WiMic: Recovering sound with wireless signals

Aviv Adler
adlera@mit.edu

David Alvarez-Melis
dalvmel@mit.edu

Sayeed Tasnim
sayeedt@mit.edu

ABSTRACT

We propose and develop *WiMic*, a sound capturing system that can recover simple sounds using only reflected wireless signals. Our method uses WiTrack to measure small scale vibrations caused by sound emitting devices, and applies several layers of processing to recover the sound contained in these noisy signals. Besides the WiTrack apparatus, WiMic requires no further hardware. In particular, it does not use microphones of any type, it requires no knowledge of the sound emitting device, and it works even at a distance of several meters from the source of the sound. Our method works also when multiple sources are simultaneously emitting different sounds, and it is able to recover these signals independently as long as the sources are not too close in terms of radial distance from the transmitter. We demonstrate the effectiveness of our method in several experimental setups.

1. INTRODUCTION

1.1 Problem Statement

The purpose of this work is to apply recent advances in indoor localization systems in a new context: extracting audio information from wireless signals. Inspired by the applications of WiTrack [1] in detecting breathing and heart beat patterns as well as VisualMicrophone's [3] ability to reconstruct audio signals by capturing minute vibrations of objects in video, we aim to extract similar information for audio signals using the WiFi signals. By using the subtle vibrations of audio on receptive materials, the audio signals may become quantifiable at the receiver antenna. Similarly, vibrations in the throat and mouth areas may provide reliable information to reconstruct speech signals and perform recognition on them.

Since WiFi signals are ubiquitous in the environment, they could potentially allow for sound detection and speech recognition without any additional hardware. In addition, users would not need to carry any microphone or other devices. A potential application of this approach could be for recording audio in meetings. One might forget to bring a microphone and opt for the WiFi receiver to record and log the audio data for the meeting. The hearing impaired could also benefit from a speech recognition system that does not require other people to carry any devices and which could alert them of important sounds in the environment, such as the doorbell or telephone.

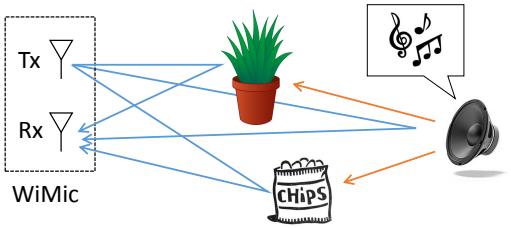


Figure 1: A basic illustration of WiMic's concept. Something (possibly a speaker or a person) makes a sound, and the sound waves (in orange) cause nearby objects to vibrate. The WiMic apparatus (the same hardware as the WiTrack system) reads these vibrations by reflecting WiFi signals (in blue) off of them, and possibly off the source of the sound as well, and reconstructs the sound.

One big advantage of this system over other standard microphones is that it has the ability to isolate and separate two different coinciding sounds, as well as to record and play back sounds which are masked by louder sounds. This is because instead of receiving the combined sound at one location (the way a microphone does), it looks at the effects of the sounds at many different places; if there are two sound sources, it will see different sounds being prevalent at different areas. For example, to record conversations at an event with many independent discussions, this method is ideal as it can locate the sources of different sounds and (possibly even plotting them on a map of the room) and play them back independently.

Another advantage of WiMic is that, in theory, it should be able to "hear" through walls (in the same manner that WiTrack can see through walls), which a microphone might not necessarily be able to do.

A final advantage of this method (in theory) is that, by collecting many different signals corresponding to the same sound, it might be possible to reconstruct the original sound with higher resolution than the sampling rate of the system; a similar technique was used by the Visiual Microphone group to reconstruct sounds with resolution higher than the framerate of the video taken.

1.2 Previous Work

There has been a great deal of past work on detecting people and interpreting their motions and gestures using Wifi

signals, notably WiSee and WiTrack [1, 2, 5]. In particular, WiTrack has been shown to have a resolution fine enough to detect the heartbeat of a person, so it seems natural that it should be possible to detect the vibration of objects caused by sound waves passing through them. Similar techniques date back even to the 1940's using infrared light (the Soviet *Buran* eavesdropping system, invented by Leon Theremin), and more recently with lasers ("laser microphones").

More recent work along these lines, by the Visual Microphone group at MIT, has shown that sound can be recovered from videos of objects (such as a houseplant, a bag of chips, or a bottle of water)[3]. They managed to accurately reproduce a MIDI tune and human speech, despite the fact that sometimes the frequency of the sound exceeds the frame rate of the video (they achieve this by exploiting image artifacts caused by the rolling shutter in most commercial video cameras).

There has even been recent work on recovering speech using Wifi in the WiHear project at the Hong Kong University of Science and Technology [6], though they use a markedly different method. The WiHear method relies on recognizing the different poses of the subject's mouth, which then allows deduction of the phoneme being produced (so they are recovering speech only, rather than all sound). This technique has advantages and disadvantages due to the fact that it only recovers the speech of the subject; in particular, it cannot work on non-speech sounds or even speech coming from a loudspeaker. Another serious drawback is that WiHear needs to learn the mouth poses for each person it tries to transcribe, whereas the direct sound recovery technique we propose doesn't require this step.

1.3 Summary of the System and Results

Our method uses WiTrack's hardware apparatus and uses WiTrack as a subroutine for detecting the vibrations of objects. It then measures the sounds from these vibrations and automatically detects significant signals (by finding measurements containing frequencies far above what could be expected from measurement noise) and isolates the various source sounds in the environment (by applying a clustering algorithm to identify which signals correspond to the same sound, and which to different sounds). It proceeds in the following steps:

1. **Measurement:** Use WiTrack to measure distances to various objects in the environment; the vibrations of objects can then be inferred from these measurements. Specifically, WiTrack returns measurements in various "bins" corresponding (roughly) to different distances from the apparatus.

2. **Preprocessing:** Process the signal from each bin by de-trending the signal, applying a bandpass filter to remove frequencies which wouldn't correlate to sound (mostly very low frequencies) and removing noise.

3. **Signal Detection:** Detects the bins corresponding to a sound (rather than just noisy measurements) by identifying which bins contain frequencies well above their noise levels.

4. **Isolating Sounds:** Determines which bins correspond to different sounds in the environment by clustering the signals of the 'significant' bins.

5. **Bin Alignment and Aggregation:** For each cluster (which represents a sound present in the environment), aligns all the bins in that cluster and adds them up to create a reconstruction of the original sound.

Finally, the system outputs the various sounds which were detected and reconstructed.

In experiments, we were able to reliably reconstruct sounds using this system, and we were even able to reliably isolate and identify sounds from different sources. Specifically, we were able to reconstruct sounds played from a speaker, such as a multi-tone music clip and frequency sweeps (in which the sound is continuous but constantly increasing the pitch), by measuring the vibration of the speaker using the WiTrack method. We were also able to reliably separate sounds being played by two different speakers at the same time. Thus, our experiments serve as a proof-of-concept that sound can be recognized from the vibrations of objects detected with the WiTrack method.

However, we were unable to produce sounds by measuring the vibration of objects which were not the source. While it should be theoretically possible to reconstruct sounds from the vibrations of objects in the environment (as proved by the Visual Microphone group), so far we have only been able to reproduce sounds by measuring the vibration of the source (in this case, a speaker) directly. Another serious limitation is that our experiments had only very minor success with recording human speech (as tested with excerpts from John F. Kennedy's inauguration speech and an excerpt of Morgan Freeman in *The Shawshank Redemption*). While we were able to identify a bin with Morgan Freeman's voice present, it was too faint against the noise to discern what he was saying, and (probably due to this faintness) the bin had to be discovered by searching for it manually, rather than with our automatic system.

2. DATA COLLECTION

We performed two sets of experiments, the first to see what materials could reliably reproduce sounds, and the second to see if our system could not only reproduce sounds from vibrations, but automatically detect significant bins (throwing away signals from bins which don't contain the sound) and cluster them according to which sound in the environment they correspond to.

2.1 Experimental Design

In both sets of experiments we performed, the basic physical set-up is the same. We place the WiTrack apparatus a few feet away from an object (either a sound source or a static object) from which we hope to reproduce the sound. We used speakers as the source of the sound in all experiments. In experiments where the object of interest was not the source of the sound, the source was placed off to the side, out of the angle of vision of the WiTrack apparatus.

Experiment set 1

The first phase of the project explored the viability of detecting sound frequencies using WiTrack. Initially, we began sound detection directly from the reflections of the FMCW signal off the membranes of a speaker and also from the reflections off other objects susceptible sound vibrations. We explored the results of WiTrack to see if the frequencies of the sound tones we played were present among the different bins of WiTrack.

The experiments were performed in common lab areas in CSAIL. A wooden tall chair was placed approximately 2 feet in front of the WiTrack transmitter and receiver antennas. We placed the object of interest on the wooden chair directly facing the antennas. If the object was the speaker membrane, it would directly face the antenna. However, if the object were another material such as a chips bag or a foam cup, the object would be placed on the chair and the speaker would be placed in front of the object as well at an approximate 45° counterclockwise angle facing downwards from the WiTrack transmitter-receiver antenna pair.

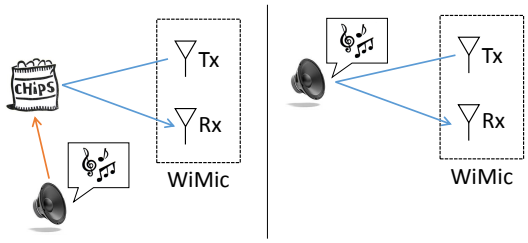


Figure 2: WiMic two-source experimental setup. *Left*: Measuring the vibration of an object in the presence of sound. *Right*: Measuring the vibration of the sound's source directly.

The experiments from the first set that we performed are shown in Table 3. During the first day of testing, we used a speaker with an exposed membrane so the sounds would visibly vibrate the speakers. This is the source for the direct membrane experiments. A Lays plastic potato chip bag (with some wrinkles) laid flat against the back of the chair was used for the chips bag experiment. A styrofoam cup placed vertically and normally was used for the foam cup experiments. A balloon about 5 inches in diameter was used for the balloon experiment. During the second day of testing, a different long speaker was used with a covered membrane. In order to amplify the low frequency tones, we used an improvised subwoofer. Two foam cups were attached to the sides of the speaker to amplify low frequencies for WiTrack detection.

The sounds used for the first set of experiments are as follows; for compactness in Table 3, we refer to each sound with a particular label, as given below.

Most of the sounds were monotone pure sinusoids generated online. In some instances, we played a frequency sweep that would increase linearly in frequency over some range and duration. We label this as `1-tone(`X`)` where X is the frequency of the sound played (in Hz). We also used a sequential three-tone sound with frequencies of 50, 100, and 200 Hz. These tones were played consecutively in succession for approximately 3 seconds each with about a 1 second pause in between tones. We label this `3-tone`. We also used frequency sweeps, in which a tone smoothly changes from X_1 to X_2, and which we label `sweep(`X_1`, `X_2`)` (if $X_1 < X_2$, it's a rising pitch, and otherwise a falling pitch). Finally, we created a sinusoidal signal carrying the song *Mary had a little lamb*, using MATLAB. We generated it in key of G, shifting an octave down from the natural octave (ie. with A-440Hz). With this, we wanted to ensure the sound would be in a low frequency interval, easy for our method to detect. We label this tune `mhll`.

Our experiment with the three-tone sound and the subwoofer was the one on which we did the most extensive analysis. In particular, this experiment allowed us to see in which bins the frequencies are presenting themselves. Interestingly, some bins may exhibit one tone strongly, but the other two only weakly or not at all; we hypothesize that this is due to some materials naturally resonating with different frequencies. This suggests that for future applications the ability to collect and aggregate bins may be very important in order to reconstruct the full sound (rather than just some subset of its frequencies) since there may not be a single bin which correctly represents the entire sound.

Source	Sound	Duration
Direct Membrane	`1-tone(20)`	3 s
Direct Membrane	`1-tone(80)`	3 s
Direct Membrane	`1-tone(83)`	3 s
Direct Membrane	`1-tone(300)`	3 s
Direct Membrane	`sweep(1,400)`	4 s
Direct Membrane	`mhll`	6 s
Chips Bag	`1-tone(20)`	3 s
Chips Bag	`1-tone(100)`	3 s
Chips Bag	`sweep(100,300)`	4 s
Chips Bag	`sweep(300,100)`	4 s
Foam Cup	`1-tone(80)`	3 s
Foam Cup	`1-tone(100)`	3 s
Balloon	`1-tone(100)`	3 s
Subwoofer	`1-tone(100)`	3 s
Subwoofer	`1-tone(200)`	3 s
Subwoofer	`3-tone`	11 s
Subwoofer	`sweep(50,200)`	4 s

Figure 3: WiMic Preliminary Audio Tone Experiments

Figure 4: WiMic chips bag experimental setup

Experiment set 2

The second set of experiments was primarily aimed at testing if our method could reliably distinguish sounds from different sources, and thus used two speakers rather than one. We additionally ran some experiments to test whether we could reproduce more complex sounds like human speech and also to test what frequencies we could reproduce. Due to our results from Experiment set 1, we tested only direct measurement of the speakers and the subwoofer.

We tested the following sounds in the second experiment set:

- `3-tone` and `mhll`, as in the first set
- a permutation of `3-tone` (100, 200, 50 Hz): `3-tone-v2`
- `sweep(1,500)` (the large range is to allow us to see at what frequency WiMic starts to lose the ability to detect the sound)
- an extended series of tones played sequentially and rising at each step ranging from 100 Hz to 500 Hz: `step` (this serves the same purpose as `sweep(1,500)`)
- an excerpt of John F. Kennedy's inauguration speech ("Ask not what your country can do for you..."): `jfk`
- an excerpt of Morgan Freeman speaking, taken from *The Shawshank Redemption*: `freeman`

In our list of experiments, we refer to these sounds by the labels given here. We chose to use John F. Kennedy's inauguration and Morgan Freeman in *The Shawshank Redemption* in order to capture a range of frequencies in the human voice.

We played these sounds from two speakers: Speaker 1 was placed on a chair approximately 4 feet from the WiTrack apparatus, and Speaker 2 was placed approximately 9 feet away from the WiTrack apparatus (see Figure 5). Speaker 1 was the speaker with non-visible membrane, with the improvised subwoofer, and speaker 2 was the speaker with the visible membrane (both are the same as the ones in Experiment set 1). Our experiments are listed in Table 6.

We remark that we don't list the durations in Table 6 because it is not relevant for this set of experiments.

2.2 WiTrack Output

The WiTrack system provided both magnitude and phase plots of the response coefficient. In our experiments, the phase plots showed little to no information about the generated audio signals. The magnitude plots provided the best demonstration of these frequencies. The main theory idea behind this behavior is in the relation $h \propto \frac{1}{d} e^{j\frac{2\pi}{\lambda}d}$. The small signal fluctuation in the magnitude term, $\frac{1}{d}$, appears to have a more pronounced behavior than the phase $\frac{2\pi}{\lambda}d$ term. An example plot of the magnitude is shown in Figure 7. Note that all three frequencies are present in bin 276, but other bins show only one or two of the three frequencies. Due to the magnitude plot providing the most information, our work focuses on processing the corresponding data in the magnitude plots.

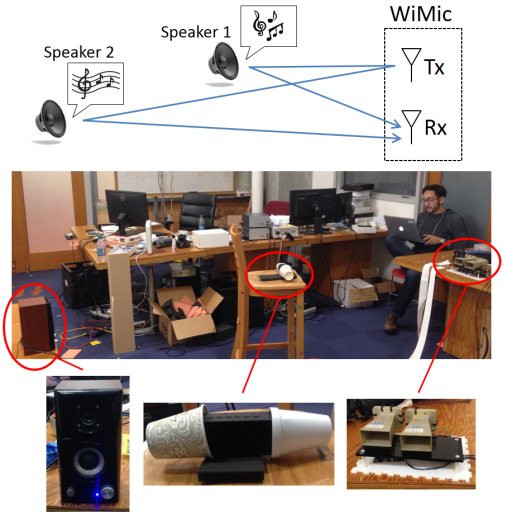


Figure 5: WiMic two-source experimental setup. *Top*: a basic diagram of the setup. *Bottom*: a photo of the experiments in progress with the speakers and WiTrack apparatus highlighted. Speaker 1 is shown with the 'subwoofer' (made out of cups) described in the Experimental set 1.

Speaker 1	Speaker 2
`3-tone`	`3-tone-v2`
`3-tone-v2`	`3-tone`
`3-tone`	`mhll`
`mhll`	n/a
`step`	n/a
`sweep(1, 500)`	n/a
`jfk`	n/a
n/a	`jfk`
n/a	`freeman`

Figure 6: WiMic two-source audio experiments. See Figure 5 to identify Speaker 1 and Speaker 2

We conjecture bin 276 to be the bin corresponding to the speaker itself due to all three frequencies being distinctly present. We also conjecture the responses from the other bins to be reflections of other nearby objects such as the wooden chair. Only some frequencies are present in these bins because of their varied resonance properties.

3. PREPROCESSING

Even though the sound signal might be visible in some bins, it is often hidden by the noise in the received signal. For the simple tones that we have experimented with so far, the signal is not completely discernible. The signal in Figure 8 exemplifies this phenomenon.

3.1 Trend removal

Our first approach to handle the bin-signals in WiTrack's output was to remove the mean over the whole duration of the input. This, however, is not very effective since many of the bin's have drifting signals, probably due to changes in the medium. This causes the mean to shift considerably

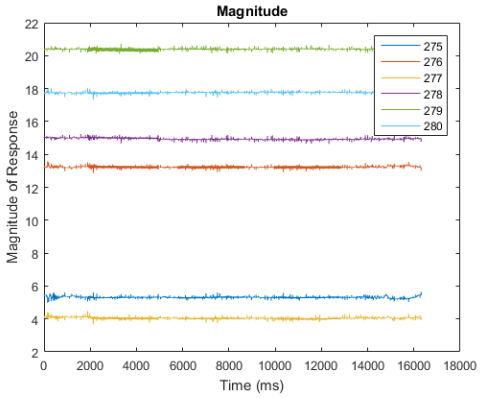


Figure 7: Magnitude of response plot of three tone signal from subwoofer

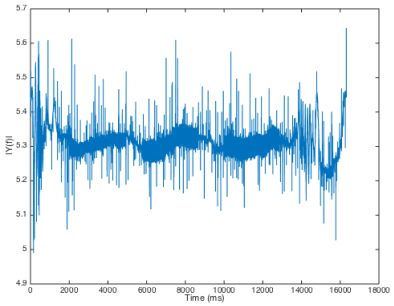


Figure 8: Raw signal from a single WiTrack bin

during the period considered. This can also be seen in 8, although the effect is much larger in other bins.

For this reason, we shifted to techniques to remove non-constant trends. We experimented with i) computing a running mean and subtracting it from the signal, ii) Median filtering, iii) Fitting a smoothed polynomial trend and removing it. We found these approaches to perform better in different situations, although most of the time the running mean yields better results that the other two.

3.2 Bandpass filtering

As a next step, we decided to try filtering out frequencies outside an "interval of interest", particularly low frequency interference. Since most of our experiments so far involved tones ranging from 20 to 400 Hz, we designed filters to blur out frequencies outside this range. In particular, removing very low frequencies is important since those might correspond to moving objects in the environment (as opposed to vibrations of the objects due to sound). We experimented with i) Digital Butterworth filtering of different orders, ii) Finite Impulse Response (FIR) filters, with different types of windowing (hamming, Chebyshev, etc). So far we have found FIR-hamming filters of order ~ 50 to work best in general, although again this seems to be depend on the particular type of sound analyzed.

3.3 Frequency spectrum cleaning

We then clean out "noise" frequencies so that the sound itself can be heard more clearly; we have several methods for doing this, with different advantages and disadvantages. However, all our methods are based on the idea of using an FFT to convert the time-domain signal to the frequency domain, and then identify frequencies with responses significantly above those of the others as "important"; "unimportant" frequencies are then cleaned out in some way.

We will illustrate these methods on the signal from bin 276 in the three-tone experiment; this signal is shown in Figure 9.

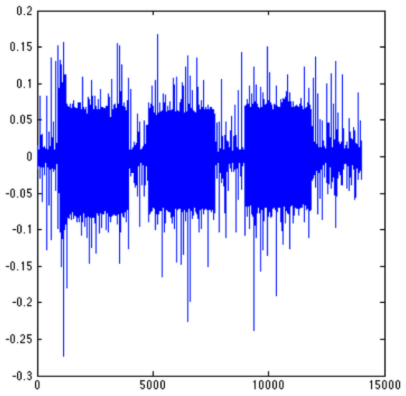


Figure 9: The detected signal from bin 276 in the time domain after de-trending and bandpass-filtering.

Formally, we use a threshold to separate the "important" frequencies from the "unimportant" ones. In order for this threshold to scale naturally with the magnitude of the frequency responses, we compute the mean frequency response and the standard deviation of the frequency responses. Then, all frequencies whose responses are more than a certain number of standard deviations above the mean are considered "important". We compute the number of standard deviations by using the inverse of CDF with the threshold, so that a natural intepretation of the threshold's value is "the fraction of the noise (if it were Gaussian) which would be cut out by this threshold"; we typically set a threshold in the range of $[0.90, 0.99]$. Thus, setting a higher threshold will remove more of the noise, but at the risk of accidentally removing some of the true frequencies. Figure 10 depicts the results of this method.

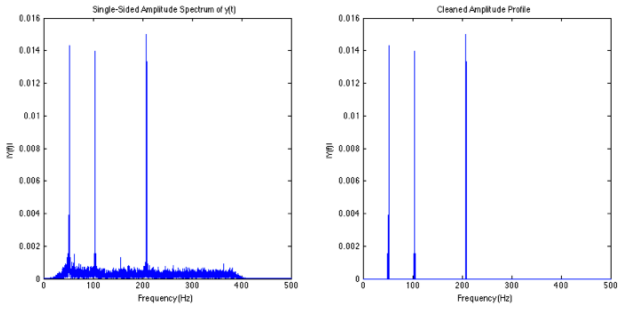


Figure 10: Left: the amplitudes of the frequency responses of the original signal from bin 276 (after detrending and bandpass-filtering). Right: the "important" frequencies only ($p = 0.98$)

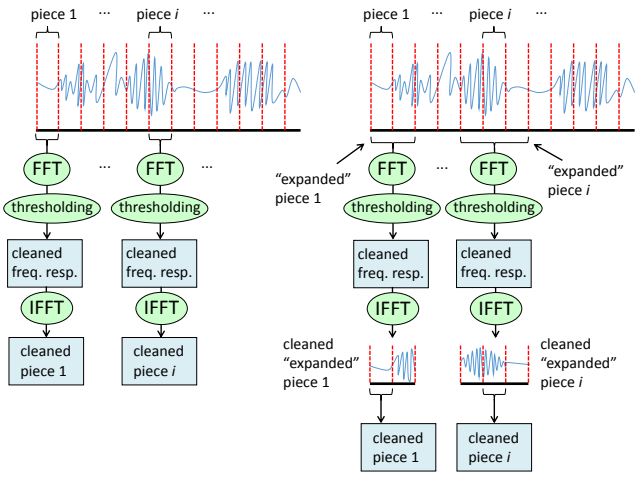


Figure 11: An illustration of noise-cleaning method (4), which we chose to use in our system, depicted on the right (in contrast to method (3), which is depicted on the left). In particular, this figure is meant to illustrate the creation of the "expanded" pieces and how the final signal is extracted from them.

We tested the following four methods for cleaning out noise:

1. Compute the spectrum over the whole signal with an FFT, zero out unimportant frequencies, and convert back to the time domain.

2. Compute the spectrum over the whole signal with an FFT, scale each frequency by a constant depending on how close it is to "important". In particular, we take a binary vector over the frequencies, with "1" indicating an important frequency and "0" indicating an unimportant one. We then convolve this with a pyramid vector (a vector v of length $2k+1$ – indexed from 0 to $2k$ – such that $v_j = j/k$ for $0 \leq j \leq k$, and $v_j = (2k-j)/j$ for $k < j \leq 2k$), cap the resulting vector at 1 (i.e. any entry with value > 1 is cut to 1) and use the resulting vector to scale the frequency responses.

3. Divide the time-domain signal into equal size pieces and run method (1) independently on each, concatenating the results into the cleaned signal.

4. Divide the time-doman signal into equal size pieces; then define the "expanded" piece corresponding to a given piece as it plus its neighbors, and run (1) independently on each "expanded" piece. Then, take each "expanded" clean signal and extract the (non-"expanded") piece we are considering. Finally, concatenate these pieces to form the final signal. See figure 11 for illustration.

All four methods vastly reduce the noise, but (1) produces a high-pitched ringing sound as an artifact, and in (3) the signal decays sharply at the transitions between different pieces, producing an artifact sound similar to the ringing of a telephone. (2) and (4) were in fact designed to correct these problems, respectively, and exhibit these problems to a much lesser degree (e.g. (2) produces a much more subdued ringing, and (4) produces much subtler transitions between different pieces). The resulting 'clean' signals from these four methods are shown in the time domain in Figure 12.

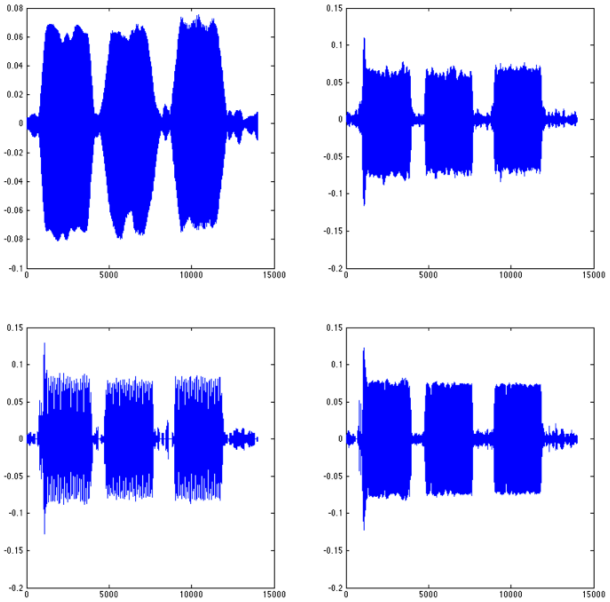


Figure 12: The time-domain signals of the sound after cleaning by the four methods. Threshold = 0.98, and interval size = 128 for methods (3) and (4). *Top*: method (1) on left, method (2) on right. *Bottom*: method (3) on left, method (4) on right

As a result of this analysis, we proceed with method (4) as our standard frequency-cleaning system, as it produces the fewest audible artifacts.

3.4 Complete processing pipeline

As detailed above, we have experimented with different methods to process and clean the data. Our current preprocessing for each bin consists of the following steps:

1. Removing the time-domain trend using one of: i) K-order running mean removal, ii) Median filtering or iii) Polynomial fit removal.

2. Bandpass filtering i) Digital Butterworth filter or ii) FIR filter using window method (hamming, chebyshev).

3. Frequency spectrum cleaning (as described in Section 3.3), using the expanded time intervals method illustrated in Figure 11.

An example of this processing pipeline is shown in Figure 13, where for each step we present time-domain, frequency-domain and spectrogram plots. Note that this process removes the harmonics seen in the spectrogram in the first three rows.

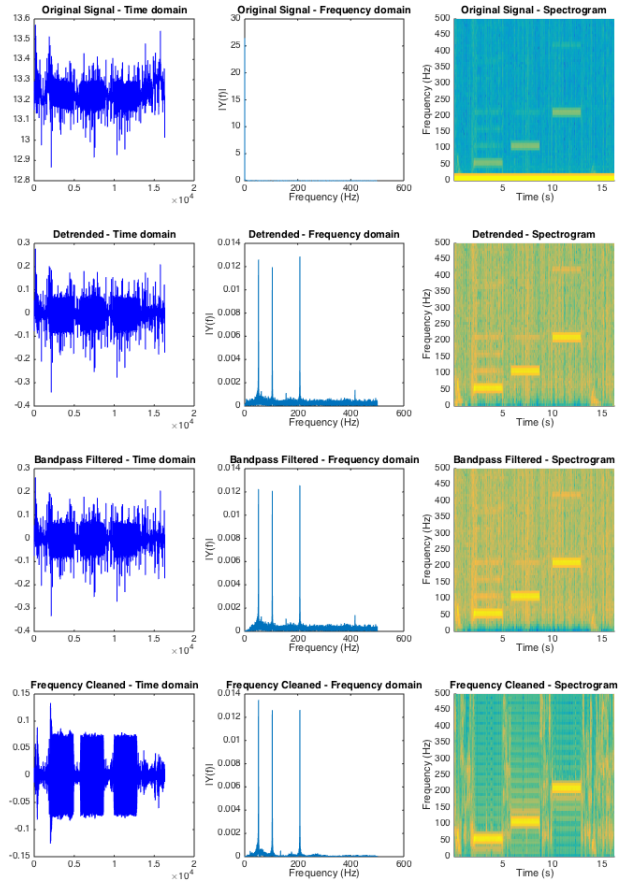


Figure 13: Step-by-step de-noising process for the three tone sound example. The top row shows the original signal, and each subsequent row adds another processing step on the previous one. For each step we show: time-domain plot, frequency-domain plot and spectrogram.

4. SOUND DETECTION

4.1 Spectrogram Denoising

Once the signals for the individual signals have been cleaned, we need to identify those that carry the sound of interest. Even though this can sometimes be done by visual inspection on the magnitude plots, it might not always be feasible to do so, particularly in the multiple-source scenario, where it might not be trivial to distinguish bins carrying sound from one source to the other.

The key idea behind our approach for detecting bins with sound is that sounds are captured in the WiTrack bins as periods of constant frequency for at least a few milliseconds. This, in turn, results in piece-wise constant spectrograms. Thus, we seek to identify bins with constant blocks of high value in their spectrogram. For this, we rely on *total variation denoising* (TVD), a smoothing signal processing technique that penalizes excessive variation in the data. In particular, we use a robust version of TVD tailored to noise removal from piecewise constant signals [4]. We use frequency bins of the spectrogram as inputs to this method, and retrieve signals that have been "flattened out" of excessive variation. An example of the output we get from TVD

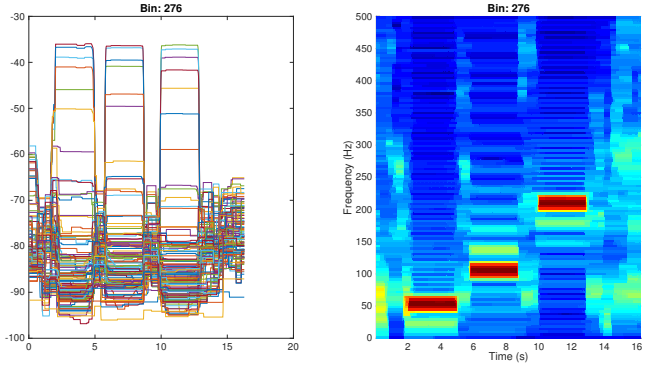


Figure 14: Single-bin TVD-denoised spectrogram. Left: denoised spectrogram frequency bins. Right: Reconstructed spectrogram.

is shown in Figure 14. After reconstruction, we obtain denoised spectrograms with sharp distinction between sound and non-sound regions.

4.2 Automatic Bin Selection

Once we have computed TVD-denoised spectrograms for all the bins, we use a heuristic selection based algorithm that we devised to identify "regions of interest" in the spectrograms. We flag as *potential sound sections* regions of the spectrogram that: i) have flat regions of more than k contiguous blocks and ii) have power above a certain threshold in those regions. We select all bins that satisfy these conditions in at least one region of their spectrogram, and rank them based on the number of regions satisfying these criteria. After this, we are left only with bins that most likely contain the sound of interest.

As a byproduct of the bin selection process, we get information not only about whether a bin carries sound or not, but also for *which frequencies* and *at what times* this happens. Thus, we can use this information to create data-driven bandpass filters that enhance the frequencies thought to be present in each signal.

4.3 Bin Clustering

The last step in our sound-detection pipeline involves grouping the selected bins into clusters of similar sounds. This is necessary to classify bins depending on which sound they are carrying in the multiple source scenario.

Most clustering algorithms take as an input a matrix of similarities (or distances) between datapoints. In our case, we compute the similarity of two bins as the inverse of the discrepancy between their TVD-denoised spectrograms. We quantify this discrepancy by computing the Frobenius norm between the matrix spectrograms, although other techniques could be used. Then we perform k-means clustering on these similarities, finally obtaining groupings of the signals. Continuing with the same two-source example, we show in Figure 15 the similarities between the 13 bins chosen during the automatic bin selection step. In this example, the bins 255-260 contains the first sound, and the bins 267-278 contain the second one. After feeding these similarities to the clustering algorithm, it clusters these 13 bins correctly.

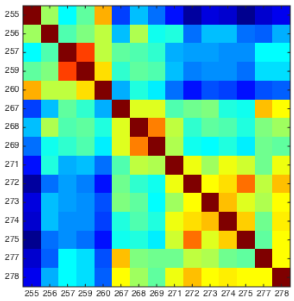
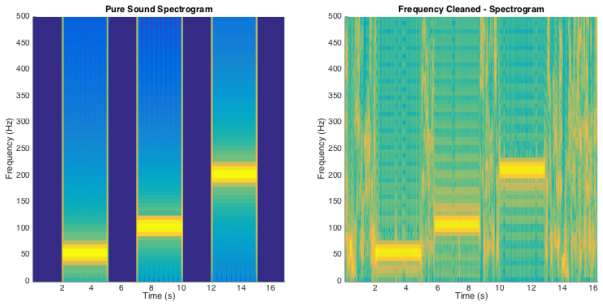


Figure 15: Typical heat map of bin similarities for the two source scenario. Blue colors indicate lower similarity. Bins 255-260 contain the the first sound, while 267-278 contain the second one.



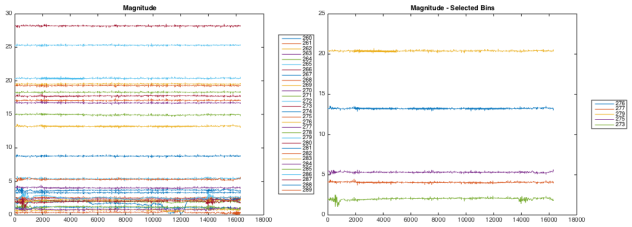
(a) Original. (b) Reconstructed.

Figure 17: Spectrograms of original and reconstructed signals, for the `3-tone` experiment.

5. RESULTS

5.1 Single-source sound recovery

In our first set of experiments, we tested the ability of our method to recover the sound from a single source. We used sounds of increasing complexity, starting from a single frequency tone, up until the *mhll* tune. We show only the most relevant results here, but additional plots can be found in the Appendix.



(a) All bins (above 250). (b) Selected bins.

Figure 16: Raw bins as constructed by WiTrack for the `3-tone` experiment. The left panel shows all the bins, while the right panel shows the bins automatically detected to contain the sound of interest.

The raw signal, as received from WiTrack's output, for the three tone example is shown in Figure 16a. Although the sound is visible in some of these bins, we rely completely on our fully automatic method to detect those bins carrying sound, and selecting the best ones.

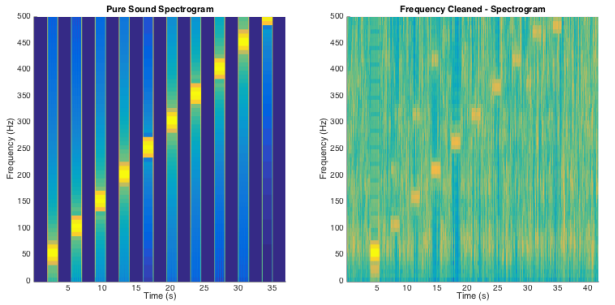
The full cleaning process is shown in Figure 13. This is done for all the bins, although here we show for reference only the one for the bin that was selected *a posteriori* to be the best one. After the pre-processing, we ran the bin selection subroutine, which selected the bins shown in Figure 16b. The tones are now conspicuously present in all of these bins, confirming the effectiveness of our bin selection algorithm. Among these, the highest-scored bin is number 276. The spectrogram for the cleaned version of this signal (shown in the bottom right of Figure 13) clearly shows the three tones in the correct frequencies (50, 100, 200 Hz) and at the right times. When played using MATLAB's command `soundsc`, there is still some noise left and some distortion due to the frequency cleaning process, but the original sequence of tones is clearly discernible.

The three-tone signal described above was purposely restricted to frequencies below 200 Hz as a precautionary measure. We expected the method to perform better in lower frequencies, since these produce wider oscillations in the speaker. Having verified the success in this restricted setting, we now verify the ability of *WiMic* to recover sounds in the full range of its theoretical frequency range (0, 500 Hz), which is determined by the sampling frequency. For this, we use the `steps` sequence described in Section 2. In Figures 18a - 18b. we compare side by side the true *pure* original sound and the recovered version from the top-scored bin (no. 276).

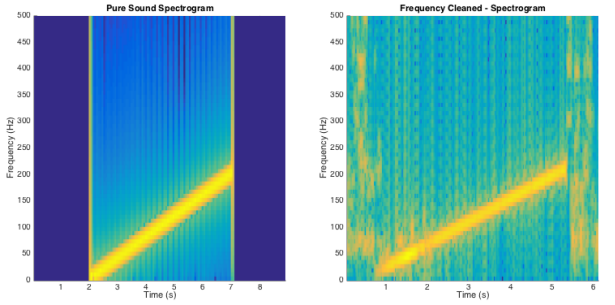
In the next scenario with which we experiment, we recover a *chirp*, which sweeps a range of frequencies continuously. This setting is more challenging for our approach, since our method for detecting sounds in bins relies on relatively flat regions of the spectrogram. Somewhat surprisingly, out method succeeded in identifying the most reliable bin also in this case, without the need to modify our default parameter setting. This is due to the fact that even if the sounds is continuous, the spectrogram presents a discretized version of this signal, which does exhibit flat regions. The original and reconstructed signals for the `sweep(1,200)` and `sweep(1,500)` are shown in Figures 18c - 18f. Note the striking similarity to the original sounds' spectrograms. The two chirps are completely distinguishable in the sound files produced from the cleaned signals, particularly for frequencies in the range (50 Hz, 300 Hz).

In the next single-source experiment, we play and reconstruct the *Mary had a little lamb* tune described in Section 2. Again, our method was able to recover the original tune, to the point that it can be clearly distinguished when played. The results are shown in Figures 18g - 18h.

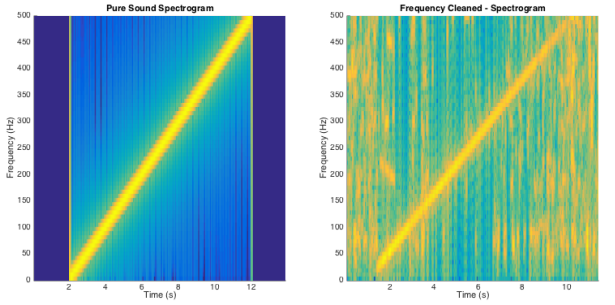
In our final single-source experiments, we attempted to recreate human speech as a test of our system on a more complex sound: the voices infrom JFK's inaugural speech or Morgan Freeman's lines from *The Shawshank Redemption*. Although our method wasn't able to find the most relevant bins in thes cases, using the top-rated bin for previous experiment resulted in a cleaned signal in which the voices are audible, although incomprehensible.



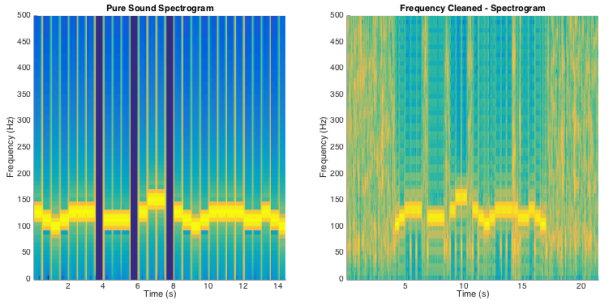
(a) `step`: original. (b) `step`: reconstructed.



(c) `sweep(200)`: original. (d) `sweep(200)`: reconstructed.



(e) `sweep(500)`: original. (f) `sweep(500)`: reconstructed.



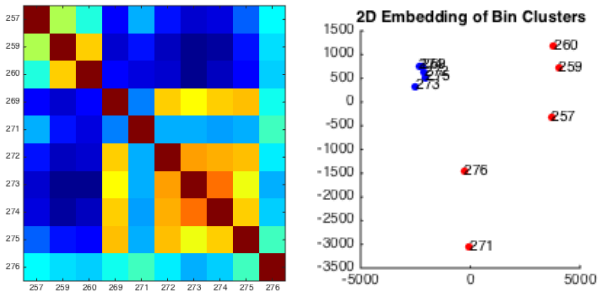
(g) `mhll`: original. (h) `mhll`: reconstructed.

Figure 18: Spectrograms of original (left) and reconstructed (right) signals, for the `step`, `sweep(200)`, `sweep(500)` and `mhll` sounds. The reconstructed version correspond to the top-scored bin in each case.

5.2 Multiple-source sound recovery

After having successfully recovered simple sounds from one source, we attempt recovery with two audio sources emitting sounds simultaneously.

In the first multiple-source experiment, we played a sequence of three pure frequency tones in each speaker. We used the same three frequencies in each speaker, but played in different order and with a slight offset. The latter was done to simulate real life scenarios in which independent sources will most likely not be coordinated. The first speaker played the sequence 50Hz, 100Hz, and 200hz, and the second one played 100Hz, 200 Hz, and 50 Hz. Each tone was played for three seconds with a gap of two seconds in between tones.



(a) Bin similarities. (b) Bin clustering.

Figure 19: Source disentangling with bin clustering for the simultaneous `3-tone` experiment.

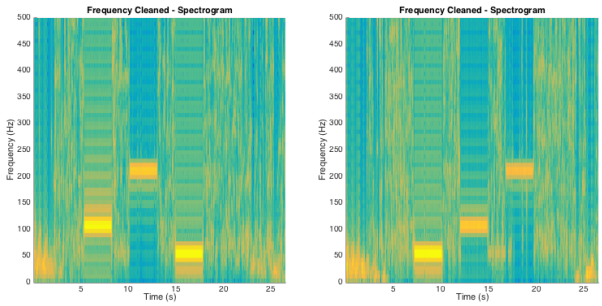
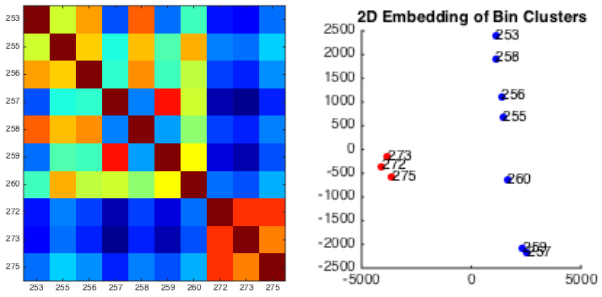


Figure 20: Reconstructed signals in the two source playing permuted versions of `3-tone` simultaneously. These spectrograms correspond to cleaned version of bins 257 and 275, identified by the method as being the most reliable bins for each of the two clusters.

After cleaning and processing the raw WiTrack bins, the sound detection algorithm found 13 bins with sounds. We kept the 10 bins which received the best score according to our method. The matrix of pair-wise similarities between these bins, computed as described in Section 4.3, is shown in Figure 19a. Note how there are two obvious clusters formed by the bins 257-260 and another one for 272-275. The bins 271 and 276 carry a mixture of the two sounds, but, surprisingly, exhibit the sound of the second speaker better, even though they correspond physically to regions in the space closer to the first one. We believe this shows that our method is indeed detecting sounds from external sources which are resonating to both sounds. Despite this confounding factors, our method correctly clusters all corresponding bins, as shown in Figure 19b.

Finally, we show in Figure 20 the cleaned version of the best-scored bin in each of the two groups. As can be seen from these spectrograms, our method was able to disentangle and correctly recover the two sounds. Note that timestamps in the x-axis confirm the fact that these two sequences of sounds were played simultaneously.



(a) Bin similarities. (b) Bin clustering.

Figure 21: Source disentangling with bin clustering for the `3-tone`/`mhll` experiment.

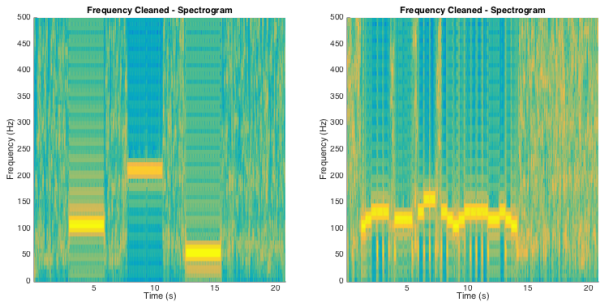


Figure 22: Reconstructed signals from two sources playing `3-tone` and `mhll` simultaneously. These spectrograms correspond, respectively, to the cleaned version of bins 256 and 273, identified by the method as being the most reliable bins for each of the two clusters.

.

In our second experiment with simultaneous sounds, we played the same sequence of three tones as above in one speaker and the `mhll` tune in the other. The similarities between the top scored bins and the corresponding clustering are shown in Figure 21. Note that in this case the sound of the second (closer) speaker has only three bins in the top-10 bins, and these are all very similar. The other seven bins in the top 10 are dominated by the sound of the second (farther) speaker. This is most likely due to the fact that `mhll`, played by the speaker whose location corresponds to bins 272-276, has higher frequencies that `3-tone`, so it is harder for environment materials to resonate to it. After selecting the top scoring bin from each cluster, our method returns the signals shown in Figure 22. Note again that the timestamps confirm the fact that these two signals occurred simultaneously. As before, our method correctly identified the strongest bins for each source, and was able to recover the sound contained in this successfully.

6. CONCLUSION

In this project, we proposed and developed a method to use reflected WiFi signals to detect and reconstruct sounds by detecting the vibrations of objects. This approach, which we call WiMic, is inspired by WiTrack's success in tracking movement [1] (to a degree of accuracy that even allowed the detection of heartbeats), and by the Visual Microphone [3], which successfully recreated sounds by analyzing silent videos for minute vibrations. While the primary advantage of such a system is to allow the reconstruction of sound (when a WiFi emitter is nearby) without recourse to a microphone, it offers another notable advantage over a microphone because such a system can naturally distinguish and isolate different sounds in its environment by analyzing the vibrations of objects at different distances and angles.

It also offers several advantages over a similar proposed system, WiHear, which tries to identify the poses of a person's mouth by use of WiTrack-like motion tracking, and to deduce from that the pronounced phoneme [6]. These advantages include the ability to reconstruct sounds, not just human speech. While the WiHear approach requires the system to learn how to recognize the phonemes of each individual person, the WiMic approach does not need this 'training' period.

Our system uses the same hardware as WiTrack with no modification, and is designed to detect vibrations of objects in its environment and process this information so as to accurately recreate the sounds being emitted in its environment. The processing includes denoising the signals it receives, removing signals which don't correspond to vibrations due to sound, clustering signals according to the sound producing them, and aggregating the signals in each cluster to reproduce the original sound.

We demonstrated that our method can reliably detect sounds by analyzing the vibrations of a speaker, up to a distance of several meters, and can automatically distinguish and process important signals as described above. We also successfully demonstrated multi-source distinguishing: when more than one source of sound is present (and they are sufficiently different in their distance to the WiTrack apparatus), it is able to isolate each sound and reproduce them independently. We also experimented with the range of sounds which could be detected and reproduced by WiMic by playing a "frequency sweep" sound, and found that we could reproduce the entire range up to 500 Hz. Since WiTrack has a sampling rate of 500 Hz, we cannot reconstruct frequencies above 500 Hz.

However, our method currently has two major limitations which must be overcome before WiMic can be used in practice. First, our experiments failed to reproduce sounds when measuring the vibrations of objects which were *not* the source of the sound; however, given the results achieved using video by the Visual Microphone group and the ability of WiTrack to discern heartbeats, we remain confident that it is possible to successfully read sounds from passive objects. Second, our method has had great difficulty reproducing complex sounds like human speech, being only able to reproduce a noisy sound in which a very faint human voice is barely discernible and incomprehensible from the excerpt of *The Shawshank Redemption*. However, again based on the success of the Visual Microphone group in reproducing human speech, we believe our general method should be able to handle speech effectively with some refinement.

Finally, although our current techniques can only detect frequencies below 500 Hz (likely due to the sampling rate of WiTrack itself), it might be possible to dramatically increase

this range by using data from multiple bins corresponding to the same source sound, in a similar vein to the Visual Microphone's successful recreation of frequencies higher than the sampling rate of the videos they analyzed. Currently, our bin aggregation techniques are mostly geared at an additional reduction of noise, but in the future (with a careful analysis of the mathematics behind WiTrack's detection techniques) it may be possible to weave together different signals and increase the effective sampling rate of WiMic.

7. REFERENCES

[1] F. Adib, Z. Kabelac, D. Katabi, and R. C. Miller. 3d tracking via body radio reflections. In *11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14)*, pages 317–329, Seattle, WA, Apr. 2014. USENIX Association.

[2] F. Adib, H. Mao, Z. Kabelac, D. Katabi, and R. C. Miller. Smart homes that monitor breathing and heart rate. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 837–846. ACM, 2015.

[3] A. Davis, M. Rubinstein, N. Wadhwa, G. Mysore, F. Durand, and W. T. Freeman. The visual microphone: Passive recovery of sound from video. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 33(4):79:1–79:10, 2014.

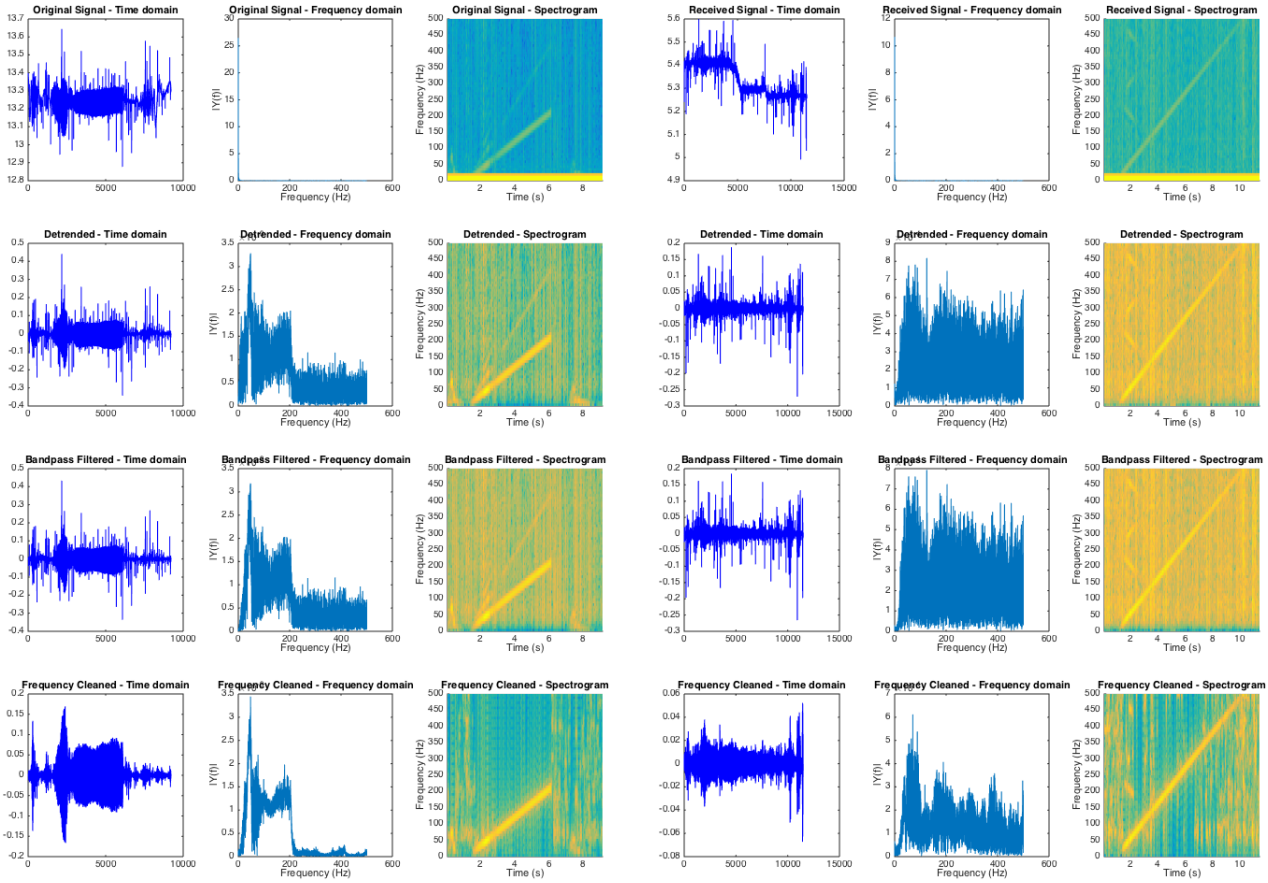
[4] M. A. Little and N. S. Jones. Generalized methods and solvers for noise removal from piecewise constant signals. i. background theory. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 2011.

[5] Q. Pu, S. Gupta, S. Gollakota, and S. Pate. Whole-home gesture recognition using wireless signals. In *Proceedings of the 19th Annual International Conference on Mobile Computing and Networking*, pages 39–50. ACM, 2013.

[6] G. Wang, Y. Zou, Z. Zhou, K. Wu, and L. M. Ni. We can hear you with wi-fi! In *Proceedings of the 20th annual international conference on Mobile computing and networking*, pages 593–604. ACM, 2014.

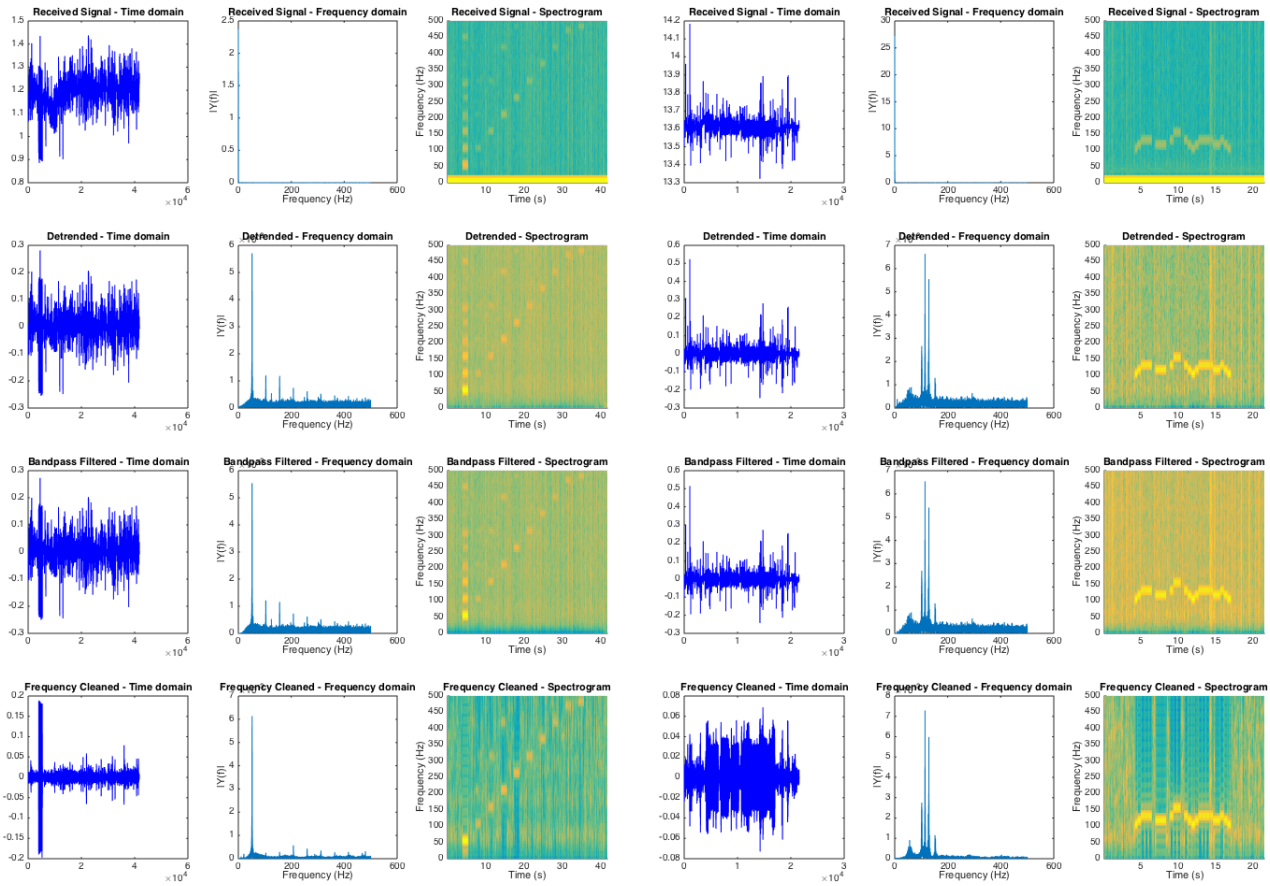
APPENDIX

We show here all the steps in the cleaning pipeline for the top-rated bins selected according to our algorithm, for all the experiments considered.



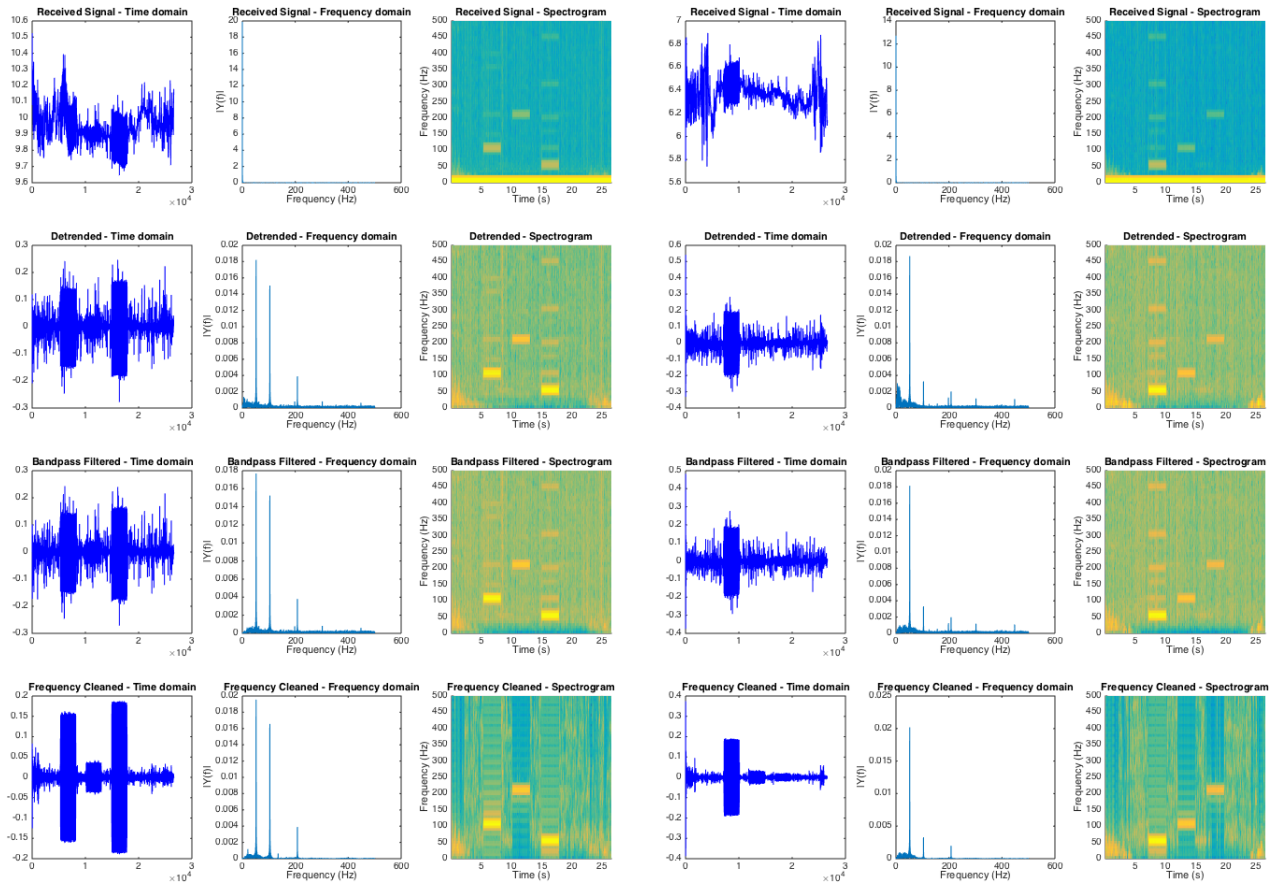
(a) All bins (above 250). (b) Selected bins.

Figure 23: Step-by-step de-noising process for the top-scored bin of `sweep(1,200)` (left) and `sweep(1,500)`. The top row shows the original signal, and each subsequent row adds another processing step on the previous one. For each step we show: time-domain plot, frequency-domain plot and spectrogram.



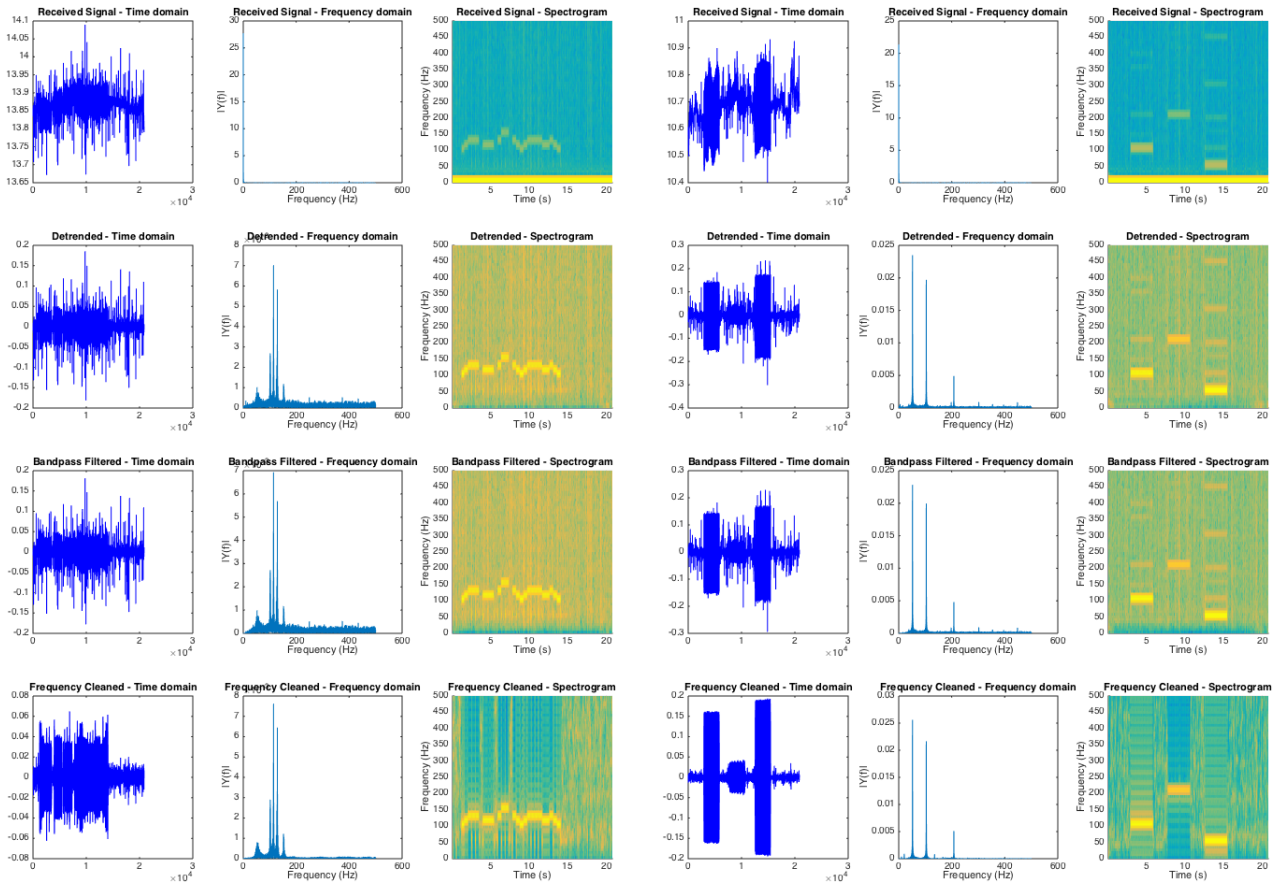
(a) All bins (above 250). (b) Selected bins.

Figure 24: Step-by-step de-noising process for the top-scored bin of `step` (left) and `lamb`. The top row shows the original signal, and each subsequent row adds another processing step on the previous one. For each step we show: time-domain plot, frequency-domain plot and spectrogram.



(a) All bins (above 250). (b) Selected bins.

Figure 25: Step-by-step de-noising process for the top-scored bin of `step` (left) and `lamb`. The top row shows the original signal, and each subsequent row adds another processing step on the previous one. For each step we show: time-domain plot, frequency-domain plot and spectrogram.



(a) All bins (above 250). (b) Selected bins.

Figure 26: Step-by-step de-noising process for the top-scored bin of `step` (left) and `lamb`. The top row shows the original signal, and each subsequent row adds another processing step on the previous one. For each step we show: time-domain plot, frequency-domain plot and spectrogram.