (e.g., SIFS/CSMA/Backoff), and Link Layer (e.g., rate adaptation).

1. INTRODUCTION

The U.S. economy largely depends on wireless communication technologies to fuel its growth. As a striking example, Apple Inc. is now the largest publicly traded company in the world by market capitalization, thanks to its mobile smartphones and tablet devices.

In this context, Wi-Fi (IEEE 802.11), is emerging as the primary medium for wireless Internet access. Cellular carriers are increasingly offloading their traffic to Wi-Fi Access Points (APs) to overcome capacity challenges, limited RF spectrum availability, cost of deployment, and keep up with the traffic demands driven by user generated content. Wi-Fi Offloading is facilitated by 3GPP standards for Non-3GPP Access Networks Discovery and Roaming [1], IETF seamless USIM-based strong authentication and secure communication protocols such as EAP-SIM/AKA [19], [13], and IEEE seamless handover and authentication protocols across networks IEEE802.11u [20]. Studies forecast a sustained 50% yearly growth in Wi-Fi offloading for many years to come [32], [37], [31]. This trend is paved by the increasing deployments of Hotspot 2.0 (HS2) Access Points enabled by seamless handover across networks implementing the IEEE802.11u amendment [20]. Moreover, manufacturers of laptops and streaming devices, such as the Apple MacBook Pro and the Roku streaming player, are removing Ethernet ports and entirely relying on Wi-Fi, and several new variants of Wi-Fi are being developed to suit different environments (e.g., IEEE 802.11p for vehicular networking and IEEE 802.11af for TV white spaces).

Despite the central role of Wi-Fi in today's wireless communications systems, there is still no platform available to the research community to analyze Wi-Fi networks from the physical layer to the network layer with the precision that it deserves. Over the last few years researchers made great progress in characterizing Wi-Fi networks by extrapolating information provided by the drivers of commodity Wi-Fi cards. These systems supported by clever algorithms were able to infer surprising properties, sometimes in relatively larger scale setups [34], [35], [29]. However, they remain intrinsically limited by the information provided by commodity cards drivers (e.g., RSSI per carrier with a ms granularity). Precise information about the timing of packets, backoff, collisions, packet capture effects, hidden terminals, fine grain state of the channel, interference from smart misbehaving devices and stealthy jamming remain mostly invisible through the driver's API eye.

While focusing on commodity Wi-Fi cards for analyzing the RF spectrum was partially a deliberate choice because of their ubiquity, low-cost, ease of programming, and large scale deployments, other researchers encouraged by the availability of software defined radio platforms, endeavored in the task of developing a Wi-Fi protocol stack from the physical to the link layer. Since the open source implementation of IEEE802.11 by BBN (operating at 1 & 2Mbps with downsampled baseband signals to 4MHz) [3], to the Utah University FPGA-assisted implementation [12], to the more recent attempts [5], [6]. As of the time of this writing, we are not aware of any work other than the gr-ieee802-11 GNU Radio implementation in [6], that operates up to 54 Mbps.

Motivated by the potential of a Wi-Fi SDR in enabling wireless research, and the expanding offerings of hardware SDR platforms, we developed SWiFi. We note that SWiFi is limited by the hardware SDR delay, and does not run within the SDR FPGA. Therefore it cannot send ACK packets within the SIFS interval. It is however designed with the goal to enable the analysis of Wi-Fi networks in ways not possible before. Furthermore, despite the timing constraints that prevents it from engaging in a unicast communication with other devices, it can still send broadcast packets, spoof, and decode any IEEE802.11abg packets.

Beyond the substantial amount of work spent in carefully implementing and testing all the mechanisms of the abg physical layer, the most challenging component was to devise novel techniques for Frequency Offset correction, and Frequency Domain Equalization that overcome the limited quality wideband RF Front End of SDR platforms. We successfully tested our implementation on two types of SDR platforms the mid-range USRP 2 & N210, and the low cost $275 HackRF [17]. SWiFi on these platforms compares favorably with commercial cards. However, as we will discuss in the evaluation section, the HackRF requires a firmware reprogramming to sustain the Wi-Fi necessary sampling rate (20Msps).

We believe that SWiFi will enable research at a new level. To support our claims, we developed some preliminary companion tools that enable the analysis of 802.11 MAC timing (SIFS), and 802.11 links (rate adaptation). We plan to make SWiFi open source, along with its companion tools, to enable the research community to use it in their respective projects but also to extend it. For instance, we will create an ORBIT image that allow other researchers to easily and reproducibly test the performance of SWiFi.

Our main contribution can be summarized as (1) new algorithms for frequency domain equalization, (2) a rigorous and comprehensive evaluation demonstrating performance at least similar to commercial Wi-Fi cards, (3) preliminary tools for analyzing Wi-Fi networks using SWiFi.

The paper is organized as follows. We first provide an overview of OFDM principles and their instantiation in IEEE 802.11ag, followed by a description of our transmitter and receiver designs. We provide a survey of existing frequency domain equalization techniques and their limitations before introducing and comparing to our own mechanisms. We then present our evaluation testbeds and methodology and report on the comparison of SWiFi with commercial Wi-Fi cards. We finally, provide some results from our preliminary companion tools.

2. OVERVIEW OF OFDM

We start our discussion with a brief overview of the OFDM principle. In later sections, we then describe in detail our GNU Radio implementation of SWiFi-OFDM Transceiver with a focus on the receiver part.

The principle of Orthogonal Frequency Division Multiplexing (OFDM) is to transmit data over multiple subcarriers in a way that the subcarriers can overlap, but create no interference to each other. This feature renders OFDM as the core technique for boosting the performance of today's wireless systems in wideband communications such as 4G LTE/WiMax & IEEE802.11agn. OFDM's success is proven with the dense deployment of Wi-Fi networks today.

Let $X_1, \ldots, X_N$ be the data symbols that will be transmitted at each time period T. We view X_k as values in the frequency domain, where each symbol X_k corresponds to the k-th subcarrier. The chunk $\mathbf{X} = \{X_1, \ldots, X_N\}$ forms an OFDM symbol in the frequency domain. To transmit the data, every X_k symbol is transformed to the time-domain signal $x_n^{(k)} = X_k e^{j2\pi kn/N}$ corresponding to the k-th subcarrier, where n indicates the n-th slot time in an OFDM symbol period. The sum of all subcarriers' signal in the n-th slot is represented by

$$x_n = \sum_{k=1}^{N} x_n^{(k)} = \sum_{k=1}^{N} X_k e^{j2\pi kn/N}. \tag{1}$$

The time-domain OFDM symbol, denoted by $\mathbf{x}$, includes N samples: $\mathbf{x} = \{x_1, \ldots, x_N\}$. Since $\sum_{n=1}^{N} x_n^{(k)} (x_n^{(m)})^* = 0$ for $k \neq m$ (where * denotes the complex conjugate), N subcarriers are orthogonal, resulting in no inter-carrier interference. This is in contrast with non-orthogonal FDM system, where inter-channel spacing must be reserved in order to avoid mutual interference. The computation in Equation (1) can be efficiently performed by Inverse Fast Fourier Transform (IFFT) technique. At the receiver, the original data symbols are recovered by an FFT operation on the time-domain signal x_n, specifically

$$X_k = \frac{1}{N} \sum_{n=1}^{N} x_n e^{-j2\pi nk/N}.$$

The orthogonality is the key feature of OFDM systems. On one hand, orthogonality allows overlapping subcarriers, therefore increasing the channel efficiency. On the other hand, loss of orthogonality dramatically affects the system performance. To enable practical OFDM receivers, IEEE 802.11ag specifies $N = 64$ subcarriers for 20MHz bandwidth, among which only $N_D = 48$ subcarriers are used for data, while other $N_P = 4$ pilot subcarriers are placed equally in between data subcarriers to assist the receiver's channel estimation, and the rest $N_0 = 12$ subcarriers are left unused (null carriers) for avoiding DC offset and inter-channel interference (See Figure 2). During the signal propagation in the environment, blocking and reflecting objects can create multiple copies of the transmitted signal and distort the subcarrier orthogonality. To reduce the multipath effect, a guard interval with cyclic prefix is introduced at the beginning of each OFDM symbol. In particular, the last $N_G = 16$ symbols $x_{N-N_G+1}, \ldots, x_N$ of the time-domain OFDM symbol $\mathbf{x}$ are copied and put before $\mathbf{x}$ to result in $N + N_G = 80$ samples $\{x_{N-N_G+1}, \ldots, x_N, x_1, \ldots, x_N\}$ to be transmitted per OFDM symbol.

The concrete estimation, equalization and decoding techniques for the receiver, however, are left to the specific im-

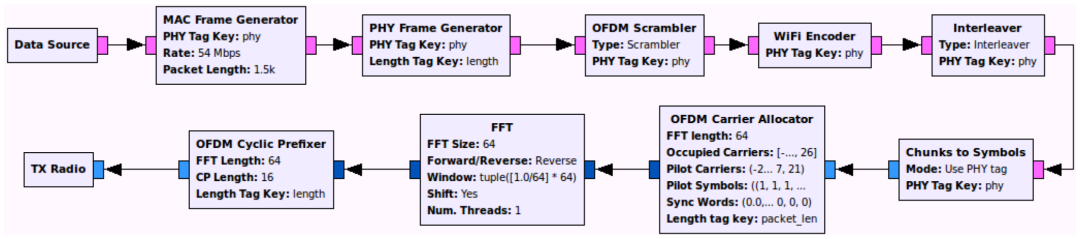


Figure 1: Transmitter OFDM GNU Radio block diagram

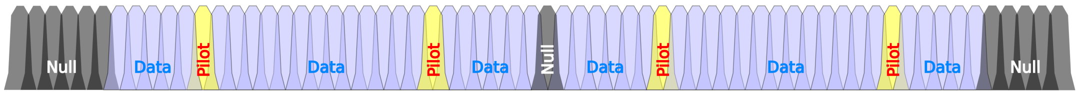


Figure 2: IEEE 802.11 OFDM subcarriers mapping.

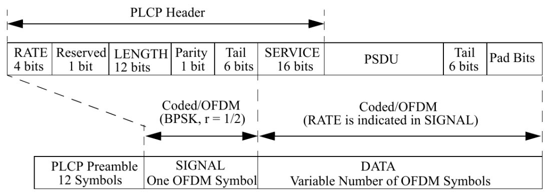


Figure 3: OFDM PHY frame format [21].

plementation by the chipset manufacturers. In the following subsections, we describe our specific algorithms with comparison to the state-of-the-art techniques. First, we present the IEEE 802.11 OFDM frame structure and the transmit chain of SWiFi.

3. TRANSMITTER DESIGN

We develop the SWiFi-OFDM Transmitter using the built-in GNU Radio blocks of FFT, Carrier Allocator and Cyclic Prefixer along with our own blocks for generating and encoding the OFDM PHY frame, as depicted with GNU Radio Companion – a GUI signal processing flowgraph development environment – in Figure 1.

PHY Frame Structure: The time-domain OFDM PHY frame, illustrated in Figure 3, consists of OFDM header and payload, which are prepended by a special training sequence of short and long preambles. The short preamble is composed of 10 repeated symbols each of $N_{LP} = 16$ samples, while the long preamble contains 2.5 repeated symbols each of $N_{SP} = 64$ samples. The resulting preamble duration is 320 samples, equal to the length of 4 OFDM symbols. The repeated patterns present in both short and long symbols allow the receiver to locate the frames inside the received stream and to perform frequency offset correction (described in Section 4.1).

Following the preamble are the SIGNAL field, which specifies the rate and size of the payload. The DATA field contains a 16-bit SERVICE subfield used to synchronize the receiver and transmitter's scrambling seed. The payload, tail and paddings are placed in the rest of the DATA field. Since the encoding process is different on SIGNAL and DATA fields, our PHY Frame Generator employs GNU Radio tags to inform the scrambling, encoding and interleaving blocks of the rate and length corresponding to each part.

Scrambling: For each raw PHY frame, only the DATA field is scrambled by a synchronous (additive) scrambler, while the SIGNAL field is left untouched. The 802.11ag scrambler has 7 registers and requires both transmitter and receiver to synchronize on the same scrambling seed in order to correctly decode the packet. This is assisted by the transmitter by setting the first 7 bits of the SERVICE subfield to all zeros before scrambling. Since wrong detection of the seed results in an undecodable frame, it is important to properly receive the first OFDM symbol in the DATA field. We provide a more detailed discussion in later subsections on channel estimation and equalization.

Encoding: The encoding of the frame header and payload relies on a convolutional code of coding rate 1/2 defined by 802.11ag. While the header is always encoded at coding rate 1/2, the payload coding rate can be increased to 2/3 and 3/4 by puncturing, i.e., periodically omitting several bits in the encoded bit stream. The WiFi Encoder block comprises two sub-blocks for handling convolutional encoding and puncturing tasks.

Interleaving: The interleaving of encoded bits is performed within each OFDM symbol. To minimize the computation cost, we predefine interleaving tables for different modulation and coding rate accordingly to the rules by 802.11ag.

Modulation: Each OFDM symbol of interleaved bits is mapped to $N_D = 48$ data subcarriers by the modulation defined in the SIGNAL field. In 802.11ag, the same modulation is used across data subcarriers. Therefore, we employ only one common modulator to map serial bit stream to constellation symbols.

Pilot insertion: At this stage, symbols on data subcarriers are ready to be separated into groups with $N_P = 4$ pilot subcarriers inserted between them. The pilot symbols are defined as BPSK modulated values. The built-in GNU Radio

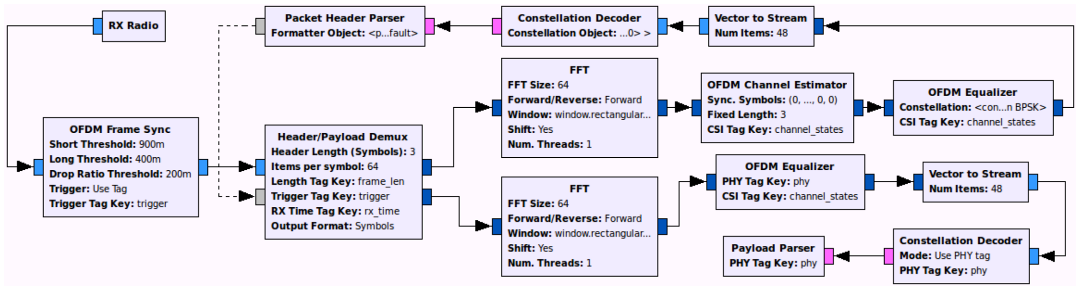


Figure 4: Receiver OFDM GNU Radio block diagram

Carrier Allocator block inserts not only the pilot but also null subcarriers into the carrier map to create 64-subcarrier OFDM symbols.

Generating time-domain signal: Finally, each OFDM symbol $\mathbf{X}$ is ready to be transformed into the time domain $\mathbf{x}$ by the use of FFT. The cyclic prefix of N_G symbols are prepended to the beginning of $\mathbf{x}$ by the GNU Radio Cyclic Prefixer block, resulting in $N + N_G = 80$ samples for every transmission of an OFDM symbol.

4. RECEIVER DESIGN

SWiFi-OFDM Receiver block diagram is shown in Figure 4, in which the core components are:

- The OFDM Frame Sync block is responsible for frequency offset estimation and frame synchronization. For each recognized frame, it sends the time-domain samples to the Header/Payload Demux, which removes the cyclic prefix on every OFDM symbol and demultiplexes to the header and payload receive chains, where the FFT blocks transform the samples back to the frequency domain for decoding and parsing.
- The OFDM Channel Estimator is only present on the header chain, since it performs an initial channel estimation based on the training sequence defined in the preamble. The initial channel state information (CSI) is crucial for demodulating the subcarriers.
- The OFDM Equalizer establishes the equalization process with initial CSI, then it handles the dynamic channel variations along the reception of the frame.

4.1 Frequency Offset Correction

Due to the common clock mismatch between the transmitter and receiver RF front ends, the original signal s_n is rotated at the receiver by an amount of $n\theta + \phi$, where θ represents the frequency offset between the transmitter and receiver, and ϕ denotes the unknown phase offset. In SWiFi, we compensate the frequency offset as a first step before any other signal processing tasks. At the same time of frequency offset estimation, a coarse detection of OFDM frames is also achieved. Our algorithm is based on Schmidt-Cox method [39], which utilizes the repeated pattern in short and long preamble symbols.

Coarse estimation: Let $\mathbf{p} = \{p_1, p_2, \ldots, p_L\}$ be a time-domain short symbol consisted of $L = 16$ samples defined in 802.11ag preamble. The principle of the method is based on the assumption that the frequency offset θ is relatively smaller than $2\pi/L$ and remains constant in the duration of an OFDM symbol, by which a phase difference of $L\theta$ will be observed between two consecutive preamble symbols. The limit $2\pi/L$ will be made clear shortly. At the receiver, we observe the received signal as

$$\{r_n\} = \ldots, \underbrace{\hat{p}_1, \ldots, \hat{p}_L, \hat{p}_{L+1}, \ldots, \hat{p}_{2L}, \ldots}_{\text{preamble}}, \underbrace{\ldots, \hat{x}_k, \ldots}_{\text{data}}$$

where $\hat{p}_k = p_{[k]_L} e^{j(k\theta+\phi)}$, $[k]_L = k \bmod L$, and $\hat{x}_k = x_k e^{j(k\theta+\phi)}$ denote the rotated version of preamble and data samples. We compute the correlation A_n between two consecutive chunks of L samples and the energy E_n of the current chunk:

$$A_n = \sum_{k=0}^{L-1} r_{n+k+L} r^*_{n+k}, \quad E_n = \sum_{k=0}^{L-1} |r_{n+k}|^2$$

The ratio $|A_n/E_n|$ determines whether r_n contains a preamble sample p_k for some k. Specifically, the preamble is found if $|A_n/E_n| \geq \alpha$, where the parameter $\alpha = 0.9$ is used in SWiFi implementation. At the same time of identifying the preamble, we observe that $A_n = \sum_{k=0}^{L-1} |p_{[n+k]_L}|^2 e^{jL\theta}$. Now, due to the assumption $\theta < 2\pi/L$, we have $\arg(A_n) = L\theta$, and obtain the estimate $\hat{\theta} = \frac{1}{L}\arg(A_n)$. Considering the Wi-Fi short preamble symbols with length $L = 16$, the maximum correctable frequency offset by SWiFi is $\pi/8 \approx 0.4$ rad/sample, which is acceptable for today's RF front ends.

We emphasize that in the GNU Radio platform, we implement the mechanism in a single block rather than using multi-threaded multiple blocks as the typical GNU Radio approach. Our solution is to maintain a state machine in order to control the estimation and correction logic more efficiently. In particular, we stop the computation for the estimation once the frame is detected, and resume it when there is an energy drop in the signal indicating the frame end. During the packet processing, small variations of the frequency offset are handled by the equalization which will be described in Section 4.2.

Fine estimation: After a coarse estimation based on the

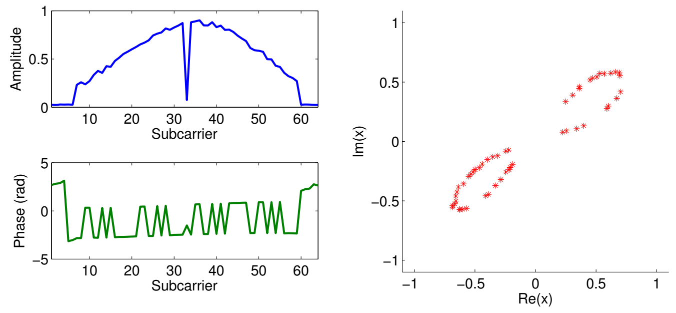


Figure 5: Analysis of BPSK-modulated SIGNAL field: 64 subcarriers experience different attenuations and phase offsets (on the left), resulting in the leaves-shape on the constellation (on the right).

short preamble, we repeat the same algorithm on the long preamble to compensate for any residual offset not detected by short symbols. Though more computation is required for working with the long preamble, it is only performed after the threshold for the coarse estimate is exceeded, thus only a slight overhead is introduced.

Frame synchronization: While the algorithm for frequency offset correction can detect the frame, it is ambiguous to locate exactly where the frame is started, as the ratio $|A_n/E_n|$ remains high in the duration of repeated symbols and gradually decreases when the preamble is passed. To precisely reveal the exact location of the first sample of the frame, we correlate the received samples (after frequency offset correction) with the time-domain long preamble to find the peak corresponding to the repeated pattern of the long preamble. This approach gives an accurate synchronization with the OFDM symbols, as also shown in [5].

4.2 Frequency Domain Equalization

Frequency offset correction in the time domain is not enough for a successful demodulation of the OFDM symbols, because in a wireless environment, the dynamic channel causes the subcarriers to experience different attenuations and phase offsets. Figure 5 shows the frequency-domain OFDM symbol corresponding to the SIGNAL field captured over the air. It can be seen that the subcarriers are distorted in both amplitude and phase, which result in the SIGNAL field's BPSK constellation points being rotated and deviated from their original points. More importantly, even though the frequency offset has been corrected in the time domain by the OFDM Frame Sync block, a small amount of frequency offset can still be observed in the frequency domain, which accumulate and rotate the symbols from their original locations.

While amplitude correction is not required for PSK modulations, it is crucial for demodulating QAM signals/ In the following, we review existing equalization methods for OFDM systems and present our new techniques that allow the SWiFi receiver to perform comparably to commodity Wi-Fi cards. We use similar notations introduced in Section 2 for our discussion, i.e., $\mathbf{X}$ denotes the frequency-domain transmitted

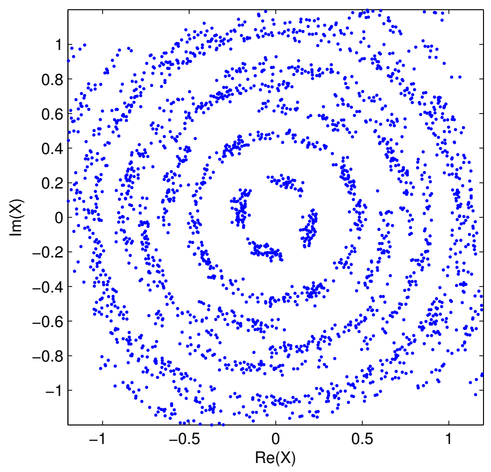


Figure 6: Phase tracking by LS Equalization results in rotated constellation.

OFDM symbol, $\mathbf{Y}$ the frequency-domain received OFDM symbol. For each frame, we index the OFDM symbols as follows: the two long preamble symbols are $\mathbf{X}_{-2}, \mathbf{X}_{-1}$, the SIGNAL field is $\mathbf{X}_0$, and the DATA field starts from $\mathbf{X}_k$, $k = 1, \ldots, n$.

Least Square (LS): The Least Square estimation is based on the long preamble symbols, namely the channel is estimated as

$$\mathbf{H} = \frac{1}{2}\left(\frac{\mathbf{Y}_{-2}}{\mathbf{X}_{-2}} + \frac{\mathbf{Y}_{-1}}{\mathbf{X}_{-1}}\right) \tag{2}$$

and is used to equalize the rest of all OFDM symbols. Due to frequency and phase offset occurring in the transmission, the LS method cannot keep track of the phase of the symbols over the frame duration. As a result, a rotated constellation can be observed as shown in Figure 6.

Linear Regression (LR): In the Linear Regression approach, the pilot symbols on 4 pilot subcarriers are used to infer, by linear regression, the phase offset of the subcarriers based on the observation that there is a constant phase offset from subcarrier to subcarrier within an OFDM symbol (as seen previously in Figure 5). To compensate the amplitude, a scaling factor is derived from the average amplitude of pilot subcarriers and it is used to restore the amplitude of data subcarriers. While this approach can adaptively recover the symbols modulated by PSK modulations [5], the QAM signal cannot be decoded[1]. This is explained by the fact that subcarriers are attenuated by different gains (See Figure 5), as a result, a common scaling factor does not properly recover the amplitudes for all subcarriers, which are required for amplitude-sensitive modulations like QAM.

Low Pass Interpolation (LPI): Based on the assumption that the received OFDM symbol have a sinc shape (e.g., Fig-

[1]The work in [5] has been extended in [6] to correct the issue.

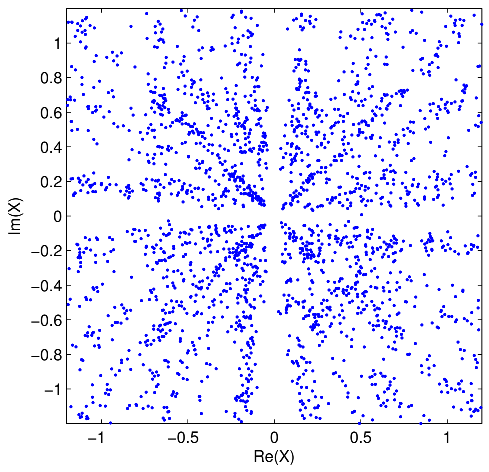


Figure 7: QAM symbol amplitudes are not recovered properly by LR Equalization.

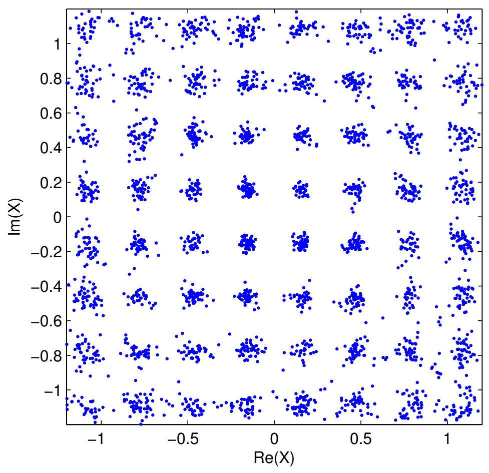


Figure 8: Effect of LPI equalization: phase and amplitudes are recover, but a significant amount of incorrect estimation is still present.

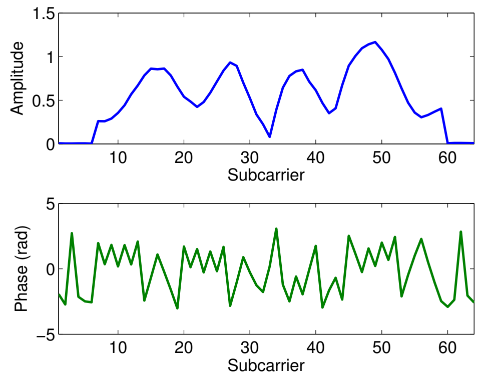


Figure 9: OFDM symbol is affected by unpredicted channel distortions .

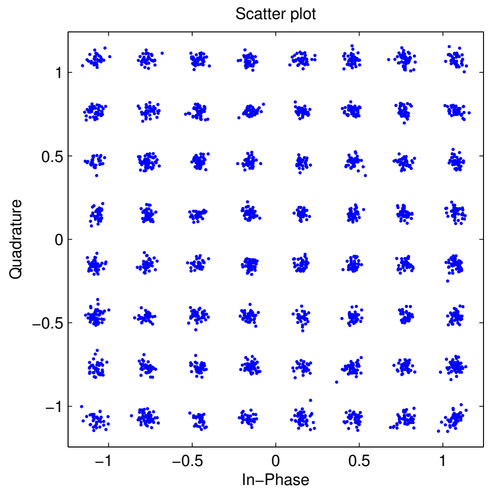


Figure 10: The constellation symbols can be clearly seen by our equalization method.

ure 5), the Low Pass Interpolation method uses the pilot subcarriers to reconstruct the data subcarriers by applying an ideal low pass filter on the pilot subcarriers. Specifically, let $X_{p_1}, \ldots, X_{p_4}$ be the known symbols on pilot subcarriers in the current OFDM symbol. The channel state for those pilot subcarriers are derived by:

$$H_{p_k} = \frac{Y_{p_k}}{X_{p_k}}, k = 1, \ldots, 4 \tag{3}$$

Let $\mathbf{H}^{(P)} = \{H_k^{(P)}\}$, where $H_k^{(P)} = H_k$ if $k \in \{p_1, \ldots, p_4\}$, and $H_k^{(P)} = 0$ otherwise. Now using a low pass filter $\mathbf{F}$ with predefined coefficients $F_1, \ldots, F_N$, the channel states for data subcarriers are estimated by:

$$H = H^{(P)} * \mathbf{F} \tag{4}$$

where $*$ denotes the convolution operation. The effect of LPI equalization is shown in Figure 8, where subcarrier symbols are reconstructed to their original phase and amplitude. However, there are a considerable amount of symbols not recovered correctly, which can be seen as noise in between constellation points in Figure 8. By experimentation, we conclude that this fraction of noise is, unfortunately, higher than the error correction capability of the Wi-Fi convolutional code, resulting in incorrect packet reception. We find that the LPI equalization method fails to reconstruct the signal because the sinc shape assumption does not always hold. Figure 9 shows an example of an abnormal shape of an OFDM symbol, where subcarriers are distorted in a unpredictable manner.

Decision Directed (DD): We have seen that the above discussed equalization methods only rely on the pilot symbols or preamble symbols to estimate the channel for data subcarriers in subsequent OFDM symbols. This limits the estimation accuracy because a channel distortion on training

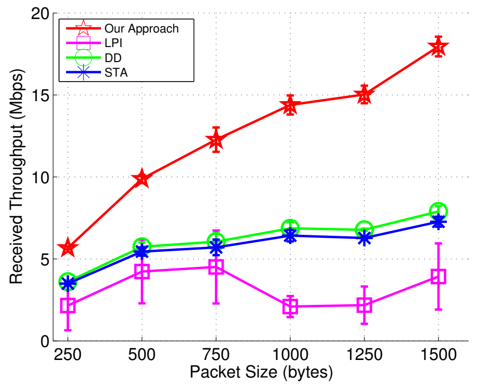


Figure 11: Performance comparison of equalization methods.

and pilot sequences results in a critical impact on the data subcarrier channel recoverability.

The principle of Decision Directed equalization [33],[7] is to use the statistical characteristics of data subcarriers to estimate the channel. In particular, each subcarrier is decoded according to the known modulation of the current OFDM symbol, and the decoded symbol is used as a training symbol to estimate the channel for the next OFDM symbol. It is illustrated by the following steps. First, the received symbol $Y_k^{(n)}$ of the k-th subcarrier at time n is equalized with the previously estimated channel state $H_k^{(n-1)}$ to yield the equalized symbol $\hat{X}_k^{(n)} = \frac{Y_k^{(n)}}{H_k^{(n-1)}}$. Using $\hat{X}_k^{(n)}$, the demodulator finds the closest constellation point $X_k^{(n)}$ and decides it as the decoded symbol. Now with the belief of $X_k^{(n)}$ as the original data symbol, the channel state of the k-th subcarrier for the next OFDM symbol is estimated by

$$H_k^{(n)} = \frac{Y_k^{(n)}}{X_k^{(n)}}. \tag{5}$$

The drawback of DD method is that the error occurring at the current OFDM symbol can propagate to subsequent symbols and destroy the whole packet.

Spectral Temporal Averaging (STA): The Spectral Temporal Averaging method extends the Decision Directed equalization by performing averaging of the channel estimates in both frequency and time domain [11]. Namely, after channel estimates $H_k^{(n)}$ are obtained in Equation (5), the channel states corresponding to adjacent subcarriers are spectral averaged, i.e., $\hat{H}_k^{(n)} = \sum_{m=-\beta}^{\beta} \gamma_m H_{k+m}$. Finally, a temporal averaging is performed to obtain the channel states for the next symbol: $H_k^{(n)} = \alpha \hat{H}_k^{(n)} + (1-\alpha) H_k^{(n-1)}$. The STA method relies on the assumption of likelihood of adjacent subcarriers and channel states changing slowly over time. However, as we show later, this approach does not improve the equalization quality, and in most of the cases, it performs worse than the DD method.

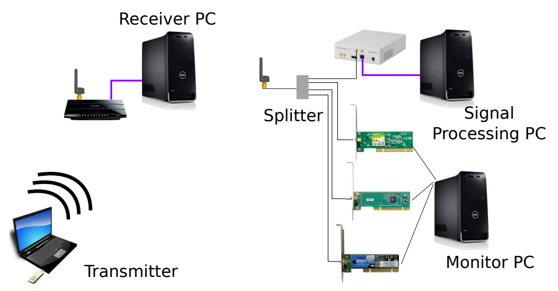


Figure 12: Over the air testbed setup.

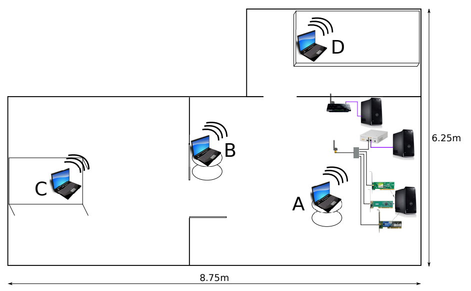


Figure 13: Locations of transmitters/receivers for over the air experiment.

Our approach: During our experimentation, we observe that while the amplitudes of pilot subcarriers can fluctuate significantly due to the channel variations, the pilots' phase is more stable. This motivates us to use pilot subcarriers only for phase tracking. Our idea is that we first derive the phase offset of the current OFDM symbol based on the pilot information. After compensating for the phase offset, the amplitudes of data subcarriers are recovered by applying the principle of Decision Directed method. However, different from the basic Decision Directed equalization, we only update the channel states if the mean squared error of the decoded symbols does not exceed a threshold β. In addition, to avoid the wrong channel estimates of the current OFDM symbol leading to error propagation in subsequent symbols, we update the channel states by a moving averaging over the previous states.

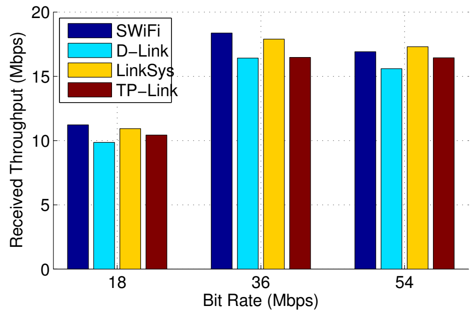
Our detailed solutions contains of the following steps:

- First, based on the long preamble symbols, we compute the initial channel states:

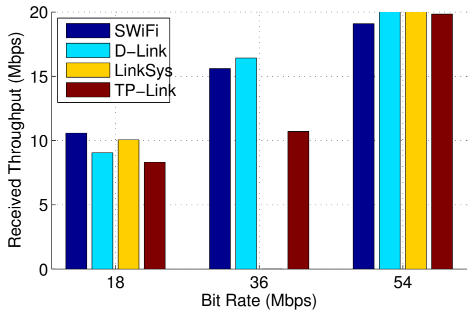
$$\mathbf{H}^{(-1)} = \frac{1}{2}\left(\frac{\mathbf{Y}_{-2}}{\mathbf{X}_{-2}} + \frac{\mathbf{Y}_{-1}}{\mathbf{X}_{-1}}\right).$$

  This step is handled by the OFDM Channel Estimator in the receive chain.

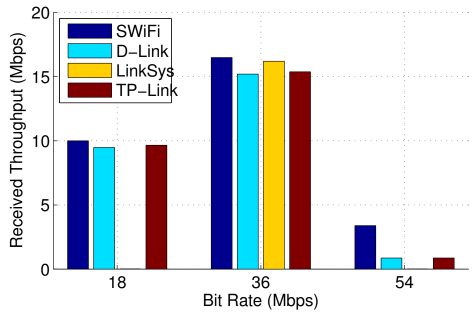
- Let ϕ be the phase offset experienced in the current OFDM symbol, and $X_{p_k}^{(n)}$ denote the training symbol in the pilot subcarrier p_k. We have $Y_{p_k}^{(n)} = H_{p_k}^{(n)} X_{p_k}^{(n)} = |H_{p_k}^{(n)}| e^{j\phi} X_{p_k}^{(n)}$. To estimate the phase offset ϕ, we first



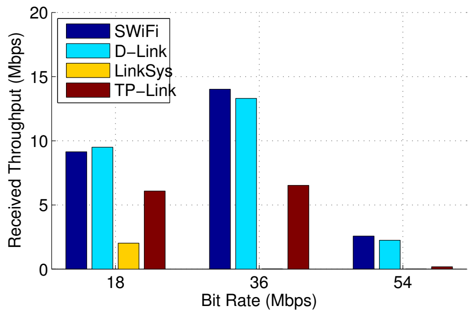
(a) Location A



(b) Location B



(c) Location C



(d) Location D

Figure 14: Throughput comparison between SWiFi and commercial Wi-Fi cards in wireless setup with 1500-byte packet transmission.

compute

$$A = \sum_{k=1}^{4} \frac{Y_{p_k}^{(n)}}{X_{p_k}^{(n)}} \left(H_{p_k}^{(-1)}\right)^* = e^{j\phi} \sum_{k=1}^{4} |H_{p_k}^{(n)}|$$

and derive the estimate $\phi = \arg(A)$.

- Now using ϕ, we compensate the phase offset for all data subcarriers k to obtain $\hat{Y}_k = Y_k^{(n)} e^{-j\phi}$. Based on the constellation, we find the closest symbol $\hat{X}_k$ to $\hat{Y}_k$ and evaluate the mean square error (MSE) $\epsilon = \frac{1}{N}\sum_{k=1}^{N} |\hat{Y}_k - \hat{X}_k|^2$. We update the channel states for the next symbol only if the computed MSE does not exceed a threshold β.

- The update of channel states is performed by averaging out:

$$H_k^{(n)} = \alpha \frac{\hat{Y}_k}{\hat{X}_k} + (1-\alpha) H_k^{(n-1)}.$$

Our experimental evaluation shows that the performance of our technique in comparison with LS, LR, LPI, DD, STA, we report an improvement of almost 100% in comparison with the second best technique (DD) for any of the considered rates and packet sizes (Figure 11). We notice an increasing improvement of performance as a function of packet size. We also note that the performance of LS and LR is very low, and therefore is not reported here. From the results, we can observe an increasing improvement of performance as a function of packet size for all equalization methods except for the LPI method. With the LPI equalization, the performance is quite dependent on the dynamic environment, because the channels are estimated and interpolated solely based on the pilot subcarriers. The dynamic environment causes the OFDM symbols belonging to the same packet to experience completely different channel attenuations and phase rotations, resulting in a drop in performance when the packet is longer than an environment-dependent threshold; for instance, increasing packet size from 750 bytes to 1000 bytes reduces the overall received throughput. We note that when the packet size is increased beyond the threshold, the LPI performance is increased slightly due to the larger packet size.

5. EVALUATION

In order to accurately characterize SWiFi, we methodically compared its performance to several commercial IEEE 802.11abg cards. We focussed on the most challenging aspects namely the performance of the OFDM receiver. We carried out measurements for both typical over the air communications and wired communication with controlled attenuators. The experiments considered the impact of packet size, rate, attenuation, and location.

5.1 Components and common methodology

We first summarize the various components used in our measurement experiments. For comparing with commercial Wi-Fi devices, we selected cards from well known and popular manufacturers (D-Link, Linksys, TP-Link) based on the widely used Atheros and Ralink chipsets. We selected cards for which it is possible to connect a single external antenna and which have Linux drivers available. The three selected cards have a PCI interface. Table 1 lists the various components, manufacturers, models, and key characteristics.

Table 1: Components used for over the air and wired testbeds.

Component	Brand/Model	Misc.
Hosts	Dell XPS 8500	i7 quad-core 3.4GHz, 16GB RAM
Traffic generator	Iperf	Ubuntu 14.04
SDR 1	Ettus USRP N210	SBX 0.4-4.4GHz
SDR 2	HackRF One	10MHz-6GHz
Access Point	TP-Link TL-WDR4300	Atheros AR9344
Wi-Fi USB	TP-Link TL-WN722N	Atheros AR9271 (ath9k_htc)
Wi-Fi NIC 1	TP-Link TL-WN751ND	Atheros AR9227 (ath9k)
Wi-Fi NIC 2	D-Link DWA-525	Ralink RT5360 (rx2x00pci)
Wi-Fi NIC 3	Linksys WMP54G	Ralink RT2500 (rt2x00pci)
Antenna	Antenova Titanis Swivel	2.2dBi (peak)
Splitter	L-com Hypergain	
Attenuators	Mini-Circuits 1-30dB	
RF Cables	L-com SMA	

In both the over the air and wired experiments, the devices under comparison operate as sniffers of the traffic between a Laptop transmitter and the Access Point (See Figures 12 and 16). The traffic is generated using the Iperf throughput measurement tool from the laptop to a PC connected to the Access Point. To obtain meaningful and fair results, all the devices under comparison are connected to a splitter to obtain a copy of the same RF signal.

Each experiment is run for 10 seconds and repeated 5 times. For each experiment we filter all the packets that are received without errors (correct CRC) and derive the throughput as seen by each device. We run experiments for each of the following IEEE802.11ag rates 18 Mbps (QPSK), 36 Mbps (16-QAM), and 54 Mbps (64-QAM), all with 3/4 convolutional code. For each experiment, we fixed the rate of the transmitter (driver) and verified that all received and counted packets are at the specified rate. We focused on rates with high order modulations because they are the most challenging. We considered both packets of size 1000 Bytes and 1500 Bytes. Based on the number of correctly received packets we computed the net throughput (IP and above) as seen by each Wi-Fi card/SDR. Note that this throughput is typically smaller than the raw physical layer rate since it does not include the IEEE802.11 overhead (preamble, header, ACK, SIFS, DIFS, Backoff) and concurrent traffic (for over the air experiments). All experiments are over the 2.4GHz band. We plan to comprehensively evaluate SWiFi over the 5GHz band once we setup a testbed with a USRP CBX daughterboard that can reach 6GHz, splitters (for 5GHz), and 802.11a transmitters with external antennas.

Our comprehensive evaluation focused on the USRP N210. Our early preliminary evaluation of the HackRF indicates that it achieves a slighly lower performance than the USRP N210 despite a significantly lower quality 8 bits ADC instead of a 14 bits for the N210. Note that 8 bit ADCs are typical for commercial Wi-Fi cards. The main obstacle to carry a comprehensive evaluation of the HackRF is that it intermittently overflows the USB link when using a sampling rate of 20 Msps. This is because the HackRF transfers 8 byte per sample (a 4 bytes float for I and respectively 4 bytes for Q) resulting in 1.28Gbps bandwidth requirements which far exceeds the 480 Mbps theoretical limit of USB. However, this is not a fundamental problem as a reprogramming of the HackRF CPLD/microcontroller (ARM LPC4330 Cortex M4/M0) can reduce the bandwidth requirements by a factor of 4, sending the ADC values as 2 bytes instead of 8. The preliminary values we obtained for windows of samples that did not experience overflows indicates a slight but not substantial performance degradation in comparison with the USRP N210.

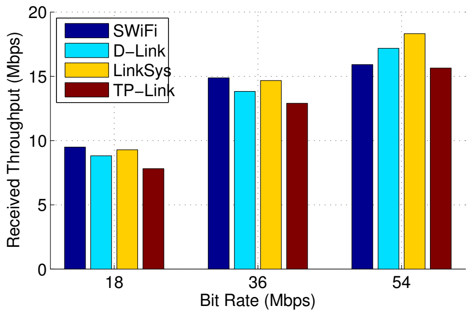
5.2 Performance over the air

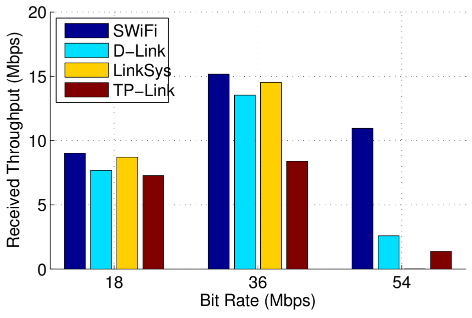
We started with a set of measurements over the air. Our goal was to confirm that, in a typical environment, SWiFi has a similar performance as commercial cards. Given that the RF environment is highly sensitive to location and time, the RF signal is sniffed by a single antenna and connected to a splitter that feeds the USRP and other cards under comparison.

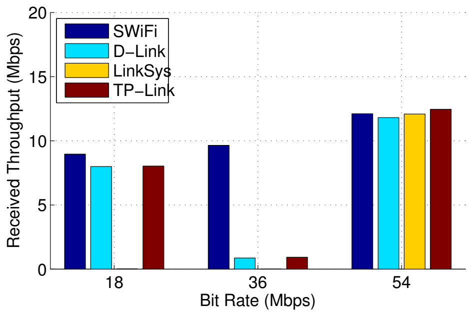
We considered four locations for the transmitter, while keeping the receivers fixed (Figure 12). The transmiter locations were selected to create different types of link, from short distance un-obstructed to the most challenging ones with no line-of-sight. We evaluated the performance with packets of 1000 bytes and 1500 bytes. We note that most of the time SWiFi slightly outperforms the commercial cards. One interesting observation is that at low SNR, the performance of commercial cards is very unstable in comparison with SWiFi. In particular the Linksys WMP54G card was the most unstable which might be due to its relatively older chipset Ralink RT2500. It has the best performance of the three cards in good channel conditions, and the worst in harsh channel conditions. The D-Link DWA-525 with Ralink RT5360 chipset performed the closest to SWiFi except in one configuration where SWiFi significantly outperformed all the commercial cards. As is well knows over the air performance evaluation is very sensitive to time, location, and instantaneous interference. It is therfore possible to compare the receivers to each other, as they experience the same environment, but hard to predict performance of individual cards. An illustration of this is that in our experiments 1500 bytes packets seem to result in better throughput. This can only be partially explained by the lower overhead (i.e., headers, DIFS, backoff, ACK), an other explanation is that the environment was different. This also motivate our evaluation and comparison in the controlled attenuation/propagation setup.

5.3 Controlled attenuation evaluation

In order to obtain a clearer view on how the different cards fair against SWiFi as the SNR decreases, we carried a set of controlled experiments where the transmitter's signal is first attenuated by 30 dB, then connected to a first splitter. One output of this splitter is further attenuated and connected to


(a) Location A


(b) Location B


(c) Location C

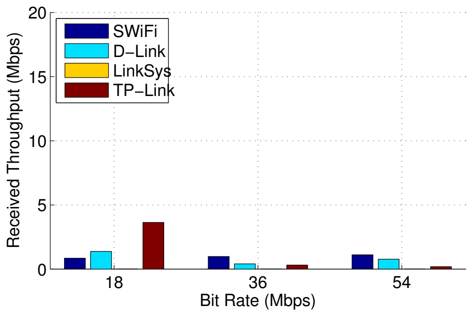

(d) Location D

Figure 15: Throughput comparison between SWiFi and commercial Wi-Fi cards in wireless setup with 1000-byte packet transmission.

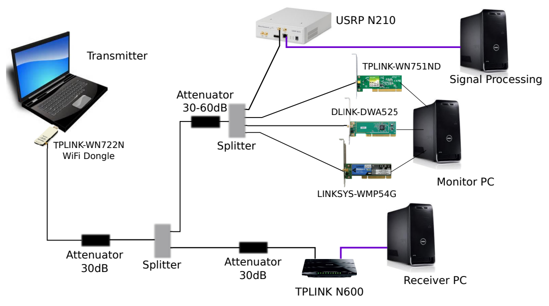


Figure 16: Controlled attenuation testbed setup.

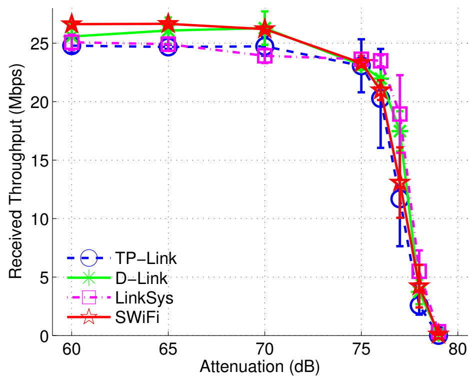


Figure 17: Throughput comparison between SWiFi and commodity WiFi cards in wired setup with 1500-byte packet transmission.

the access point, while the other output is attenuated within a range of 30-60dB before being split and connected to the sniffing devices under comparison. This allowed us to understand how the performance of each of the cards degrades as a function of signal attenuation, packet size, and rate.

We observe that at high SNR all cards perform well. SWiFi achieves slightly better performance than the commercial cards. When the attenuation is increased beyond 75 dB the performance of all the cards drops to 0 at 79-80dB attenuation. The similarity in performance of all the cards is remarkable for 1500 bytes packets (See Figure 17). It is also interesting to note that for packets of length 1000 bytes, the Linksys card achieves a slightly better performance than the other cards (1-2dB See Figure 18). Combined with the results for over the air evaluation, it seems that the Linksys is sensitive to interference and propagation effects more than to low values of SNR. This is consistent with the high volatility in the over the air results of Linksys as such environments are highly dynamic in the crowded 2.4GHz band. A second observation is that 1000 bytes packets achieve lower throughput consistently at high SNR due to the unnecessary overhead of PHY/MAC headers, SIFS/ACK/DIFS and backoffs.

6. ENABLING WI-FI ANALYSIS

We believe that SWiFi will enable more sophisticated analysis of Wi-Fi networks in particular for high order modulation rates, in addition to enabling the implementation of novel techniques in a Wi-Fi compatible physical layer. To support our claim, we started the development of several

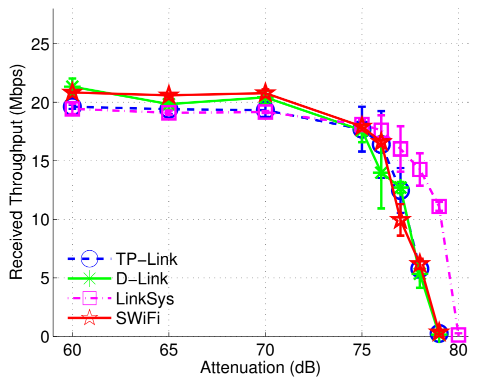


Figure 18: Throughput comparison between SWiFi and commodity WiFi cards in wired setup with 1000-byte packet transmission.

companion tools that we plan to make available to the research community as open source.

6.1 Timing Analysis

Our first tool can analyze the SWiFi trace and output timing measurements on a per card basis. For example, Table 2 shows the SIFS between data and ACK frames collected by our SWiFi receiver when monitoring the traffic between the commercial cards. Note that while a tool such as wireshark reports a timestamp as part of the radiotap header, that is supposed to correspond to the time when the first bit of a packet is received, our analysis indicates that it is highly inaccurate leading to estimated values for SIFS sometimes corresponding to several hundred microseconds.

Table 2: Average SIFS values computed based on monitoring IEEE 802.11gn transmissions between commercial cards on the 2.4GHz.

Card	**SIFS**
TP-Link Atheros AR9485	17 us
Intel Centrino Wireless N-1000	18 us
TP-Link (air capture)	16.8 us

This tool also allow to visualize and navigate a SWiFi trace providing information about the src, destination, rate, and most importantly an exact timing information (see Figure 19). We plan to extend it to also visualize collisions and overlapping packets (based on some of the preliminary results we have for extracting the header of overlapping packets using successive interference techniques).

6.2 Open Source Release to the Community

Once the restrictions on the anonymous submission are lifted, we will make the SWiFi platform available to the research community. Beyond making the source code available, we plan to create an ORBIT image with the source code of SWiFi, companion tools, and scripts running experiments to validate and confirm our claims. Examples of such experiments would consist of an ORBIT configuration with Wi-Fi traffic between two nodes, and sniffed by the USRP N210 and by the Wi-Fi interface of some of the ORBIT

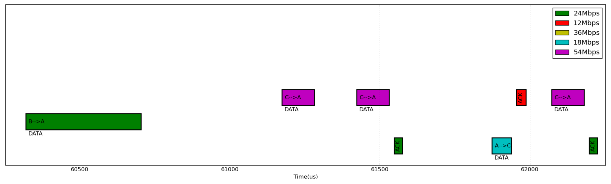


Figure 19: Packet flow visualization with rate and precise timing information.

nodes. The output of SWiFi can be compared to the output of the Wi-Fi sniffers. Such results can easily be reproduced. Unfortunately, ORBIT does not allow hardware reconfiguration (e.g., installation of splitters) to enable testbeds such as the one we used for comparing commercial cards to SWiFi. However, we hope that the detailed description of our testbed will enable others to reproduce and validate our results.

7. RELATED WORK

The success of the IEEE802.11 standard and the ubiquity of Wi-Fi networks, attracted a significant amount of research, devoted to their modeling, analysis and evaluation. Early work modeled 802.11 MAC layer using Markov chains and developed an analytical formulation for saturation throughput [4]. Despite the elegant formulation and analytical power, these results were limited to a simple interference model that assumes that all nodes are within range of each other, are IEEE802.11 compliant, and that collisions are binary. More sophisticated models extended the Markov chain framework by considering the hidden terminal scenarios [22], multi-rate physical layer [10], and worst-case interference (jamming) [2]. However, these extensions still retained many of the limitations of the original model, namely a simplistic physical layer and a prediction power that focuses on the steady state behavior and saturation throughput. A large body of work focused on analyzing IEEE802.11 networks using discrete event network simulators such as ns-2 [26], ns-3 [27], OPNET [36], GloMoSim [43], and QualNet [38]. While such environments simplified the analysis of more complex network topologies and accounted for some of the RF signal propagation effects, they were still limited by models that did not correspond to (1) realistic propagation environments, (2) realistic models of IEEE802.11 devices, and (3) realistic models of Bit Error Rate and Frame Error Rate as a function of the considered modulation, coding, noise, interference, and propagation. This can be illustrated by the early models of ns-2 that assumed that all packets with Signal to Noise Ratio exceeding a given ratio will be correctly received, and even ns-3 still retains several flaws in computing the frame error rate of IEEE802.11. Note that discrete event simulators typically focus on the MAC and higher layers behavior, while simulators such as Mathworks Matlab/Simulink are better at simulating the physical and link layers but are not able to scale to networks for nodes with realistic propagation environments.

The limitations of analytical frameworks and simulators motivated wireless systems researchers to develop experimental methods, platforms, and testbed to characterize and develop better models for Wi-Fi networks, but also to evaluate the performance of new algorithms while accounting for realistic propagation, interference, and RF front ends limitations. The ORBIT project provides several flexible wireless networks [28], [18]. Its largest testbed consists of 400 Wi-Fi nodes arranged in a 20 by 20 grid and is supplemented by a limited number of Ettus USRP SDR peripherals. Orbit was successful in enabling large scale, fairly reproducible Wi-Fi experiments, in a controlled environment. It is however not designed for analyzing Wi-Fi networks in the wild. In parallel, several researchers developed measurement-driven approaches to analyze and experiment with dense Wi-Fi networks in real world setting such as [8]. Measurement-driven approaches to Wi-Fi led to better protocols for frequency/channel auto-configuration, load-balancing, power-control, and rate adaptation [42], [40], [25]. However, most of the early work was limited by the limited view of the Wi-Fi channel and link provided by the Hardware Abstraction Layer and the driver API (e.g., RSSI, number of retransmissions). More recently, the availability of a relatively richer set of information about the Wi-Fi channel (RSSI per OFDM sub-carrier at KHz rate) combined with clever algorithms enabled a finer grain characterization of the channel. In particular, it became possible to detect and characterize a wide variety of non-Wi-Fi devices sharing the ISM band [34], [35], [30], [29].

Several research groups considered an alternative approach, to analyzing Wi-Fi networks and their co-existence with other wireless devices over ISM bands. They relied on flexible software defined radios to obtain a much finer grain characterization of the RF spectrum. The introduction, by BBN, of a first SDR implementation of IEEE802.11 (1 & 2Mbps), that runs on the popular Ettus USRP, was a first step to enabling a wide variety of projects from measurement, analysis and optimization of Wi-Fi protocols, to localization, wireless security, and cognitive radios (See [14] for a list of projects). For example RFDump used this implementation to develop a real time wireless multi-protocol analysis tool [23]. Others demonstrated the feasibility of stealthy man-in-the-middle attacks against previously believed to be secure WPA-Enterprise networks [9]. Key to this sophisticated attack is the capability to detect & jam Wi-Fi probes, sent by targeted devices, before the transmission of the CRC, making them invisible to neighboring Access Points. The main limitation of the BBN implementation (besides only supporting rates 1 & 2 Mbps) was the constrained bandwidth of the USRP USB link used to transfer baseband samples to the host PC. This forced the developers to down-sample the baseband signal from 11MHz to 4MHz, therefore significantly degrading the quality of the signal. An alternative implementation by the University of Utah utilized the FPGA capability to detect the frame preamble and despread the baseband signal before transfer to the host computer over USB [12]. This significantly reduced the bandwidth requirement on the USB link, however it still focused on IEEE802.11b at rates 1 & 2 Mbps. In February 2015, National Instruments announced a commercial OFDM implementation based on the IEEE802.11 standard [24]. Despite its price of $5K, this implementation is not fully compatible with the IEEE802.11 as it implements a simplified PHY frame. The only programmable platform supporting IEEE802.11abg is Rice University's WARP that is commercialized by Mango Communications [41]. Besides its high cost, WARP requires an FPGA implementation of IEEE802.11 which limits the flexibility, ease of programmability and leveraging of the computation capability of the host computer.

The most related to our work is the work by Bloessl et al. [5], [6]. In the first version [5], this open source GNU Radio based implementation is limited to QPSK and not able to correctly receive neither 16-QAM nor 64-QAM modulated signals. Their extended work [6] has improved the equalization method by an estimation based on long preamble symbols. With their improved version, 54 Mbps packet decoding has been fully supported.

8. CONCLUSION

We introduced SWiFi, an Open Source Wi-Fi SDR stack capable of successfully decoding Wi-Fi packets up to 54 Mbps. SWiFi relies on a combination of algorithms for frequency offset correction, and frequency domain equalization utilizing pilot-based phase tracking, and decision directed method for amplitudes equalization. We evaluate the performance of SWiFi on the Ettus USRP N210 over the 2.4 GHz band and compare its performance to three commercial Wi-Fi cards from well known manufacturers and based on popular chipsets. We demonstrate that SWiFi performs at least as well as the commercial cards and exhibits a higher stability in harsh environments. We also demonstrate the potential of this platform for enabling cross-layer research such as timing analysis. We believe that SWiFi can be extended in many ways for Wi-Fi networks analysis. For example, the incorporation of soft-decoding decoding and successive interference cancellation can enable a much better view of capture and hidden terminal effects in dense networks.

9. REFERENCES

[1] 3GPP TS 24.312. Access Network Discovery and Selection Function (ANDSF) Management Object (MO). `http://www.3gpp.org/DynaReport/24312.htm`, 2014.

[2] E. Bayraktaroglu, C. King, X. Liu, G. Noubir, R. Rajaraman, and B. Thapa. Performance of IEEE 802.11 under jamming. *Mobile Networks and Applications*, pages 1–19, 2011.

[3] BBN Technologies. ADROIT GNU Radio development. `https://moo.cmcl.cs.cmu.edu/trac/cgran/wiki/BBN80211`.

[4] G. Bianchi. Performance analysis of the ieee 802.11 distributed coordination function. *IEEE Journal on Selected Areas in Communications*, 18(3):535–547, March 2000.

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

[6] B. Bloessl, M. Segata, C. Sommer, and F. Dressler. Towards an Open Source IEEE 802.11p Stack: A Full SDR-based Transceiver in

GNURadio. In *5th IEEE Vehicular Networking Conference (VNC 2013)*, pages 143–149, Boston, MA, December 2013. IEEE.

[7] A. Bourdoux, H. Cappelle, and A. Dejonghe. Channel tracking for fast time-varying channels in ieee802.11p systems. In *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*, pages 1–6, Dec 2011.

[8] I. Broustis, K. Papagiannaki, S. V. Krishnamurthy, M. Faloutsos, and V. P. Mhatre. Measurement-driven guidelines for 802.11 WLAN design. *IEEE/ACM Transactions on Networking*, 18(3):722–735, June 2010.

[9] A. Cassola, W. Robertson, E. Kirda, and G. Noubir. A practical, targeted, and stealthy attack against WPA-Enterprise authentication. In *Proceedings of the 20th Annual Network & Distributed System Security Symposium, NDSS*, 2013.

[10] J. Choi, K. Park, and C.-K. Kim. Cross-layer analysis of rate adaptation, dcf and tcp in multi-rate wlans. In *IEEE INFOCOM*, pages 1055–1063, May 2007.

[11] J. Fernandez, K. Borries, L. Cheng, B. Kumar, D. Stancil, and F. Bai. Performance of the 802.11p physical layer in vehicle-to-vehicle environments. *Vehicular Technology, IEEE Transactions on*, 61(1):3–14, Jan 2012.

[12] H. Firooz, N. Patwari, J. Zhang, and S. K. Kasera. Implementation of full-bandwidth 802.11b receiver. `http://span.ece.utah.edu/pmwiki/pmwiki.php?n=Main.80211bReceiver`, 2008.

[13] Free Forfait Mobile. FreeWiFi secure EAP-SIM. `http://mobile.free.fr/assistance/261.html`, August 2010.

[14] GNU Radio. The comprehensive GNU Radio archive network. `https://moo.cmcl.cs.cmu.edu/trac/cgran/wiki/Projects`.

[15] S. Gollakota, F. Adib, D. Katabi, and S. Seshan. Clearing the RF smog: making 802.11n robust to cross-technology interference. In *Proceedings of the ACM SIGCOMM 2011 conference*, SIGCOMM'11, 2011.

[16] S. Gollakota and D. Katabi. Zigzag decoding: Combating hidden terminals in wireless networks. In *Proceedings of the ACM SIGCOMM 2008 Conference on Data Communication*, SIGCOMM '08, pages 159–170, 2008.

[17] Great Scott Gadgets. Hackrf one. `https://greatscottgadgets.com/hackrf/`.

[18] G. C. Hadjichristofi, A. Brender, M. Gruteser, R. Mahindra, and I. Seskar. A wired-wireless testbed architecture for network layer experimentation based on ORBIT and VINI. In *Proceedings of the ACM International Workshop on Wireless Network Testbeds, Experimental Evaluation and Characterization*, WinTECH '07, pages 83–90, 2007.

[19] H. Haverinen and J. Salowey. Extensible Authentication Protocol Method for Global System for Mobile Communications (GSM) Subscriber Identity Modules (EAP-SIM). RFC 4186 (Informational), January 2006.

[20] IEEE. IEEE 802.11 Interworking with External Networks. `http://standards.ieee.org/findstds/standard/802.11u-2011.html`.

[21] IEEE. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. February 2012.

[22] B. Jang and M. Sichitiu. IEEE 802.11 saturation throughput analysis in the presence of hidden terminals. *IEEE/ACM Transactions on Networking*, 20(2):557–570, April 2012.

[23] K. Lakshminarayanan, S. Sapra, S. Seshan, and P. Steenkiste. Rfdump: An architecture for monitoring the wireless ether. In *Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies*, ACM CoNEXT'09, 2009.

[24] National Instruments. LabVIEW communications 802.11 application framework. `http://www.ni.com/product-documentation/52533/en/`, February 2015.

[25] M. Neufeld, J. Fifield, C. Doerr, A. Sheth, and D. Grunwald. SoftMAC - Flexible Wireless Research Platform. In *Fourth Workshop on Hot Topics in Networks (HotNets-IV)*, 2005.

[26] ns 2. The network simulator - ns-2. `http://www.isi.edu/nsnam/ns/`.

[27] ns 3. Nsnam - ns-3. `https://www.nsnam.org`.

[28] M. Ott, I. Seskar, R. Siraccusa, and M. Singh. ORBIT testbed software architecture: supporting experiments as a service. In *Testbeds and Research Infrastructures for the Development of Networks and Communities, 2005. Tridentcom 2005. First International Conference on*, pages 136–145, Feb 2005.

[29] A. Patro, S. Govindan, and S. Banerjee. Observing home wireless experience through WiFi APs. In *Proceedings of the 19th Annual International Conference on Mobile Computing and Networking*, MobiCom'13, pages 339–350, 2013.

[30] A. Patro, S. Rayanchu, and S. Banerjee. Mobicom 2011 poster: AirTrack: Locating non-WiFi interferers using commodity WiFi hardware. *SIGMOBILE Mob. Comput. Commun. Rev.*, 15(4):52–54, Mar. 2012.

[31] Qualcomm. 3G LTE Wifi offload framework: Connectivity Engine (CnE) solution, July 2013. `http://www.qualcomm.com/media/documents/3g-lte-wifi-offload-framework`.

[32] M. Ramsay. Wi-Fi offload rising amid soaring data traffic. `http://www.wirelessweek.com/News/2012/07/technology-WiFi-Offload-Rising-Amid-Soaring-Data-Traffic/`, July 2012.

[33] J. Ran, R. Grunheid, H. Rohling, E. Bolinth, and R. Kern. Decision-directed channel estimation method for ofdm systems with high velocities. In *Vehicular Technology Conference, 2003. VTC 2003-Spring. The 57th IEEE Semiannual*, volume 4, pages 2358–2361 vol.4, April 2003.

[34] S. Rayanchu, A. Patro, and S. Banerjee. Airshark: Detecting non-WiFi rf devices using commodity WiFi hardware. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, IMC'11, pages 137–154, 2011.

[35] S. Rayanchu, A. Patro, and S. Banerjee. Catching whales and minnows using WiFiNet: Deconstructing non-WiFi interference using WiFi hardware. In *Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation*, NSDI'12, 2012.

[36] Riverbed. OPNET modeler. `http:www.riverbed.com/products/performance-management-control/`.

[37] B. Rooney. Data-hungry 4G users gorge on Wi-Fi, report finds. `http://blogs.wsj.com/tech-europe/2013/09/19/data-hungry-4g-users-gorge-on-wi-fi-report-finds/`, September 2013.

[38] Scalable Network Technologies. QualNet communications simulation platform. `http://scalable-networks.com/content/qualnet`.

[39] T. Schmidl and D. Cox. Robust frequency and timing synchronization for OFDM. *Communications, IEEE Transactions on*, 45(12):1613–1621, Dec 1997.

[40] A. Sharma, M. Tiwari, and H. Zheng. Madmac: Building a reconfigurable radio testbed using commodity 802.11 hardware. In *IEEE SECON-WSDR*, 2006.

[41] WARP. Wireless open-access research platform. `http://warp.rice.edu/`.

[42] H. Wu, X. Wang, Y. Liu, Q. Zhang, and Z.-L. Zhang. Softmac: layer 2.5 mac for voip support in multi-hop wireless networks. In *IEEE SECON*, pages 441–451, 2005.

[43] X. Zeng, R. Bagrodia, and M. Gerla. GloMoSim: A library for parallel simulation of large-scale wireless networks. In *Proceedings of the Twelfth Workshop on Parallel and Distributed Simulation*, PADS '98, pages 154–161, 1998.