Multi-User Gesture Recognition Using WiFi

Raghav H. Venkatnarayan
North Carolina State University
Raleigh, North Carolina
rhampap@ncsu.edu

Griffin Page
North Carolina State University
Raleigh, North Carolina
gcpage@ncsu.edu

Muhammad Shahzad
North Carolina State University
Raleigh, North Carolina
mshahza@ncsu.edu

ABSTRACT

WiFi based gesture recognition has received significant attention over the past few years. However, the key limitation of prior WiFi based gesture recognition systems is that they cannot recognize the gestures of multiple users performing them simultaneously. In this paper, we address this limitation and propose WiMU, a WiFi based Multi-User gesture recognition system. The key idea behind WiMU is that when it detects that some users have performed some gestures simultaneously, it first automatically determines the number of simultaneously performed gestures (N_a) and then, using the training samples collected from a single user, generates virtual samples for various plausible combinations of N_a gestures. The key property of these virtual samples is that the virtual samples for any given combination of gestures are identical to the real samples that would result from real users performing that combination of gestures. WiMU compares the detected sample against these virtual samples and recognizes the simultaneously performed gestures. We implemented and extensively evaluated WiMU using commodity WiFi devices. Our results show that WiMU recognizes 2, 3, 4, 5, and 6 simultaneously performed gestures with accuracies of 95.0, 94.6, 93.6, 92.6, and 90.9%, respectively.

CCS CONCEPTS

• **Human-centered computing → Gestural input**;

KEYWORDS

Gesture recognition, WiFi, Multi-user

ACM Reference Format:
Raghav H. Venkatnarayan, Griffin Page, and Muhammad Shahzad. 2018. Multi-User Gesture Recognition Using WiFi. In *MobiSys '18: The 16th Annual International Conference on Mobile Systems, Applications, and Services, June 10–15, 2018, Munich, Germany*. ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3210240.3210335

1 INTRODUCTION

Motivation: Human gesture and activity recognition is the core technology that enables a countless number of applications in diverse areas, such as health care monitoring [11, 23], sleep monitoring [24], fitness tracking [12, 19], and elderly care [22, 35]. Recently, a new class of human gesture recognition systems has spawned that leverages WiFi signals to recognize human gestures [6, 13, 14, 17, 20, 25, 27, 28, 30, 34]. The fundamental principle that enables human gesture recognition using WiFi signals is that the wireless channel metrics, such as channel state information (CSI) and received signal strength (RSS), change when a user moves in a wireless environment. The patterns of change in these wireless channel metrics are unique across different gestures. WiFi based gesture recognition systems first learn these patterns of change using machine learning techniques for each predefined gesture and then recognize them as the user performs them at runtime.

The key limitation of prior WiFi based gesture recognition systems is that they can recognize a user's gestures only if a single user moves in the environment. If multiple users move simultaneously, prior WiFi based systems cannot recognize their gestures anymore. The only exception is WiSee [25] that made an effort towards recognizing gestures in the presence of *interfering* users. However, when considered from the perspective of multi-user gesture recognition, WiSee has two limitations (as also mentioned by the authors in [25]). First, it does not recognize gestures of all users when multiple users perform them simultaneously; it only recognizes gestures of one user and considers the remaining as *interfering* users. Second, in the presence of interfering users, it can recognize only two gestures, push and pull, and the accuracy drops to 60% when the number of interfering users reaches four. It is worth noting, however, that the primary objective of WiSee was not to develop a multi-user compatible gesture recognition system, rather it was to demonstrate the feasibility of using WiFi signals for gesture recognition. Consequently, WiSee's treatment of handling multiple users was only preliminary. To conclude, the inability of existing WiFi based gesture recognition systems to recognize simultaneously performed gestures significantly restricts their practical usability. Therefore, to bring WiFi based gesture recognition a step closer to real world deployment, it is imperative to address and overcome this key limitation of prior art.

Problem Statement: In this paper, our objective is to design a WiFi based gesture recognition system that can recognize gestures of multiple users when the users perform them simultaneously. For a set of gestures to be considered simultaneously performed, the time when any given gesture in that set was performed should have some overlap with the time of performance of at least one other gesture in that set. The gesture recognition system should further satisfy three requirements: 1) the start and end times of gestures are not required to be synchronized across users; 2) each user can perform any of the predefined gestures; and 3) the system can be implemented on commodity WiFi devices. Note that our objective is only to recognize predefined gestures when the users perform them simultaneously, and not to further determine which user performed which predefined gesture.

Proposed Approach: A possible solution to recognize simultaneously performed gestures of multiple users is to first request

MobiSys '18, June 10–15, 2018, Munich, Germany

ACM ISBN 978-1-4503-5720-3/18/06…$15.00
https://doi.org/10.1145/3210240.3210335

users to provide training samples for all possible combinations of gestures and then use these training samples to learn the patterns of change in wireless channel metrics for each combination. While theoretically plausible, this solution is impractical because the number of training samples to collect would be prohibitively large. For example, to recognize N_p predefined gestures of up to N_a simultaneously performing users, the total number of possible combinations (with repetitions allowed, *i.e.*, more than one user can perform the same gesture) is $\sum_{i=1}^{N_a}(N_p)^i$, which evaluates to 19530 for $N_p = 5$ and $N_a = 6$. With 10 training samples per combination, this solution will require over a whopping 195300 training samples. The number of these combinations actually become infinite when we consider the fact that in practical settings, the start and end times of gestures of users are not synchronized.

Another approach is to first separate the contribution of each user's movements from the net measurements of wireless channel metrics and then evaluate the separated contribution against the training samples of each predefined gesture. The advantage of this approach is that it requires training samples for each predefined gesture from only a single user. Consequently, regardless of the value of N_a, if we collect 10 training samples per gesture, only one user has to provide just $10N_p$ samples. When $N_p = 5$, the number of required training samples is just 50, which is over three orders of magnitude smaller than 195300, and practically easy to provide. Unfortunately, separating the contribution of each user's movements from the net measurements of wireless channel metrics is difficult, and we are unaware of any existing methods that accurately and effectively do this. This is, arguably, the key reason that the problem of WiFi based multi-user gesture recognition has not yet seen a successful solution. Our own exploration of this approach by using independent component analysis [3, 18] and blind signal separation [10, 21]) based methods did not result in acceptable gesture recognition accuracies.

In this paper, we present WiMU, a WiFi based Multi-User gesture recognition system that recognizes the gestures of multiple users when the users perform them simultaneously. The key component of WiMU is our novel method that can generate a *virtual sample* for any desired combination of gestures using a real sample of each gesture in that combination. The key property of our method is that the virtual samples that it generates for any desired combination of gestures are identical to the real samples that would result from real users performing that combination of gestures simultaneously. Consequently, instead of requiring multiple users to provide training samples for all possible combinations of predefined gestures, WiMU requires only a single user to provide training samples for each gesture only individually. It then uses these training samples to generate virtual samples for all those combinations of gestures for which training samples are required, and uses these virtual samples for training. The key idea behind WiMU is that, as soon as it detects that some users have performed gestures simultaneously, it first automatically determines the number of simultaneously performed gestures (represented by N_a, where $N_a \geq 1$) from the detected sample. It also identifies the start and end times of these N_a gestures. Next, it generates virtual samples for various plausible combinations of N_a gestures and compares the detected sample with them to recognize the simultaneously performed gestures.

Note that WiMU does not suffer from the shortcomings of either of the two approaches described earlier: it neither requires users to provide training samples for all possible combinations nor requires to separate the contribution of each user's movements from the net measurements of wireless channel metrics. WiMU further satisfies all three requirements mentioned in the problem statement. WiMU operates on channel state information (CSI), a well known wireless channel metric that has been extensively used in several existing WiFi based gesture and activity recognition systems [6, 17, 32, 34].

Key Contributions: In this paper, we make three key contributions. First, we propose a method to generate virtual samples that enables WiFi based multi-user gesture recognition without requiring users to provide training samples for all possible combinations of gestures and without requiring to separate the contribution of each user's movements from the net measurements of wireless channel metrics. Second, we propose a method to automatically identify the number of gestures as well as the start and end times of different gestures when multiple users perform them simultaneously. Third, we present our implementation and extensive evaluation of WiMU on commodity WiFi devices. Our results show that WiMU recognizes 2, 3, 4, 5, and 6 simultaneously performed gestures with average accuracies of 95.0, 94.6, 93.6, 92.6, and 90.9 percent, respectively.

2 RELATED WORK

Prior WiFi based gesture recognition systems cannot recognize gestures when multiple users perform them simultaneously. We categorize prior work on wireless signal (including WiFi) based human sensing systems into two broad categories: commodity device (CD) based and specialized hardware (SH) based. The CD-based systems can be implemented on commodity WiFi devices without requiring any hardware modifications. The SH-based systems use software defined radios, such as USRPs [1, 5], often along with specialized hardware, such as directional antennas or custom analog circuits. Next, we give an overview of prior work on both CD-based systems and SH-based systems. As our focus is on multi-user gesture recognition and due to space limitation, we only describe those prior systems that perform activity and gesture recognition.

2.1 CD-based Human Sensing Systems

WiFall detects a user's fall using features such as activity duration and rate of change in CSI values [17]. E-eyes generates histograms of the CSI values and uses them as features to recognize gestures such as brushing teeth, showering *etc.* [34]. HeadScan [14] and BodyScan [13] put antennas on user's body. HeadScan recognizes mouth related gestures such as coughing and eating using features that include mean, median, and standard deviation in CSI values. BodyScan recognizes gestures similar to E-eyes but using the shapes of CDFs of CSI values as features. WiFinger uses DWT coefficients of combined time series of CSI values to recognize finger gestures that differ in the number of extended and folded fingers [28]. WiGest distinguishes between gestures that give rise to three primitives in RSS values: rising edge, falling edge, and pause [6]. WiDraw tracks the hand of a user by monitoring the changes in the magnitudes of signals arriving at different angles from the hand [27]. WiAG proposes a translation function to enable position and orientation agnostic gesture recognition [30]. Unfortunately, none

of these systems recognize simultaneously performed gestures, as acknowledged in almost all of the papers where these systems were first proposed. FrogEye counts the number of people in a crowd using the hypothesis that the variance in the CSI values increases monotonically with the number of people in a crowd [37]. Although this hypothesis enables FrogEye to count the people, it does not provide any insights that could be used to enable the recognition of simultaneously performed gestures. WiHear recognizes a predefined set of spoken words and further proposed a zigzag cancelation technique to recognize spoken words of multiple users [31]. Unfortunately, the zigzag cancellation technique is not well-suited for the problem we are addressing because it captures only a small portion of each person's gesture and not the entire gesture.

2.2 SH-based Human Sensing Systems

AllSee uses an analog envelope-detector to extract the amplitude of the received TV and RFID signals to identify eight gestures performed by a single user at a time [20]. WiSee uses a USRP to extract micro-level doppler shifts in a carrier wave due to human movements to recognize nine different gestures performed by a single user at a time [25]. WiSee further proposed a preliminary method to recognize gestures in the presence of *interfering* users. WiSee does not recognize gestures of all users when multiple users perform them simultaneously; it recognizes gestures of one user and considers the remaining as *interfering* users. Furthermore, in the presence of interfering users, it can recognize only two gestures, push and pull, and the interfering users should not be close to the target user. In contrast, WiMU can recognize gestures of multiple users performing them simultaneously and is largely unaffected by the distance between the users.

Vital-Radio measures the variations in the phase of the signal reflected by the user to measure the heart and breathing rates of a user [9]. Wi-Vi can count the number of humans behind an obstacle and tracks the direction of their motion by emulating an inverse synthetic aperture radar using directional antennas [8]. WiTrack2.0 localizes multiple people simultaneously [7]. Although the objectives of Vital-Radio, Wi-Vi, and WiTrack2.0 are not to recognize human gestures, we have mentioned them because they handle "multiple" users. However, the methods employed by them to handle multiple users cannot be trivially adapted to enable the recognition of simultaneously performed gestures for two reasons: 1) they employ models that are geared towards quantifying the effects of human *presence* on wireless signals, and not of human *movements* on wireless signals, and 2) they use specialized hardware. The work on radio tomography imaging generalizes to multiple people, but requires much larger number of transmitters and receivers [36].

3 MULTI-USER MOVEMENT MODEL

In this section, we model the effects of simultaneous movements of multiple users on CSI values. The insights that we will develop from this model will guide the design of various components of WiMU such as to automatically determine the number of simultaneously performed gestures in a detected sample, to identify the start and end times of each gesture in the detected sample, and to generate virtual samples for any desired combination of gestures. Before deriving this model, we give a quick overview of what CSI is.

WiFi devices typically consist of multiple transmit (Tx) and receive (Rx) antennas. An 802.11n/ac MIMO channel between each Tx-Rx antenna pair comprises multiple sub-carriers. Let $X(f,t)$ and $Y(f,t)$ be the frequency domain representations of transmitted and received signals, respectively, on an OFDM subcarrier with frequency f at time t between a given Tx-Rx pair. The two signals are related as $Y(f,t) = H(f,t) \times X(f,t)$, where $H(f,t)$ represents the complex-valued channel frequency response (CFR) for subcarrier with frequency f at time t. Let N_{Tx} and N_{Rx} represent the number of Tx and Rx antennas, respectively, and let S represent the number of subcarriers between each Tx-Rx pair. Each CSI measurement comprises $S \times N_{Tx} \times N_{Rx}$ CFR values, one for each subcarrier between each Tx-Rx pair. As WiFi network interface cards (NICs) generate CSI measurements repeatedly, we essentially obtain $S \times N_{Tx} \times N_{Rx}$ time-series of CFR values. Onward, we will call each time-series of CFR values a *CSI-stream*.

3.1 CFR Power Model

Wang *et al.* modeled the effects of movements of a single user on CSI values [32]. Our model generalizes their model from single to multiple users. Wang *et al.* showed that although human movements affect both phase and magnitude of CFR values, the CFR phase measured on commodity WiFi devices has large error [32, 33]. The phase sanitization methods, such as [26], are not useful as they filter out any phase shifts due to human movements. Wang *et al.* further showed that while it is hard to precisely measure the CFR phase on commodity WiFi devices, it is simple to accurately measure the CFR power [32, 33]. Thus, next, we derive an expression that quantifies the effects of movements of multiple users on CFR power, represented by $|H(f,t)|^2$.

As surrounding objects reflect WiFi signals, a transmitted signal arrives at the receiver from multiple paths. Let $a_k(f,t)$ be the complex-valued representation of the initial phase and the attenuation of the k^{th} path at time t for a signal with carrier frequency f. Let $d_k(t)$ represent the length of the k^{th} path at time t. If a transmitted signal with carrier frequency f arrives at the receiver from multiple paths and the difference between the carrier frequencies of sender and receiver is Δf, then the aggregate CFR of the wireless channel is (see [29]):

$$H(f,t) = e^{-j2\pi\Delta f t} \sum_{\forall k} a_k(f,t) e^{-j\frac{2\pi d_k(t)}{c/f}}$$

We can represent this aggregate CFR as the sum of a dynamic and a static component. The dynamic component changes as the users move and is the sum of the CFRs of all those paths that arrive at the receiver after reflecting from the moving body parts of the users. The static component is not affected by the movement of any user and is the sum of the CFRs of all those paths that arrive at the receiver without reflecting from any moving body parts. Let N_a users simultaneously perform N_a gestures, and let $\mathcal{P}_i$ represent the set of all those paths that reflect from the moving body parts of the i^{th} user, where $1 \le i \le N_a$, and arrive at the receiver with or without any further reflections. Let $H_s(f)$ represent the static component of the aggregate CFR. We can express the aggregate CFR in the equation above as follows.

$$H(f,t) = e^{-j2\pi\Delta f t}\left(H_s(f) + \sum_{\forall k \in \bigcup_{i=1}^{N_a} \mathcal{P}_i} a_k(f,t) e^{-j\frac{2\pi d_k(t)}{c/f}}\right)$$

Let v_k represent the rate at which the length of the k^{th} path changes. We call v_k the *speed* of the k^{th} path. Thus, $d_k(t) = d_k(0) + v_k t$. By multiplying both sides of the equation above with their respective complex conjugates, and after some algebraic manipulations, we get the following expression for the aggregate CFR power.

$$|H(f,t)|^2 = \sum_{\forall k \in \bigcup_{i=1}^{Na} \mathcal{P}_i} |a_k(f,t)|^2 + |H_s(f)|^2 + \sum_{\forall k,l \in \bigcup_{i=1}^{Na} \mathcal{P}_i;\, k \neq l} 2|a_k(f,t)a_l(f,t)| \cos\left(\frac{2\pi(v_k - v_l)t}{c/f} + \frac{2\pi(d_k(0) - d_l(0))}{c/f} + \phi_{kl}\right) + \sum_{\forall k \in \bigcup_{i=1}^{Na} \mathcal{P}_i} 2|H_s(f)a_k(f,t)| \cos\left(\frac{2\pi v_k t}{c/f} + \frac{2\pi d_k(0)}{c/f} + \phi_{sk}\right) \quad (1)$$

where $2\pi f\,(d_k(0) - d_l(0))\,/(c/f) + \phi_{kl}$ and $2\pi f d_k(0)/c + \phi_{sk}$ are constants that represent the initial phase offsets for different paths.

3.2 Insights

Next, we present three insights from Eq. (1) and describe how these insights guided the design of various key components of WiMU.

Insight 1: Eq. (1) states that the aggregate CFR power is the sum of a constant offset and a set of sinusoids, where the frequencies of the sinusoids are the functions of the speeds of the paths. As the speed of a path is determined by the speed of the moving body part from which that path reflects, the frequencies in the aggregate CFR power are essentially the functions of the speeds of the moving body parts of the users. This leads to our first insight: *as different gestures involve movements of body parts at different speeds, each combination of gestures gives rise to a unique pattern of frequencies in the aggregate CFR power.* By first learning these patterns for any given combination of N_a gestures from the virtual samples of that combination, WiMU recognizes that combination of gestures at runtime by matching the patterns of frequencies in the aggregate CFR power with the learnt patterns.

Insight 2: The speed of each path reflecting from a stationary user is zero. As soon as a stationary user moves, the speeds of the paths reflecting from the now-moving parts of his/her body attain a non-zero speed. As per Eq. (1), all such paths either introduce certain new frequencies in the aggregate CFR power or significantly change the magnitudes of those frequencies if they were already present. Similarly, if a moving user stops, the speed of each path reflecting from this now-stationary user becomes zero, and any frequencies in the aggregate CFR power due to the speeds of these paths now vanish. This leads to our second insight: *increase (decrease) in the number of moving users results in an increase (decrease) in the number of frequencies with non-negligible magnitudes in the aggregate CFR power.* WiMU leverages this insight to detect the start/end of each gesture in a given set of simultaneously performed gestures by looking for a rapid increase/decrease in the number of frequencies with non-negligible magnitudes in the CFR power.

Insight 3: We call the frequencies introduced by the set of cos terms with argument $2\pi v_k t/(c/f)$ in Eq. (1) as *primary* frequencies and those introduced by the set of cos terms with argument $2\pi(v_k - v_l)t/(c/f)$ as *secondary* frequencies. We observe from Eq. (1) that the cosine terms with the argument $2\pi v_k t/(c/f)$ are being *linearly summed* over all non-zero-speed paths reflected from each user. This observation leads to our third insight: *whether a user performs a given gesture in solitude or simultaneously with other users, in both settings, his/her movements introduce the same set of primary frequencies in the aggregate CFR power.* WiMU leverages this insight to generate virtual samples for any desired combination of gestures by first extracting the primary frequencies from the aggregate CFR power of the individually collected real training samples of the gestures in that combination, and then aggregating them. The frequencies in the resulting virtual sample are very similar to the frequencies in a real sample that would result if real users performed that desired combination of gestures simultaneously.

4 WIMU – OVERVIEW

Next, we give an overview of WiMU's modules and describe how they interact to perform multi-user gesture recognition. Figure 1 shows WiMU's block diagram.

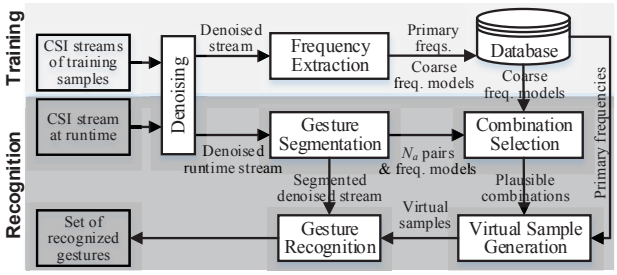

Figure 1: Block diagram of WiMU

1) Denoising: The denoising module removes the noise from the CSI-streams of both training and test samples. It takes the $S \times N_{Tx} \times N_{Rx}$ raw CSI-streams as input and converts each CFR value in each stream into CFR power by multiplying that value with its complex conjugate. Next, it applies principal component analysis based denoising technique, proposed in [30, 32], on these streams and gets $S \times N_{Tx} \times N_{Rx}$ new streams, called principal component streams. As demonstrated in [30, 32], the third stream has the highest human movement signal to noise ratio. Therefore, we use the third principal component stream for further processing, and call it the *denoised-stream*. As noise removal from CSI-streams is a well-studied problem in literature, we do not provide more details on the denoising module in the rest of this paper, and refer interested readers to [30, 32].

2) Frequency Extraction: This module extracts primary frequencies from training samples performed individually by only a single user. It takes the denoised-stream of any given training sample as input and slides a window of short time-width over it in small steps. At each window step, it applies short time fourier transform (STFT) on the portion of the denoised-stream covered by the window in that step, and selects all frequencies with non-negligible magnitudes. Next, it determines which of the selected frequencies are primary and which are secondary, and stores the values of the primary frequencies at each window step for the given training sample in a database. This module also generates and stores a coarse frequency-model of the gesture from each training sample. This module is the only module that processes the training samples. All modules that we will discuss next are used when users perform gestures simultaneously at runtime.

3) Gesture Segmentation: This module determines the start and end times of each gesture in a set of simultaneously performed gestures at runtime. It takes the denoised-stream at runtime as input

and continuously slides a window of short time-width over it. At each window step, it applies STFT and counts the number of frequencies with non-negligible magnitudes. As soon as it sees a rapid increase/decrease in the number of such frequencies, as per Insight 2 from Section 3.2, it declares that a gesture has started/ended. If the number of gestures in a set of simultaneously performed gestures is $N_a \geq 1$, then as soon as the last of the N_a users finishes his/her gesture, this module outputs N_a pairs of times, one pair for each gesture. The first item in any given pair is the start time of a gesture and the second item is the end time of that gesture. With each pair, this module also outputs a coarse frequency-model of the gesture associated with that pair.

4) Gesture Combination Selection: To explain the need for this module, consider an example where six users simultaneously perform gestures and each user can perform any of five predefined gestures. The number of combinations in which these six users can perform gestures is $5^6 = 15625$. This is very large and can lead to low gesture recognition accuracy. Thus, for any given test sample containing a set of simultaneously performed gestures, we must identify fewer but plausible combinations, which is what this module outputs. It takes as input the coarse frequency-models of the N_a gestures outputted by the gesture segmentation module, compares each model against the frequency-models of each gesture outputted by the frequency extraction module, and eliminates the implausible combinations.

5) Virtual Sample Generation: This module generates virtual samples for each combination of gestures outputted by the combination selection module. For each gesture in a given combination of gestures for which virtual samples are desired, this module takes the primary frequencies at each window step from a real sample of that gesture as input. It slides a window of the same time-width as in the frequency extraction module over the duration of the virtual sample. At each window step, as per Insight 3, it inserts the primary frequencies from the appropriate windows of the real samples of all those gestures that should be present in the virtual sample at that window step. At each step, it also inserts the secondary frequencies. The output of this module is a sequence of sets of frequencies, one at each window step, for the duration of the virtual sample. These sets are very similar to the sets of frequencies that would be present in a denoised-stream resulting from real users performing the given combination of gestures.

6) Gesture Recognition: This module takes as input all virtual samples and the denoised-stream between the start time of the first gesture and the end time of the last gesture. It slides a window of same time-width on this denoised-stream, applies STFT at each window step, and selects the set of all frequencies that have non-negligible magnitudes. Next, it compares the frequencies in the resulting sequence of sets with the frequencies in the sequence of sets from each virtual sample, and as per Insight 1, declares the denoised-stream to contain that combination of gestures whose virtual samples achieve the highest match.

5 FREQUENCY EXTRACTION

In this section, we describe how WiMU extracts the primary frequencies from any given training sample of duration t_{tr} performed individually by a single user. For this, WiMU slides a window of width w seconds over the denoised-stream in small s second steps, where $w = 200$ms and $s = 5$ms in our implementation. At each window step, it applies an n_{FT} point STFT, and obtains a vector containing the magnitudes of frequencies. We represent the frequency-vector obtained at the j^{th} window step with $\mathbf{F}^j$, where $j \in [0, \lceil (t_{\text{tr}} - w)/s \rceil]$. The key challenge here is to determine which frequencies in any given frequency-vector are primary and which are secondary. To facilitate distinguishing between these two types of frequencies, we set n_{FT} equal to the CSI sampling rate so that each value in the frequency-vector covers a range of 1Hz. In our implementation, as our sampling rate is ≈1kHz, we used $n_{\text{FT}} = 1024$. A 1024 point STFT results in a frequency-vector of length 512. As human movements do not introduce frequencies >300Hz in the aggregate CFR power [32], WiMU keeps the first 300 values and discards the remaining because at 1kHz CSI sampling rate, the remaining values cover the frequency range of ≈300Hz to 500Hz. Thus, the length of each $\mathbf{F}^j$ is 300. Next, we describe how WiMU processes each frequency-vector $\mathbf{F}^j$ to determine which frequencies in it are primary and which are secondary.

5.1 Primary / Secondary Frequency Separation

As per Eq. (1), if the number of paths whose lengths change at different rates due to a human movement is p, then that human movement introduces p primary frequencies and $\binom{p}{2}$ secondary frequencies in the aggregate CFR power; thus, $p + \binom{p}{2}$ frequencies in total. WiMU uses this relationship between p and the total number of frequencies to determine the number of primary frequencies in any given frequency-vector $\mathbf{F}^j$. More specifically, it first compares the magnitudes of all frequencies in $\mathbf{F}^j$ with a baseline threshold, T_{bl}, to filter out the frequencies that appear due to leftover noise, and then counts the number of peaks in $\mathbf{F}^j$ that have magnitudes $>T_{\text{bl}}$. Next, it uses bisection search to find the value of p for which $p + \binom{p}{2}$ is closest to this count and uses this value of p as an estimate of the number of primary frequencies in $\mathbf{F}^j$. We will describe how WiMU automatically calculates the value of T_{bl} in Section 5.3.

Having estimated the number of primary frequencies, WiMU next determines which frequencies in $\mathbf{F}^j$ are primary and which are secondary. For this, it first selects $p + \binom{p}{2}$ frequency values corresponding to the peaks with the highest magnitudes in $\mathbf{F}^j$ and then performs $\binom{p+\binom{p}{2}}{p}$ iterations. In each iteration, it selects a unique combination of p frequency values as a potential set of primary frequencies and the remaining $\binom{p}{2}$ frequency values as a potential set of secondary frequencies, and calculates a penalty score for this pair of sets. After completing all iterations, it selects the pair of sets with the lowest penalty score as the correct sets of primary and secondary frequencies in the given $\mathbf{F}^j$.

To calculate the penalty score for any given pair of potential sets of primary and secondary frequencies, WiMU first calculates the absolute pairwise difference between all p frequency values in the given set of primary frequencies, and then sorts the resulting $\binom{p}{2}$ difference values. We represent these sorted $\binom{p}{2}$ difference values as a vector $\mathbf{V}$. Let $\tilde{\mathbf{V}}$ be the vector comprising sorted $\binom{p}{2}$ values in the given set of secondary frequencies. Next, WiMU calculates the distance between these two vectors and assigns this distance as the penalty score to the given pair of sets. The motivation behind using this distance as the penalty score is that if all frequencies in the given set of primary frequencies are indeed primary, then as per Eq.

(1), the $\binom{p}{2}$ pairwise frequency difference values calculated from it will be almost the same as the $\binom{p}{2}$ observed frequency values in the given set of secondary frequencies, resulting in penalty close to 0.

WiMU essentially performs exhaustive search to distinguish between primary and secondary frequencies. This makes the frequency extraction module computationally the most expensive module of WiMU. Fortunately, this module is executed offline and only once at the time of acquiring training samples. The computational complexity of this module, therefore, does not impact the runtime gesture recognition speed of WiMU.

5.2 Coarse Frequency-Models of Gestures

As per Insight 1 from Section 3.2, a gesture can be coarsely modelled by quantifying *how much* of each frequency does this gesture introduce in the CFR power. WiMU generates coarse frequency-models of each gesture from each of its training samples. It generates the model from any given training sample of duration t_{tr} as $\sum_{j=0}^{\lceil (t_{\text{tr}}-w)/s \rceil} \mathbf{F}^j / \lceil (t_{\text{tr}} - w)/s \rceil$. The summation term quantifies *how much* of each frequency does this gesture introduce in the CFR power. As the durations of different training samples can be different, we normalize the summation with the number of window steps over which the frequency-vectors are summed. Let N_p represent the number of predefined gestures that WiMU should recognize, and let s_k represent the number of training samples that WiMU has for the k^{th} predefined gesture, where $1 \leq k \leq N_p$. We represent the coarse frequency-model generated from the l^{th} training sample of the k^{th} predefined gesture with $\mathbf{M}^{\text{tr}}_{k,l}$, where $1 \leq l \leq s_k$.

5.3 Setting the Baseline Threshold

WiMU uses the baseline threshold, T_{bl}, to determine whether any given frequency in any given frequency-vector is due to a human movement or just noise. As per Insight 2, in the absence of movements, the magnitudes of all frequencies in the frequency-vector should theoretically be 0. However, in practice, frequencies exhibit small non-zero magnitudes. Keeping this in view, a simple way to set the value of T_{bl} is to identify a portion in the given denoised-stream where no user performed any movement, and use the magnitudes of frequencies in the frequency-vectors for this portion to set T_{bl}.

In the absence of any human movements, the magnitudes of *all* frequencies are close to 0, and thus relatively uniformly distributed. In the presence of human movements, the magnitudes of some frequencies are higher than the others, and thus non-uniformly distributed. Consequently, the coefficient of variation (cv) of the values in any given frequency-vector is much smaller in the absence of human movements than in the presence. In our data set, we never observed $cv > 0.1$ in the absence of human movements and < 0.5 in the presence. Thus, $cv < 0.1$ for any frequency-vector is a robust indicator that the time duration over which that frequency-vector was obtained did not have any movements.

Based on the discussion above, to set T_{bl}, WiMU continuously looks for 10 consecutive frequency-vectors with $cv < 0.1$. Every time it finds them, it calculates the mean, μ, and standard deviation, σ, of all values in these 10 consecutive frequency-vectors, and sets the threshold using the three-sigma rule as $T_{\text{bl}} = \mu + 3\sigma$. To ensure that each training sample has a portion in the denoised-stream that WiMU can use to calculate T_{bl}, we recommend that the user stay stationary for about a second before providing each sample.

6 GESTURE SEGMENTATION

In this section, we describe how WiMU processes the denoised stream at runtime to detect that some users have performed a set of gestures simultaneously and to determine the start and end times of each gesture in that set. For this, WiMU performs three steps. In the first step, it identifies the start time and the end time of the entire set. In the second step, it selects the portion of the denoised-stream between these start and end times and processes it to identify all those times at which gestures either started or ended. If the set of simultaneously performed gestures comprises $N_a \geq 1$ gestures, then in this step, it identifies N_a start times and N_a end times. In the last step, it pairs up the start times with end times such that each of the resulting N_a pairs contain the start and end times of one of the N_a gestures. Next, we describe these three steps in detail.

6.1 Detecting a Set of Simultaneous Gestures

To detect the start and end time of an entire set of simultaneously performed gestures, WiMU continuously slides the same $w = 200$ms window in $s = 5$ms steps on the denoised stream at runtime. At each step, it obtains a frequency-vector and calculates its cv. As long as no user performs any gesture, cv stays below 0.1 (see Section 5.3). As soon as cv exceeds 0.1 for 10 consecutive window steps, WiMU takes this as an indication that some user has started a gesture, and records the time at the middle of the window in the first of these 10 steps as the start time. It continues to calculate cv at subsequent window steps and as soon as cv falls back below 0.1 for 10 consecutive window steps, WiMU takes this as an indication that all users have completed gestures, and records the time at the middle of the window in the first of these 10 steps as the end time.

6.2 Detecting the Gesture Start & End Times

While we observe a rapid increase (decrease) in cv when the first gesture starts (last gesture ends), the jumps in cv are not prominent for the gestures that start and end in-between. The reason being that after the start of the first gesture, the frequencies introduced by the start of the subsequent gestures increase the means of the frequency-vectors but do not increase the standard deviations significantly. Thus, although the cv based method is robust in detecting the start of the first gesture and the end of the last, it is not reliable in detecting the starts and ends of the gestures in-between.

Leveraging the Insight 2, to detect gesture start/end, WiMU looks for rapid increase/decrease in the number of frequencies with magnitudes $>T_{\text{bl}}$ in frequency-vectors. WiMU continuously recalculates T_{bl} at runtime from the non-active portions of the denoised stream, as described in Section 5.3. WiMU takes the portion of the denoised stream between the start and end times calculated in Section 6.1, slides the same $w = 200$ms window in $s = 5$ms steps, and obtains $\mathbf{F}^j$ at each step, where $j \in [0, \lceil (t_{\text{t}} - w)/s \rceil]$ and t_t is the duration between the start and end times calculated in Section 6.1. As the frequencies in $\mathbf{F}^0$ are introduced only by the first user that just started a gesture, WiMU counts the number of frequencies with magnitudes $>T_{\text{bl}}$ in $\mathbf{F}^0$, and uses this count as an estimate of the number of frequencies each user's movements introduce in the aggregate CFR power. We represent this count with $n_{\mathbf{F}^0}$. WiMU estimates $n_{\mathbf{F}^0}$ instead of using a constant value for it because it can vary across environments depending on the number of strong reflecting objects in that environment.

After estimating $n_{\mathbf{F}^0}$, WiMU sequentially processes each frequency-vector from $\mathbf{F}^1$ through $\mathbf{F}^{\lceil (t_t - w)/s \rceil}$. In processing any $\mathbf{F}^j$, WiMU first calculates $n_{\mathbf{F}^j}$, the number of frequencies in $\mathbf{F}^j$ with magnitudes $>T_{\text{bl}}$. Next, it compares $n_{\mathbf{F}^j}$ with an exponentially weighted moving average, $\overline{\mu}^{\mathbf{F}}_{j-1}$, where $\overline{\mu}^{\mathbf{F}}_{j-1}=(\overline{\mu}^{\mathbf{F}}_{j-2} + n_{\mathbf{F}^{j-1}})/2$, and $\overline{\mu}^{\mathbf{F}}_{0}=n_{\mathbf{F}^0}$. Based on the number of starts and ends of gestures that WiMU has detected from the frequency-vectors before $\mathbf{F}^j$, let i represent the number of gestures that WiMU estimates to be currently being performed when it starts processing $\mathbf{F}^j$. If $n_{\mathbf{F}^j} - \overline{\mu}^{\mathbf{F}}_{j-1} > 0.9^i \times n_{\mathbf{F}^0}$, WiMU takes this rapid increase in the number of frequencies in $\mathbf{F}^j$ to be the indication of the start of a new gesture, and records the index j of this $\mathbf{F}^j$ as a gesture start time. Similarly, if $\overline{\mu}^{\mathbf{F}}_{j-1} - n_{\mathbf{F}^j} < 0.9^{i-1} \times n_{\mathbf{F}^0}$, WiMU takes this rapid decrease in the number of frequencies to be the indication of the end of an existing gesture, and records the index j as a gesture end time.

The motivation behind comparing $n_{\mathbf{F}^j}$ with $\overline{\mu}^{\mathbf{F}}_{j-1}$ instead of $n_{\mathbf{F}^{j-1}}$ is to avoid erroneous detections due to abrupt changes in the number of frequencies at the times when gestures start and end. We empirically determined that the weighting parameter of 50% in calculating $\overline{\mu}^{\mathbf{F}}_{j-1}$ results in high detection accuracy. Note that after the start of any new gesture, within four window steps, *i.e.*, 20ms, the contribution of the number of frequencies in the frequency-vectors before the start of this gesture to the exponentially weighted moving average falls down to just $0.5^4 = 6.25\%$, and the average reaches the number of frequencies in the frequency-vectors after the start of the new gesture. Consequently, WiMU can distinguish between one gesture's start and another gesture's end, and vice versa, as long as the two events are separated by just 20ms. To distinguish between two consecutive starts (ends), due to the 5ms window step, WiMU requires them to be separated by just 5ms. In practice, a user starting a gesture and another ending a gesture with a time difference of less than 20ms, or two users starting (ending) gestures with a time difference of less than 5ms are all rare events.

The motivation behind comparing the difference between $n_{\mathbf{F}^j}$ and $\overline{\mu}^{\mathbf{F}}_{j-1}$ with $0.9^i \times n_{\mathbf{F}^0}$ instead of $n_{\mathbf{F}^0}$ to detect the start of new gestures is that when some users are already performing gestures, some of the frequencies that a new gesture introduces in the aggregate CFR power may already be present in it due to the gestures already being performed. Thus, if WiMU were to always look for a jump of $n_{\mathbf{F}^0}$, it may miss the start of some gestures. The same motivation lies behind comparing the difference between $\overline{\mu}^{\mathbf{F}}_{j-1}$ and $n_{\mathbf{F}^j}$ with $0.9^{i-1} \times n_{\mathbf{F}^0}$ to detect the end of an existing gesture. In the rare case when the number of gesture starts that WiMU detects are fewer than the number of gesture ends, it reprocesses all frequency-vectors $\mathbf{F}^1$ through $\mathbf{F}^{\lceil (t_t - w)/s \rceil}$ again and looks for gesture starts (but not ends this time) using a reduced multiplication factor for $n_{\mathbf{F}^0}$ from 0.9^i to 0.85^i. It continues to reduce the multiplication factor in the steps of 0.05 and reprocesses all frequency-vectors until the number of gesture starts that it detects become equal to the number of gesture ends. Similarly, if the number of gesture ends that it detects are fewer than the number of gesture starts, it uses the same approach of reducing the multiplication factor in the steps of 0.05 and detecting the gesture ends again. Finally it passes N_a indices corresponding to N_a gesture starts and another N_a indices corresponding to N_a gesture ends to the third step, described next.

6.3 Pairing the Gesture Start & End Times

In this step, WiMU pairs up the N_a end times with the N_a start times such that each of the resulting N_a pairs contains the start time and end time of a unique gesture. To do this, WiMU leverages the Insight 1 from Section 3.2 that each gesture introduces a unique set of frequencies in the aggregate CFR power. Each time WiMU detects the start of a new gesture, it quantifies the effect of this gesture on each frequency in the aggregate CFR power, and stores this information as a coarse frequency-model. Every time it detects the end of a gesture, it again quantifies the effect of the end of the gesture on each frequency, and compares the resulting coarse frequency-model with the frequency-models that it generated at gesture starts. It pairs up the end time of this gesture with that start time whose frequency-model matches this frequency-model most.

Recall that on completing step 2 (described in Section 6.2), WiMU obtains $2N_a$ indices of frequency-vectors: N_a corresponding to gesture start events and N_a corresponding to gesture end events. Let j_i represent the i^{th} index after these $2N_a$ indices are chronologically arranged, where $1 \leq i \leq 2N_a$ and $j_i \in [0, \lceil (t_t - w)/s \rceil]$. WiMU sequentially processes the events at these $2N_a$ indices, starting at j_1 and sequentially working its way to j_{2N_a}. Next, we describe how WiMU processes the event at any arbitrary index j_i.

If the event at the index j_i is a gesture start, WiMU creates a frequency-model, $\mathbf{M}^s_{j_i}$, that quantifies the effect of this new gesture on each frequency in the aggregate CFR power as:

$$\mathbf{M}^s_{j_i} = \left(\sum_{j=j_i}^{j_{i+1}-1} \mathbf{F}^j \right) / (j_{i+1} - j_i) - \left(\sum_{j=j_{i-1}}^{j_i - 1} \mathbf{F}^j \right) / (j_i - j_{i-1}) \quad (2)$$

The second summation quantifies the contribution of all gestures that were being performed before the new gesture started at j_i while the first quantifies the contribution of all those plus the new. As the indices are not equally spaced in time, we normalize the summations with the durations over which they are summed. WiMU inserts $\mathbf{M}^s_{j_i}$ into a set $\mathcal{H}$ that contains frequency-models of those gestures for which WiMU has detected starts but has not yet detected ends.

Similarly, if the event at the index j_i is a gesture end, WiMU creates a frequency-model, $\mathbf{M}^e_{j_i}$, that quantifies the effect of the gesture that just ended on each frequency in the CFR power.

$$\mathbf{M}^e_{j_i} = \left(\sum_{j=j_{i-1}}^{j_i - 1} \mathbf{F}^j \right) / (j_i - j_{i-1}) - \left(\sum_{j=j_i}^{j_{i+1}-1} \mathbf{F}^j \right) / (j_{i+1} - j_i) \quad (3)$$

Next, it pairs up this index j_i with that index j_l that minimizes the magnitude of $\mathbf{M}^e_{j_i} - \mathbf{M}^s_{j_l}$, where $\mathbf{M}^s_{j_l} \in \mathcal{H}$. Finally, it removes $\mathbf{M}^s_{j_l}$ from the set $\mathcal{H}$, and returns the ordered pair (j_l, j_i). This ordered pair contains the start and end times of a unique gesture among the N_a gestures. Along with this ordered pair, it also returns an aggregate coarse frequency-model, $\mathbf{M}^U_{j_l \leftrightarrow j_i} = (\mathbf{M}^s_{j_l} + \mathbf{M}^e_{j_i})/2$, of the unknown gesture associated with (j_l, j_i).

7 GESTURE COMBINATION SELECTION

Recall that N_p represents the number of predefined gestures. If the number of gestures that the users performed simultaneously is N_a, then these gestures could be any of the $N_a^{N_p}$ combinations of the N_p gestures. In other words, the gesture associated to each pair of start and end times can be any of the predefined N_p candidate gestures. The objective of this module is to reduce the number of these candidate gestures for each pair.

To reduce the number of candidates for any pair (j_l, j_i), WiMU first max-normalizes $\mathbf{M}^U_{j_l \leftrightarrow j_i}$ associated with this pair by dividing it with $\texttt{max}(\mathbf{M}^U_{j_l \leftrightarrow j_i})$. Let us represent any max-normalized coarse frequency-model $\mathbf{M}$ with $\widehat{\mathbf{M}}$. Next, it matches $\widehat{\mathbf{M}}^U_{j_l \leftrightarrow j_i}$ with max-normalized coarse frequency-models of each predefined gesture, calculated by the frequency extraction module, and eliminates all those gestures with which the match is insignificant. More specifically, for any predefined gesture k with s_k training samples, if the value of $||\widehat{\mathbf{M}}^U_{j_l \leftrightarrow j_i} - \frac{1}{s_k} \sum_{l=1}^{s_k} \widehat{\mathbf{M}}^{\text{tr}}_{k,l}||_2 > 0.5\sqrt{300}$, WiMU removes that gesture k from the set of possible candidates. The motivation behind comparing with $\sqrt{300}$ is that as the length of any frequency-model is 300, the largest magnitude of the difference between any two max-normalized coarse frequency-models can't exceed $\sqrt{300}$. The motivation behind multiplying $\sqrt{300}$ with 0.5 is to discard all those predefined gestures that have less than 50% match with the gesture associated with (j_l, j_i). Based on the shortlisted candidate gestures for each of the N_a pairs of start and end times, WiMU lists all the combinations of gestures that are possible, and passes this list to the virtual sample generation module to create virtual samples for each combination. A combination is simply a list of all the N_a pairs with a candidate gesture listed with each pair.

8 VIRTUAL SAMPLE GENERATION

To generate a virtual sample for any given combination, WiMU performs three steps. In the first step, for each of the N_a gestures listed with the N_a pairs in that combination, it picks a randomly selected training sample of that gesture. In the second step, for each pair, if the duration represented by the pair does not match with the duration of the training sample that WiMU picked for the gesture listed with that pair, WiMU adjusts the duration of the training sample to match with the duration represented by the pair. In the last step, WiMU uses the N_a duration-adjusted training samples to generate a virtual sample for the given combination.

8.1 Matching Sample Durations

To make the duration of any given pair (j_l, j_i) consistent with the duration of the training sample that WiMU picked for that pair, WiMU expands/contracts the training sample. Recall from Section 5 that WiMU already has extracted a set of primary frequencies from each of the frequency-vectors of this training sample. Let us represent the j^{th} set of primary frequencies with $\mathcal{F}^j$, where $0 \leq j \leq \lceil (t_{\text{tr}} - w)/s \rceil$ and t_{tr} is the duration of this training sample. Let $D^{\text{p}} = j_i - j_l$ represent the duration of the pair (j_l, j_i) and $D^{\text{tr}} = \lceil (t_{\text{tr}} - w)/s \rceil + 1$ represent the duration of the training sample in terms of the number of window steps. If $D^{\text{p}} > D^{\text{tr}}$, WiMU needs $D^{\text{p}} - D^{\text{tr}}$ more sets of primary frequencies, which it obtains by first replicating every $\left((D^{\text{p}} - D^{\text{tr}})/D^{\text{tr}}\right)^{\text{th}}$ set and then chronologically renumbering the indices of all sets (the index of set $\mathcal{F}^j$ is j). Similarly, if $D^{\text{p}} < D^{\text{tr}}$, WiMU needs to remove $D^{\text{tr}} - D^{\text{p}}$ sets of primary frequencies, which it does by removing every $\left((D^{\text{tr}} - D^{\text{p}})/D^{\text{tr}}\right)^{\text{th}}$ set and renumbering the indices of the remaining sets.

Expanding/contracting a training sample implies that the gesture was performed at a slower/faster rate, which in turn implies that the speeds of signal paths reduced/increased, respectively. Thus, the frequencies in the aggregate CFR power must be decreased/increased depending on the extent of expansion/contraction of the training sample. To do this, WiMU multiplies each value in each of the D^{p} sets of primary frequencies with t^{tr}/t^{p}, where t^{p} is the duration represented by the pair (j_l, j_i) in terms of absolute time, and is equal to $(D^{\text{p}} - 1)s + w$.

8.2 Generating a Virtual Sample

Recall from Section 6.3 that $j_1 = 0$ and $j_{2N_a} = \lceil (t_{\text{t}} - w)/s \rceil$ are the indices of the frequency-vectors where the first gesture started and the last gesture ended, respectively. Thus, the virtual sample should also have $\lceil (t_{\text{t}} - w)/s \rceil + 1$ frequency-vectors. Let $\mathbf{F}^j_{\text{B}}$ represent the j^{th} *binary valued* frequency-vector that WiMU will generate, where $0 \leq j \leq \lceil (t_{\text{t}} - w)/s \rceil$. Unlike a *regular* frequency-vector that contains the magnitudes of frequencies in the CFR power, a *binary valued* frequency-vector only indicates whether a frequency is present in the CFR power or not by setting the value in the vector corresponding to that frequency 1 or 0, respectively.

To generate any arbitrary $\mathbf{F}^j_{\text{B}}$, where $0 \leq j \leq \lceil (t_{\text{t}} - w)/s \rceil$, WiMU first sets all 300 values in it to 0, and then processes each of the N_a pairs. In processing any given pair (j_l, j_i), WiMU first checks whether $\mathbf{F}^j_{\text{B}}$ lies in the range covered by (j_l, j_i) by checking if $j_l \leq j < j_i$. If yes, it retrieves the set of primary frequencies $\mathcal{F}^{j-j_l}$ from the training sample of the gesture associated with the pair (j_l, j_i), and sets all values in $\mathbf{F}^j_{\text{B}}$ corresponding to the frequencies in this set to 1. In other words, WiMU inserts all primary frequencies from the appropriate location of the training sample of the gesture associated with this pair into $\mathbf{F}^j_{\text{B}}$. After processing all N_a pairs, as per Insight 3 from Section 3.2, WiMU has inserted all primary frequencies into $\mathbf{F}^j_{\text{B}}$ from all those gestures that should be present in the virtual sample at index j. To insert all secondary frequencies into $\mathbf{F}^j_{\text{B}}$, WiMU takes pairwise differences of all primary frequencies that it just inserted into $\mathbf{F}^j_{\text{B}}$ and sets all values in $\mathbf{F}^j_{\text{B}}$ corresponding to these difference values to 1. This completes the generation of $\mathbf{F}^j_{\text{B}}$. After WiMU has generated all binary valued frequency-vectors, it concatenates them to obtain a binary matrix of size $300 \times \lceil (t_{\text{t}} - w)/s \rceil + 1$. This binary matrix is our virtual sample. We use the notation $\mathbf{B}_v$ to represent the binary matrix corresponding to any arbitrary virtual sample.

9 GESTURE RECOGNITION

To recognize the gestures in a given set of simultaneously performed gestures, WiMU first converts the denoised-stream between the start and end times of this set, calculated in the gesture segmentation module, into the binary matrix format, represented by $\mathbf{B}_t$. Next, it compares this matrix with the virtual sample matrices $\mathbf{B}_v$. For this, WiMU concatenates all frequency-vectors $\mathbf{F}^j$, calculated in the gesture segmentation module and gets a $300 \times \lceil (t_{\text{t}} - w)/s \rceil + 1$ matrix. Next, it compares each frequency magnitude value in this matrix with the threshold T_{bl}. If any frequency magnitude value is $> T_{\text{bl}}$, that frequency is present in the aggregate CFR power and thus, WiMU replaces that value with 1; otherwise, with 0. This finally results in the desired binary matrix $\mathbf{B}_t$.

To identify the simultaneously performed gestures, WiMU compares $\mathbf{B}_t$ with all virtual samples of each combination and calculates a similarity score for each combination. It declares the set of simultaneously performed gestures to be comprised of that combination

that achieves the highest score. To calculate the similarity score for any given combination, WiMU measures the Jaccard similarity coefficient [4, 16] of $\mathbf{B}_t$ with each virtual sample of that combination, calculates the average of all Jaccard coefficients, and assigns this average as the similarity score to that combination.

The Jaccard coefficient between the two matrices $\mathbf{B}_t$ and $\mathbf{B}_v$ is the ratio of the number of frequencies that are present in both matrices to the number of frequencies that are present in at least one of them. The motivation behind using Jaccard coefficient is twofold. First, it ignores those frequencies that are not present in either the virtual sample or the test sample. This is useful because such frequencies do not provide any information about the "similarity" between the two samples. Second, the calculation of Jaccard coefficient is computationally simple: take logical AND of the two matrices; take logical OR of the two matrices; divide the sum of all values in the first resultant with the sum of all values in the second.

10 IMPLEMENTATION & EVALUATION

We implemented WiMU on a commodity PC (8 core Intel Xeon processor, 16GB RAM). We used the tool presented in [15] to acquire CSI measurements in the 5GHz band from an Intel 5300 WiFi NIC connected with $N_{Rx} = 3$ omnidirectional antennas. We used NETGEAR R6700 access point (AP) using $N_{Tx} = 1$ omnidirectional antenna and pinged it every 1ms to achieve a 1000 samples/sec sampling rate. Next, in Sec. 10.1, we describe how we collected samples for various combinations of gestures. In Sec. 10.2, we compare the virtual samples generated by WiMU with real samples. In Sec. 10.3, we evaluate WiMU's accuracy in detecting the start and end times of gestures. In Sec. 10.4, we study the extent to which the combination selection module reduces the number of combinations. In Sec. 10.5, we evaluate WiMU's accuracy in recognizing the gestures performed by up to 6 users simultaneously. In Sec. 10.6, we evaluate the accuracy of WiMU on gestures performed individually, and compare the results with prior art. In sections 10.7, 10.8, and 10.9, we study the impacts of user's height and weight, distance between users, and static variations in the environment, respectively, on the accuracy of WiMU. Finally, in Sec. 10.10, we evaluate the latency of WiMU in processing training and test samples.

10.1 Data Collection

We collected training samples from 10 volunteers, 3 males and 7 females, with ages ranging from 20 to 30, heights from 5' 5" to 6' 3", and weights from 135lb to 220lb. We assigned them unique IDs, v_1 through v_{10}. We collected samples for six randomly chosen gestures: open and close door (O), circular arm movement (C), push and pull arm (P), sit down and stand up (S), kicking (K), and brushing teeth (B). For gestures O and B, the volunteers did not open and close an actual door or brush their teeth in real, rather only performed the movements of doing these gestures. We collected samples in a 25ft×16ft room. The AP and the receiver antennas were placed 5 ft apart on a table that is situated adjacent to and in the middle of the 25ft wall. The room contains a steel closet, 7 chairs, and 4 tables.

We first asked each volunteer to individually provide 10 samples for each of the six gestures at random positions of his/her choosing. Next, we collected samples for N_a simultaneously performed gestures, where $N_a \in \{2, 3, 4, 5, 6\}$. For each value of N_a, we first selected three combinations of N_a gestures to be performed. Next, for each combination, we randomly picked N_a volunteers, assigned a gesture to each volunteer, and asked them to pick any positions of their choosing in the room and provide 30 samples. In the first 15 samples, we asked the volunteers to start their assigned gestures in parallel on a voice prompt. In the remaining 15 samples, we assigned a sequence number to each volunteer and instructed them that the volunteer with sequence number i should start after the volunteer $i-1$ starts but before the volunteer $i-1$ finishes.

Table 1 summarizes the combinations of gestures for which we collected the samples and the IDs of the volunteers that performed the gestures. Each combination also has an ID. We collected 1050 samples, 600 for gestures performed individually and 450 for gestures performed simultaneously. These 450 samples contain 1800 performances of the six gestures. For each sample in our data set, we have the manually annotated ground truth values of the number of gestures in that sample, the names and sequence of gestures, and the start and end times. The data collection spanned 14 days. On each day, the location of the furniture was randomly changed to incorporate the effects of environmental changes into the samples.

ID	N_a	Volunteer IDs and Gestures					
C1	2	v_4,O	v_2,O	–	–	–	–
C2	2	v_2,O	v_4,S	–	–	–	–
C3	2	v_2,B	v_4,K	–	–	–	–
C4	3	v_3,S	v_2,S	v_1,S	–	–	–
C5	3	v_2,P	v_{10},C	v_4,K	–	–	–
C6	3	v_2,P	v_4,B	v_{10},O	–	–	–
C7	4	v_5,C	v_{10},C	v_7,C	v_3,C	–	–
C8	4	v_{10},S	v_3,P	v_7,C	v_5,K	–	–
C9	4	v_3,K	v_9,K	v_8,B	v_2,O	–	–
C10	5	v_6,P	v_{10},P	v_3,P	v_7,P	v_5,P	–
C11	5	v_3,P	v_{10},S	v_5,O	v_6,B	v_7,B	–
C12	5	v_4,P	v_1,B	v_{10},S	v_8,P	v_5,K	–
C13	6	v_8,K	v_2,K	v_{10},K	v_5,K	v_1,K	v_4,K
C14	6	v_1,B	v_2,B	v_5,P	v_{10},P	v_8,K	v_4,O
C15	6	v_4,K	v_{10},K	v_8,C	v_2,O	v_5,C	v_1,B

Table 1: Combinations of gestures in our data set

10.2 Comparison of Real vs. Virtual Samples

In this section, we study how similar the virtual samples generated by WiMU for any given combination of gestures are to the real samples of that combination. To measure this, for each combination of gestures corresponding to each value of N_a in Table 1, we generated 100 virtual samples and calculated the Jaccard coefficient between each real sample and the corresponding 100 virtual samples. For each N_a, as we have 3 combinations of gestures with 30 samples per combination, we obtain $3 \times 30 \times 100 = 9000$ Jaccard coefficient values. Figure 2 plots the CDF of these 9000 values for each value of N_a. We observe from this figure that when $N_a = 2$, the average Jaccard coefficient between real and virtual samples is 86%, *i.e.*, on average, the real and virtual samples, are 86% similar. As N_a increases, the similarity decreases. However, even when $N_a = 6$, the average Jaccard coefficient is > 80%, which is large enough to enable WiMU to distinguish across different combinations.

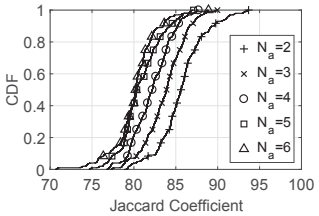

Figure 2: CDFs of Jaccard coefficients between real and virtual samples

10.3 Gesture Segmentation Accuracy

To evaluate the gesture segmentation module, we took all 1050 samples and calculated the number of gestures and the start and end times in each sample using the method described in Section 6, and compared with the ground truth. Figure 3 plots the percentage of samples corresponding to each value of N_a in which WiMU calculated the number of gestures correctly. We observe that WiMU correctly detected the number of gestures in at least 97% of the samples for any number of simultaneously performed gestures. We see a minor drop for the parallel started samples compared to the sequentially started samples because, infrequently, the gesture start times of some volunteers were so close that WiMU could not either distinguish between them or generate a reliable frequency-model.

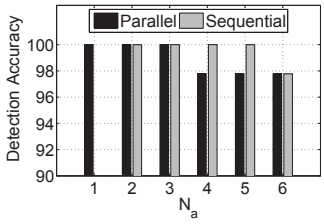

Figure 3: Percentage of samples with correctly detected N_a

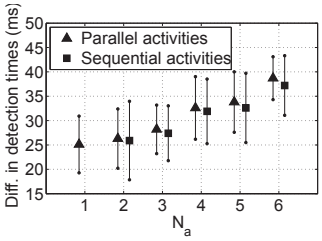

Figure 4: Difference in detected & ground truth start & end times

Figure 4 plots the average and standard deviation of the difference in the start/end times calculated by WiMU and the ground truth. We see that the average difference lies between just 25ms to 40ms, which is most likely due to the human factor in the manually annotated ground truth. Nonetheless, considering that a gesture generally takes more than a second to be performed, such a small difference is negligible. Thus, WiMU is robust in identifying the number of gestures and start and end times for at least six simultaneously performed gestures (and probably more, but we have not yet evaluated WiMU on more than six).

10.4 Combination Selection Performance

For each sample of each combination in our data set with N_a gestures, we calculated the ratio of 6^{N_a} (*i.e.*, the number of all possible combinations) with the number of combinations outputted by the combination selection module for that sample. Figure 5 shows a box plot for each value of N_a, where each box plot is obtained from 90 ratio values (recall that our data set has 90 samples for each value of N_a). We observe from this figure that the combination selection module reduces the number of combinations by an average of at least one order, and the order of reduction increases with the increase in N_a.

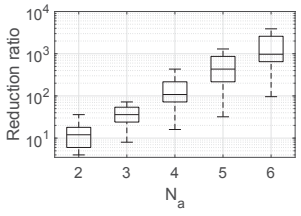

Figure 5: Ratio of 6^{N_a} with the number of selected combinations

10.5 Gesture Recognition Accuracy

Recall that we collected samples for 15 combinations, with N_a ranging from 2 to 6. To calculate the accuracy of WiMU, we classified each sample in each combination 50 times, each time generating virtual samples using a different set of randomly selected training samples. In generating virtual samples to classify any given sample, we ensured that the volunteer whose training samples WiMU used is not among the volunteers who provided the given sample.

Figure 6 plots the average accuracies of WiMU for each of the 15 combinations calculated using the samples where the volunteers started the gestures in parallel. Figure 7 plots the same using the samples containing sequentially started gestures. Each bar in these figures for any given combination of N_a gestures shows the percentage of classification rounds in which WiMU correctly identified all N_a gestures out of the 750 rounds of classification. As the volunteers randomly chose positions when providing samples and as the furniture was moved across days, the results in these figures take into account the changes in user's positions and environment.

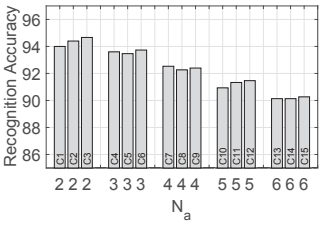

Figure 6: WiMU's average accuracy for parallel overlapping gestures

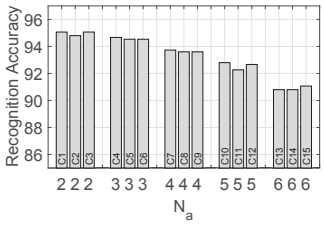

Figure 7: WiMU's average accuracy for sequential overlapping gestures

WiMU achieved an average accuracy of 95.0, 94.6, 93.6, 92.6, and 90.9% in recognizing 2, 3, 4, 5, and 6 simultaneously and sequentially performed gestures, respectively. The accuracy was about 0.81% lower when the gestures were started in parallel. The slight loss in accuracy for the parallel started gestures is due to the slightly lower percentage of their samples in which WiMU's gesture segmentation module correctly detected the number of gestures. While the average accuracy of WiMU decreases as N_a increases, the rate of decrease is very low, and the average accuracies that WiMU achieves even for $N_a = 6$ are >90%.

Figure 8 plots the percentage of classification rounds in which WiMU correctly identified at least k gestures, where $1 \leq k \leq N_a$. Each set of bars corresponding to each value of N_a is made using the classification results obtained after doing 50 rounds of classifications of all 90 samples in our data set that have N_a simultaneously performed gestures. Consider the set of bars corresponding to $N_a = 6$. We observe from this figure that while WiMU identifies all 6 gestures correctly 90.5% of the times, it identifies, for example, at least 3 gestures correctly 94.4% of the times. This shows that even in the cases where WiMU does not identify all gestures correctly, it still identifies some of them correctly.

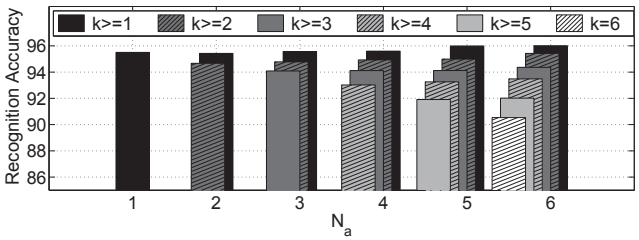

Figure 8: Average accuracies of WiMU in recognizing at least k gestures correctly, where $1 \leq k \leq N_a$

10.6 Comparison with Prior Art

Next, we study how WiMU performs on gestures performed individually and compare its accuracy with prior art. To measure the accuracy of WiMU on individually performed gestures, we calculated 10-fold cross validation accuracy of WiMU on the 600 individually performed samples. Figure 9 shows the confusion matrix for the six gestures. We observe from this figure that WiMU classified gestures with a high accuracy of 95.5% on our data set. This accuracy achieved by WiMU on individually performed gestures is at par with prior WiFi based systems. Some prominent WiFi based gesture and activity recognition systems, namely E-eyes [34], CARM [32], WiGest [6], and WiAG [30] reported average accuracies of 96%, 96%, 96%, and 92%, respectively.

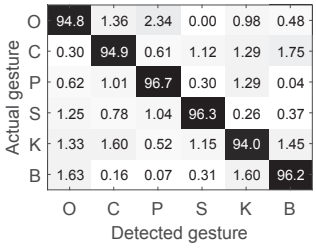

Figure 9: Confusion matrix for individually performed gestures

10.7 Impact of Height and Weight

The motivation behind studying the impact of user heights and weights on the accuracy of WiMU is to establish whether WiMU's accuracy is impacted by physiological properties of human body. The two subfigures in Figure 10 plot the accuracies of WiMU for users sorted with respect to their heights and weights, respectively. Each bar in the two figures for a user with any given height/weight shows the percentage of samples of that user correctly classified by WiMU during the 10-fold cross validation discussed in Section 10.6. We do not observe any apparent trends in WiMU's accuracy with changes in volunteers' heights and weights. This shows that the physiology of the user does not have any significant impact on the accuracy of WiMU.

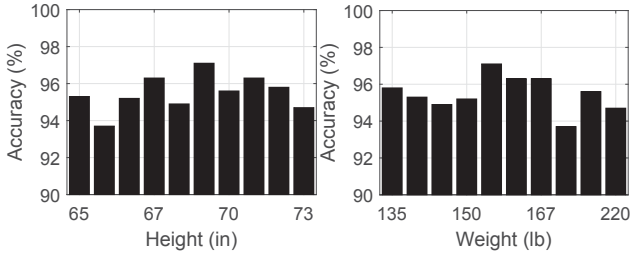

Figure 10: Impact of users' heights and weights on WiMU's accuracy

10.8 Impact of Distance Between Users

Next, we study how far apart users have to be for WiMU to recognize their simultaneously performed gestures. We collected more samples from our volunteers using $N_a = 3$, *i.e.*, three users performed gestures simultaneously. We asked the volunteers to perform combinations C4, C5 and C6 while standing at a distances of 1.5, 2, 2.5, 3, and 4 feet from each other. At each distance and for each configuration, we collected 15 samples. Figure 11 plots the accuracy in each of the three configurations at the five distance values averaged over 100 runs of evaluations, where in each run, we used a different set of virtual samples as the training samples. We observe from this figure that when the volunteers are very close to each other, *i.e.*, just 18 inches apart, WiMU's accuracy is relatively low, and improves rapidly as the distance increases. The relatively low accuracy at smaller separations can be attributed to volunteers 1) blocking each others' signal reflections, and 2) finding it difficult to properly perform gestures due to close proximity.

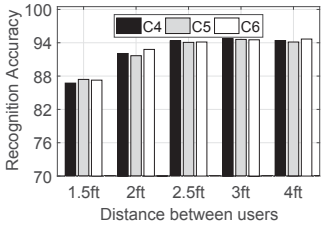

Figure 11: WiMU's average accuracy for different distances between users

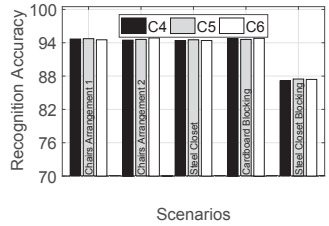

Figure 12: WiMU's average accuracy in different scenarios

10.9 Impact of Environmental Changes

In this section, we evaluate the performance of WiMU when the number and position of stationary objects in an environment change across samples. For this, we collected yet another set of samples from our volunteers, again keeping $N_a = 3$ and asking the volunteers to perform combinations C4, C5, and C6 in five different scenarios while standing at a distance of 3ft from each other. In the first scenario, we collected 15 samples for each of these three combinations using the default setup of our lab, *i.e.*, containing 7 chairs and 4 tables, neatly placed at their usual locations. In the second scenario, we again collected 15 samples for each of the three combinations, this time randomly scattering the 7 chairs throughout the lab. In the third scenario, we introduce a 6ft by 3ft steel closet in the room to add strong reflected components to the signal arriving at the receiver. Figure 13 plots the difference between the CFR power of each subcarrier in scenario 3 and the CFR power of the corresponding subcarriers in scenario 1. We observe from this figure that adding the steel closet changes the CFR power of subcarriers by up to 25dBm. This happens due to the strong reflected signal component that the steel closet adds, which constructively or destructively interferes with the existing signal components. In the fourth scenario, we placed a 6ft by 6ft room divider made of cardboard in front of one of the volunteers, such that it blocked the line of sight path between the volunteer and the receiver. In the fifth scenario, we placed the steel closet in front of one of the volunteers, such that it blocked the line of sight path between the volunteer and the receiver.

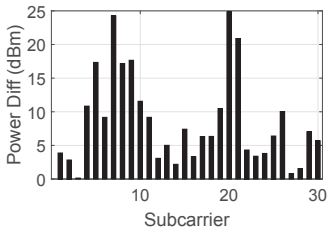

Figure 13: Difference in CFR powers of subcarriers across scenarios 1 & 3

Figure 12 plots the average accuracy for each of the three configurations in the five scenarios. We observe from this figure that WiMU achieves almost similar accuracy in the first four scenarios. However, in the fifth scenario, the average accuracy drops by about 6% due to the loss of the line of sight component. We also observe that in scenario four, where a room divider was used to block the line of sight component, the impact on accuracy is negligible because WiFi signals propagate well through cardboard.

10.10 Processing Latency

WiMU took an average of 2 minutes and 23.4 seconds to process each training sample using the method described in Section 5. Recall that this relatively large processing time is not problematic because it is a one time cost incurred at the time of setting up WiMU, and does not impact WiMU's runtime performance. WiMU took an average of 1.1, 1.6, 2.1, 2.5, 2.8, and 3.1 seconds to complete a classification round to recognize all gestures in a sample with 1, 2, 3, 4, 5, and 6 simultaneously performed gestures, respectively. Each classification round involved gesture segmentation, gesture combination selection, virtual sample generation, and gesture recognition, as described in Sections 6, 7, 8, and 9, respectively. We believe that by leveraging the GPU based parallelization that is now available on household computers, we can bring down WiMU's time to complete each classification round to under a second. Current voice based assistants experience a latency upwards of 1.5 seconds [2].

11 LIMITATIONS

While this paper is the first work that formally addresses the challenging problem of multi-user gesture recognition using WiFi, it is by no means the last, as our work has limitations that still need to be addressed before WiFi based multi-user gesture recognition systems can be used in practice. Next, we describe these limitations.

Background Movements: Currently, WiMU assumes that all gestures in any set of simultaneously performed gestures are predefined. Recall that a predefined gesture is a gesture for which WiMU has training samples performed by a real user. In case one or more users perform gestures or activities that are unknown to WiMU, such as household chores, while one or more other users perform predefined gestures, WiMU cannot recognize the predefined gestures due to the interference from the users performing the background gestures. The authors of WiSee [25] made a preliminary effort towards eliminating the impact of such interfering users. In future, we plan to extend the preliminary approach presented in [25] and augment WiMU with it to enable multi-user gesture recognition in the presence of background movements.

Machine Learning Methods: The simple Jaccard similarity coefficient based method that we used in this paper to compare test samples with virtual samples is relatively a less sophisticated machine learning method used for classification. A significantly more sophisticated machine learning based classification technique, such as support vector classification, would result in a significantly higher accuracy and robustness. Our use of the simpler technique in this paper was motivated by two factors. First, we primarily wanted to study the performance of our proposed approach of generating virtual samples. Had we used a more sophisticated machine learning method, it would have been difficult to determine who to attribute the good observed accuracy to: the virtual sample generation technique or the machine learning method. Second, the sophisticated machine learning techniques are computationally more expensive, and would lead to larger processing latency compared to that reported in Section 10.10. Now that we have demonstrated that WiMU achieves good accuracy with the simpler machine learning methods, albeit in a relatively controlled environment, in future, we plan to do a parallelized cloud based implementation of WiMU with one of the more sophisticated machine learning methods to further improve its accuracy and robustness.

Sensitivity to User Orientation: If a user performs a gesture at runtime in an orientation that is significantly different from any of the orientations in which the training samples were collected, WiMU's accuracy will suffer. To address this challenge, we plan to leverage our recent work, reported in [30], that specifically addresses the problem of recognizing gestures in arbitrary orientations, albeit for a single user. We will augment WiMU with the approach in [30] to make it agnostic to orientations of the users.

Gesture Association with Users: While WiMU recognizes simultaneously performed gestures, and if some of the users perform the same gesture, even tells how many users performed that gesture, it cannot yet associate gestures with users. Enabling WiMU to do so is particularly challenging because it requires user localization and user identification, which are not only challenging tasks but also complete projects in themselves. We hope that this work will spark interest of research community, and in near future, we will see work that can not only recognize simultaneously performed gestures but can also associate them to individual users.

Similar Gestures: While a human observer can distinguish between a yawn and a gesture where a user lifts both arms up, it is difficult for WiMU to distinguish between them. In general, it is difficult for any WiFi based system to distinguish between gestures that give rise to similar movements of body parts. Consequently, just like any other WiFi based gesture recognition systems, WiMU also requires the gestures to be sufficiently different from each other in terms of the movements of body parts.

Contiguous Gestures of Individual Users: While WiMU can recognize the gestures performed by multiple users simultaneously, for its gesture segmentation module, it requires that each user among the multiple users take a brief pause before and after performing his/her gesture. If any given user performs multiple predefined gestures contiguously, WiMU's gesture segmentation module cannot segment his/her gestures. Segmenting gestures from a set of contiguously performed gestures by any given user is also a challenging problem. To address this problem, in our future work, we plan to explore and leverage the hidden Markov chains based solutions that the speech processing community has developed to segment words in spoken sentences.

12 CONCLUSION

In this paper, we proposed WiMU, a theoretically grounded WiFi based system that recognizes simultaneously performed gestures. The key novelty of WiMU lies in the idea of generating virtual samples from individually performed gestures such that the virtual samples are very similar in properties to the real samples obtained from multiple users performing gestures simultaneously. The key technical depth of WiMU lies in the theoretically grounded designs of its modules that leverage insights from Eq. (1). We implemented WiMU using commodity WiFi devices and demonstrated that it recognizes 2, 3, 4, 5, and 6 simultaneously performed gestures with average accuracies of 95.0, 94.6, 93.6, 92.6, and 90.9%, respectively.

ACKNOWLEDGMENTS

This work is supported in part by the National Science Foundation, under grant # CNS 1565609. The authors would also like to thank the anonymous reviewers for their valuable feedback and suggestions.

REFERENCES

[1] https://www.ettus.com/product/category/USRP-Networked-Series.

[2] Amazon Echo. http://www.businessinsider.com/the-inside-story-of-how-amazon-created-echo-2016-4.

[3] Independent Component Analysis. https://en.wikipedia.org/wiki/Independent_component_analysis.

[4] Jaccard Index. https://en.wikipedia.org/wiki/Jaccard_index.

[5] USRP N210. https://www.ettus.com/product/details/UN210-KIT.

[6] Heba Abdelnasser, Moustafa Youssef, and Khaled A Harras. 2015. WiGest: A ubiquitous WiFi-based gesture recognition system. In *Proceedings of IEEE INFOCOM*.

[7] Fadel Adib, Zachary Kabelac, and Dina Katabi. 2015. Multi-person Localization via RF Body Reflections. In *Proceedings of the 12th USENIX Conference on Networked Systems Design and Implementation (NSDI'15)*. USENIX Association, Berkeley, CA, USA, 279–292. http://portal.acm.org/citation.cfm?id=2789790

[8] Fadel Adib and Dina Katabi. 2013. See Through Walls with WiFi!. In *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM (SIGCOMM '13)*. ACM, New York, NY, USA, 75–86. DOI: https://doi.org/10.1145/2486001.2486039

[9] Fadel Adib, Hongzi Mao, Zachary Kabelac, Dina Katabi, and Robert C Miller. 2015. Smart homes that monitor breathing and heart rate. In *Proceedings of the 33rd annual ACM conference on human factors in computing systems*. ACM, 837–846.

[10] J-F Cardoso. 1998. Blind signal separation: statistical principles. *Proc. IEEE* 86, 10 (1998), 2009–2025.

[11] Lisa Y Chen, Benjamin C-K Tee, Alex L Chortos, Gregor Schwartz, Victor Tse, Darren J Lipomi, H-S Philip Wong, Michael V McConnell, and Zhenan Bao. 2014. Continuous wireless pressure monitoring and mapping with ultra-small passive sensors for health monitoring and critical care. *Nature communications* 5 (2014), 5028.

[12] Christopher Dondzila and Dena Garner. 2016. Comparative accuracy of fitness tracking modalities in quantifying energy expenditure. *Journal of medical engineering & technology* 40, 6 (2016), 325–329.

[13] Biyi Fang, Nicholas D Lane, Mi Zhang, Aidan Boran, and Fahim Kawsar. 2016. BodyScan: Enabling radio-based sensing on wearable devices for contactless activity and vital sign monitoring. In *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, 97–110.

[14] Biyi Fang, Nicholas D Lane, Mi Zhang, and Fahim Kawsar. 2016. Headscan: A wearable system for radio-based sensing of head and mouth-related activities. In *Proceedings of the 15th International Conference on Information Processing in Sensor Networks*. IEEE Press, 21.

[15] Daniel Halperin, Wenjun Hu, Anmol Sheth, and David Wetherall. 2011. Tool Release: Gathering 802.11n Traces with Channel State Information. *ACM SIGCOMM CCR* 41, 1 (Jan. 2011), 53.

[16] Lieve Hamers, Yves Hemeryck, Guido Herweyers, Marc Janssen, Hans Keters, Ronald Rousseau, and André Vanhoutte. 1989. Similarity measures in scientometric research: the Jaccard index versus Salton's cosine formula. *Info. Processing & Management* 25, 3 (1989), 315–318.

[17] Chunmei Han, Kaishun Wu, Yuxi Wang, and Lionel M Ni. 2014. WiFall: Device-free fall detection by wireless networks. In *Proceedings of IEEE INFOCOM*. 271–279.

[18] Aapo Hyvärinen and Erkki Oja. 2000. Independent component analysis: algorithms and applications. *Neural networks* 13, 4 (2000), 411–430.

[19] Kanitthika Kaewkannate and Soochan Kim. 2016. A comparison of wearable fitness devices. *BMC public health* 16, 1 (2016), 433.

[20] Bryce Kellogg, Vamsi Talla, and Shyamnath Gollakota. 2014. Bringing Gesture Recognition to All Devices. In *Proceedings of the 11th USENIX Conference on Networked Systems Design and Implementation (NSDI'14)*. USENIX Association, Berkeley, CA, USA, 303–316. http://dl.acm.org/citation.cfm?id=2616448.2616477

[21] Taesu Kim, Hagai Thomas Attias, Soo-Young Lee, and Te-Won Lee. 2007. Blind source separation exploiting higher-order frequency dependencies. *IEEE Trans. on Audio, Speech, and Language Processing* 15, 1 (2007), 70–79.

[22] Dana E King, Eric Matheson, Svetlana Chirina, Anoop Shankar, and Jordan Broman-Fulks. 2013. The status of baby boomers' health in the United States: the healthiest generation? *JAMA internal medicine* 173, 5 (2013), 385–386.

[23] Changzhi Li, Victor M Lubecke, Olga Boric-Lubecke, and Jenshan Lin. 2013. A review on recent advances in Doppler radar sensors for noncontact healthcare monitoring. *IEEE Transactions on microwave theory and techniques* 61, 5 (2013), 2046–2060.

[24] Rajalakshmi Nandakumar, Shyamnath Gollakota, and Nathaniel Watson. 2015. Contactless sleep apnea detection on smartphones. In *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, 45–57.

[25] Qifan Pu, Sidhant Gupta, Shyamnath Gollakota, and Shwetak Patel. 2013. Whole-home Gesture Recognition Using Wireless Signals. In *Proceedings of the 19th Annual International Conference on Mobile Computing & Networking (MobiCom '13)*. ACM, New York, NY, USA, 27–38. DOI: https://doi.org/10.1145/2500423.2500436

[26] Souvik Sen, Božidar Radunovic, Romit Roy Choudhury, and Tom Minka. 2012. You are facing the mona lisa: spot localization using phy layer information. In *Proceedings of ACM MobiSys*. 183–196.

[27] Li Sun, Souvik Sen, Dimitrios Koutsonikolas, and Kyu-Han Kim. 2015. Widraw: Enabling hands-free drawing in the air on commodity wifi devices. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*. ACM, 77–89.

[28] Sheng Tan and Jie Yang. 2016. WiFinger: leveraging commodity WiFi for fine-grained finger gesture recognition. In *Proceedings of the 17th ACM International Symposium on Mobile Ad Hoc Networking and Computing*. ACM, 201–210.

[29] David Tse and Pramod Viswanath. 2005. *Fundamentals of wireless communication*. Cambridge university press.

[30] Aditya Virmani and Muhammad Shahzad. 2017. Position and Orientation Agnostic Gesture Recognition Using WiFi. In *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, 252–264.

[31] Guanhua Wang, Yongpan Zou, Zimu Zhou, Kaishun Wu, and Lionel M. Ni. 2014. We Can Hear You with Wi-Fi!. In *Proceedings of the 20th Annual International Conference on Mobile Computing and Networking (MobiCom '14)*. ACM, New York, NY, USA, 593–604. DOI: https://doi.org/10.1145/2639108.2639112

[32] Wei Wang, Alex X Liu, Muhammad Shahzad, Kang Ling, and Sanglu Lu. 2015. Understanding and Modeling of WiFi Signal Based Human Activity Recognition. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*. ACM, 65–76.

[33] Wei Wang, Alex X Liu, Muhammad Shahzad, Kang Ling, and Sanglu Lu. 2017. Device-Free Human Activity Recognition Using Commercial WiFi Devices. *IEEE Journal on Selected Areas in Communications* 35, 5 (2017), 1118–1131.

[34] Yan Wang, Jian Liu, Yingying Chen, Marco Gruteser, Jie Yang, and Hongbo Liu. 2014. E-eyes: In-home Device-free Activity Identification Using Fine-grained WiFi Signatures. In *Proceedings of ACM MobiCom*.

[35] Susan Krauss Whitbourne and Sherry L Willis. 2014. *The baby boomers grow up: Contemporary perspectives on midlife*. Psychology Press.

[36] Joey Wilson and Neal Patwari. 2011. See-through walls: Motion tracking using variance-based radio tomography networks. *IEEE Transactions on Mobile Computing* 10, 5 (2011), 612–621.

[37] Wei Xi, Jizhong Zhao, Xiang-Yang Li, Kun Zhao, Shaojie Tang, Xue Liu, and Zhiping Jiang. 2014. Electronic Frog Eye: Counting Crowd Using WiFi. In *Proceedings of IEEE INFOCOM*.