Ultrasonic Communication Using Consumer Hardware

Pascal Getreuer, Chet Gnegy, Richard F. Lyon, *Life Fellow, IEEE*, and Rif A. Saurous

***Abstract*—We have implemented a near-ultrasonic communication protocol in the 18.5–20 kHz band, which is inaudible to most humans, using commodity smartphone speakers and microphones to transmit and receive signals. The protocol described in this paper is a component of Google's Nearby platform, where near-ultrasound signals are used to establish copresence between nearby devices by transmitting a short token. High-frequency sound does not pass through walls (most energy is reflected), so identified devices are constrained to approximately the same room, "within earshot" of one another. Our protocol has a raw data rate of 94.5 b/s, and we find in real indoor environments that transmission between mobile devices is reliable at 2 m distance and often works at 10 m. We use direct-sequence spread spectrum modulation, which makes it highly robust to multipath, motion, and narrowband noise. We use a 127-chip pseudorandom code, repeating once per data symbol, and modulate its amplitude with orthogonal sine waveforms encoding 4-bit symbol values. We add the orthogonal sines to a constant "pedestal," which is inefficient in an information-theoretic sense, but makes synchronization easier. We describe a robust and computationally efficient transmitter and receiver implementations and show experiments on real and simulated data.**

***Index Terms*—Acoustics, audio systems, ultrasound, spread spectrum communication, device-to-device communication, indoor communication, mobile applications.**

I. Introduction

MOBILE devices conventionally transmit data using Bluetooth or Wi-Fi. Yet there are several advantages to using sound as a communication medium instead:

1) Establishing a Bluetooth or Wi-Fi connection takes multiple seconds, whereas sound playback or recording can start in tens of milliseconds.
2) Unlike Bluetooth and Wi-Fi, high-frequency sound does not pass through walls, which is useful for copresence to constrain the transmitter and receiver to approximately the same room, "within earshot."
3) Bluetooth is often disabled and is otherwise problematic [1].

Manuscript received October 10, 2016; revised June 23, 2017; accepted October 1, 2017. Date of publication October 23, 2017; date of current version May 15, 2018. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Martha Larson. *(Corresponding author: Pascal Tom Getreuer.)*

The authors are with the Google Research, Mountain View, CA 94043 USA (e-mail: getreuer@google.com; chetgnegy@google.com; dicklyon@google.com; rif@google.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2017.2766049

These properties of sound enable quick, localized exchanges, which are attractive for interactive mobile applications like device pairing, broadcasting a resource, or sharing content based on proximity.

We develop a communication protocol for transmitting small amounts of data as inaudible near-ultrasonic sound over short distances, using signals concentrated in 18.5–20 kHz. The system is based on direct-sequence spread-spectrum modulation (DSSS), which makes it robust to noise and reverberation in the environment.

The protocol we describe is a component of Google's Nearby platform,[1] which uses this protocol and other media to discover and send messages between nearby devices. Variants of the described protocol are used in Chromecast guest mode to authenticate a guest's mobile device automatically, in Google Play Games to find nearby players, and are behind Audio QR in Google Tez for discovering and pairing parties in a cash transaction.

Our goal is establishing physical copresence for such applications by transmitting a short token as inaudible sound.

a) Challenges: Data transmission as audible sound is difficult socially, since those who hear it may find it distracting or offensive, depending on the modulation scheme. Transmitting near-ultrasound 18.5–20 kHz with low signal strength avoids these problems by using spectrum where human hearing has low or no sensitivity. Mobile device speakers are fairly quiet with speaker response dropping off rapidly in this range, so that the signal is below the threshold of hearing for most humans. For example, we measure a Nexus 6 speaker playing an 18 kHz tone at max volume produces 74 dB SPL on axis at 10 cm, whereas the hearing threshold for most listeners is above 86 dB SPL at 18 kHz [2].

The main technical challenges with sound-based communication are multiple propagation paths, reverberation, interference, Doppler shift from relative motion, and hardware limitations of speakers and microphones. For interactive use on mobile devices, we also consider weak signal strength, latency, and computation cost as important constraints.

In typical indoor environments, sound reflects from walls and surrounding objects. There are usually multiple propagation paths from the transmitter to the receiver, even when devices have line of sight, and these interfere in complicated ways. Sound reflects more than radio signals do from most indoor building materials. Hard smooth surfaces like gypsum wallboard

[1] https://developers.google.com/nearby

and glass windows are very reflective at higher frequencies, bouncing back over 95% energy of the incident wave [3] and creating numerous significant propagation paths. The RT_{60} value, the time needed for reverberant sound energy in a room to decay by 60 dB, is around 500–2000ms in 1–4 kHz for office rooms and classrooms, though typically less at ultrasonic frequencies [4].

For instance with a frequency-shift keying (FSK) modulation scheme, the reverberation from one symbol interferes with subsequent symbols (intersymbol interference), and FSK must use sufficiently separated frequencies to tolerate Doppler shift. These problems practically limit FSK-modulated sound to low bit rates, with symbol times proportional to the room's RT_{60} and few bits per symbol.

b) Previous Work: Lopes and Aguiar [5] describe several experiments to demonstrate low-bitrate communication with sound using FSK, amplitude-shift keying (ASK), and frequency-hopping spread-spectrum (FHSS) modulation.

Hazas and Ward [6], later refined by Alloula and Hazas [7], use ultrasound in the 20 to 100 kHz range for position sensing of mobile nodes based on time of arrival of DSSS-modulated signals. The system does not handle Doppler shift due to motion of the nodes, though it is described how the system might be extended to do so. Vallidis [8] applies DSSS-modulated ultrasound to tracking parts of the body for human-computer interaction. Galluccio *et al.* [9] and Santagati *et al.* [10]–[12] investigate the feasibility and physical constraints of ultrasonic data communication between devices that are implanted or worn on the human body. They note that DSSS modulation with its robustness to multipath has promise for such applications.

Ultrasonic communication has been studied as a means to create air-gap covert channels. Hanspach and Goetz [13], [14] demonstrate covert near-ultrasonic communication between laptops with FHSS. Deshotels [15] and Do *et al.* [16] consider near-ultrasonic sound for exfiltrating data from mobile devices using FSK. See also the survey by Carrara [17].

Several works investigate communication between smart devices using sound under 22 kHz. Matsuoka *et al.* [18] describe an OFDM-based music watermarking system using sound in the range 5–10 kHz to transmit 40 bps at ranges on the order of 1 m. Yun *et al.* [19] describe a system using the modulated complex lapped transform and sound in 6.4–8 kHz to transmit 0.6 kpbs at ranges up to 3 m. Nandakumar *et al.* [20] describe a system using audible sound in 0–22 kHz with OFDM modulation, transmitting 2.4 kbps at a few centimeters range for NFC-like interaction between smartphones. Lee *et al.* [21] describe low data rate chirp signal-based scheme, capable of 16 bps at ranges up to 25 m.

c) DSSS: Early spread-spectrum systems were developed by research programs in the United States Army, Navy, and Air Force in the 1950's and 60's with the goal of communicating with low detectability and resistance to jamming [22]. In 1973, the Department of Defense initiated development of the Global Positioning System (GPS) based on DSSS. A constellation of U.S. Air Force NAVSTAR satellites transmits DSSS-modulated signals to allow a GPS receiver to compute its precise position by multilateration [22].

Although most interest in spread-spectrum techniques has been in radio communication rather than sound, several previous works have applied spread-spectrum techniques to audio steganography, such as hiding copyright information in music. For example, Kirovski and Malvar [23], Lazic and Aarabi [24], and Kang *et al.* [25] use DSSS to hide data in sound. In these works, the watermark is added to the source audio as a DSSS-modulated signal in a transformed domain.

Spread-spectrum techniques have been applied in underwater communication, where radio waves have extremely limited propagation. For example, Sozer *et al.* [26] note that acoustic waves in shallow water environments have significant and time-varying surface-bottom reflections. They use DSSS to overcome these multipath effects.

d) This Work: Our system transmits data as sound concentrated in the band 18.5–20 kHz, which is inaudible to most humans but narrowly within the operating range of speakers and microphones of most recent mobile devices. The data is modulated on a DSSS signal which in turn single-sideband modulates a sinusoidal carrier. Each period of the DSSS code signal defines one "frame" of the signal, on which one symbol of data is encoded by orthogonal multiple frequency-shift keying (MFSK) modulation of the code's amplitude.

We choose DSSS over other possible modulation schemes for its particular resilience to multipath. Compared to schemes like FSK or PSK, DSSS resolves the signal's time of arrival with much finer resolution, sufficient to correspond distinct paths to distinct correlation peaks. DSSS is robust to narrowband, wideband, and impulsive interference [27]. Additionally, DSSS can overcome Doppler and weak signal strength, which is necessary since our system is used by moving handheld devices with weak speakers.

Unique aspects of our system include:

1) The 18.5–20 kHz signal is inaudible to most humans, yet realizable with the commodity speakers and microphones in most recent mobile devices.
2) The protocol is resistant to reverberation, background interference, and Doppler shift due to motion. In real indoor environments, transmission between mobile devices is reliable at 2 m and often works at 10 m.
3) For low latency, synchronization is determined by an exhaustive but computationally efficient search over a finely sampled grid of time and frequency offsets so that the signal is immediately acquired.

We believe this is the first work to use DSSS modulation for sound-based communication where both transmitter and receiver are mobile devices. Also, unlike most previous work with sound-based communication, our design explicitly addresses Doppler shift and latency.

Basic properties of our DSSS-based protocol (notations are defined in the next section):

$$\begin{aligned}\text{Chip rate} &= 1/T_c = 3 \text{ kHz}\\ \text{Code bandwidth} &= 1/(2T_c) = 1.5 \text{ kHz}\\ \text{Raw data rate} &= (\log_2 K)/T_s \approx 94.5 \text{ bits/s}\\ \text{Spreading factor} &= (\text{bandwidth})/(\text{raw data rate}) \approx 15.9\end{aligned}$$

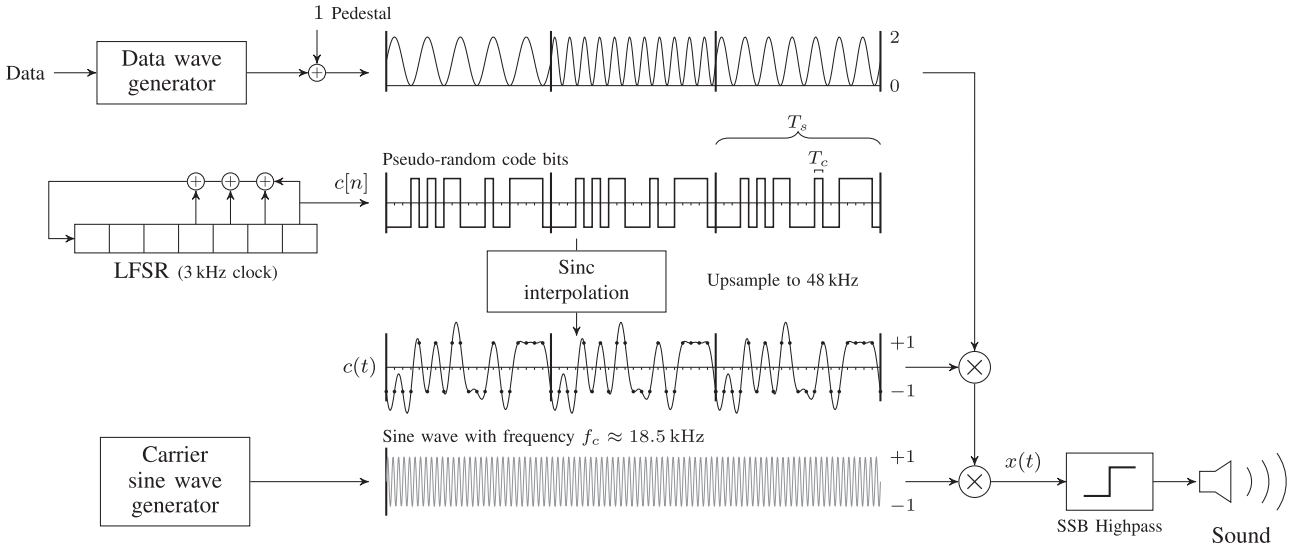

Fig. 1. System model for encoding. The code signal is modulated by the data signal, and this spread spectrum signal modulates the carrier, shifting the energy up to the carrier frequency plus and minus half the code rate. It is then single-sideband filtered and broadcast as inaudible sound.

e) Notations: Continuous-time signals are denoted with lowercase letters as functions of time t in units of seconds, e.g., $x(t)$. We denote continuous-time convolution of two possibly complex-valued functions $x(t)$ and $y(t)$ by $*$ and (cross-)correlation by $\star$,

$$(x * y)(t) := \int_{-\infty}^{\infty} x(s)y(t-s)\,ds, \tag{1}$$

$$(x \star y)(t) := \int_{-\infty}^{\infty} x(s)^* y(t+s)\,ds, \tag{2}$$

where $(\cdot)^*$ denotes complex conjugation. Discrete-time signals are denoted as sequences indexed with square brackets $x[n]$. We define the convolution and correlation of discrete sequences analogously. Continuous Fourier transform is denoted as a function of f, using the following normalization,

$$X(f) := \int_{-\infty}^{\infty} x(f)\mathrm{e}^{-i2\pi ft}\,dt. \tag{3}$$

The following is a list of parameters in our model. Values used in our implementation are shown in parentheses:

N	Chips per code period (127)
T_s	Duration (s) of one symbol frame (≈ 42.3 ms)
T_c	Duration (s) of a chip (s), $T_s/N (\approx 333\,\mu$s)
f_c	Carrier frequency (Hz) (≈ 18.5 kHz)
K	Number of distinct data symbol values (16)
$c[n]$	Discrete code sequence of period N
$c(t)$	Sinc interpolation of $c[n]$ with period T_s
$c_s(t)$	The code signal $c(t)$ restricted to $[0, T_s)$
$d_k(t)$	Data wave encoding the kth symbol value
h^{SSB}	Single sideband (SSB) highpass filter

f) Outline: Section II introduces the system model. Section III discusses the DSSS code signal, and Section IV discusses the choice of data wave functions and encoding of the data signal. Section V describes synchronization. Section VI describes our implementation of the protocol. Section VII builds on top of the symbol-level protocol to transmit short tokens of data. Section VIII shows results. Section IX concludes and discusses directions for future work.

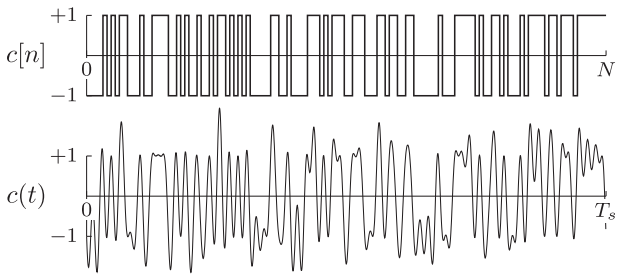

Fig. 2. One frame of a code sequence $c[n]$ with period $N = 127$ and its interpolation $c(t)$.

II. System Model

This section is an overview of our system with a continuous-time description. Later sections develop further details.

A. Encoding

Fig. 1 illustrates how data is encoded and broadcast as inaudible sound.

The *code sequence* $c[n]$ is an N-periodic pseudorandom sequence of $+1$ and -1 values, for example generated by a linear feedback shift register (LFSR). The code sequence is sinc-interpolated to obtain a T_s-periodic function $c(t)$, which is in the frequency band $|f| \le 1/(2T_s)$ and each "chip" of the sequence has duration $T_c := T_s/N$ (Fig. 2).

We call each time interval $[T_s m, T_s(m+1)), m \in \mathbb{Z}$, a *frame* of the signal. Each length-T_s frame contains one period of the code and is used to encode one data symbol of value k. Let $k[m] \in \{0, 1, \ldots, K\}$ be the symbol to encode in the mth frame, and define the data signal $d(t)$ as

$$d(t) = d_{k[m]}(t - T_s m), \quad T_s m \le t < T_s(m+1). \tag{4}$$

The data wave functions $d_0, \ldots, d_K$ are an orthogonal set of sinusoids of different frequencies. The data signal is spread with the code $c(t)$ and modulated on a sinusoidal carrier as

$$x(t) = \sin(2\pi f_c t) c(t)\big(d(t) + 1\big), \tag{5}$$

where f_c is the carrier frequency. The "+1" term is the *pedestal*, an added steady code signal component that modulates the carrier but is not modulated by data. The receiving side detects this pedestal to synchronize the signal.

The "+1" pedestal in (5) spends more than half of the signal energy on a component that does not contribute to distinguishing between data values, which is wasteful in an information-theoretic sense and is probably the biggest disadvantage of our protocol. We include it for easier synchronization (see Section V): the pedestal contributes a large response when the code signal is correlated with the received signal at baseband, allowing us to synchronize the signal without having to guess and correlate for the data signal. Separating data decoding from synchronization this way reduces computational cost.

Frequencies below f_c are filtered out to make a single-sideband (SSB) signal. The transmitted signal is

$$x^{\text{SSB}} = h^{\text{SSB}} * x, \quad H^{\text{SSB}}(f) = \begin{cases} 1 & |f| \geq f_c, \\ 0 & |f| < f_c, \end{cases} \tag{6}$$

where $*$ denotes convolution and h^{SSB} is a brick wall highpass filter with cutoff f_c. We perform this filtering in the FFT domain to one token repetition of the signal (explained in Section VII), setting coefficients below f_c to zero.

Since $c(t)\big(d(t)+1\big)$ is real, the spectrum of the double-sideband signal $X(f)$ is Hermitian symmetric around f_c and can be filtered to keep only the upper sideband without loss of information. Ignoring the data signal, x^{SSB} is in the band $[f_c, f_c + 1/(2T_c)]$. Multiplication by the data signal spreads signal energy several hundred Hz above this range. Spreading below f_c is prevented by the SSB highpass filtering (6).

We reserve symbol value K as a "spacer" to mark the beginning of the data sequence (explained in Section VII). Counting symbol values $\{0, \ldots, K-1\}$, the gross bitrate is $\frac{\log_2 K}{T_s} \approx$ 94.5 bits/s and the bandwidth-to-bitrate ratio is

$$\frac{1}{2T_c}\frac{T_s}{\log_2 K} = \frac{N}{2\log_2 K} = 15.875 \ (12.0\,\text{dB}). \tag{7}$$

In our implementation, f_c is approximately 18.5 kHz and the one-sided code bandwidth is 1.5 kHz so that the transmitted signal is concentrated in the 18.5–20 kHz band.

B. Decoding

Fig. 3 shows how data is decoded from received sound. This figure is simplified; particularly, synchronization is not shown.

The received signal $y(t)$ is first downconverted to undo the effect of the f_c carrier, obtaining a raw complex-valued baseband signal

$$b^{\text{raw}}(t) = y(t)\mathrm{e}^{-i2\pi f_c t}. \tag{8}$$

No attempt is made to synchronize the carrier phase or compensate for Doppler shift at this stage. Demodulation shifts the spectrum down by f_c so that the signal spectrum is in $[0, 1/(2T_c)]$. Since $y(t)$ is real-valued, it also has a negative copy of the spectrum approximately in $[-20\,\text{kHz}, -18.5\,\text{kHz}]$ that shifts to $[-38.5\,\text{kHz}, -37\,\text{kHz}]$ (or generally, $[-2f_c - 1/(2T_c), -2f_c]$), which for a 48 kHz sample rate aliases to $[9.5\,\text{kHz}, 11\,\text{kHz}]$. To remove the negative alias and reduce out-of-band noise, the raw baseband signal is filtered

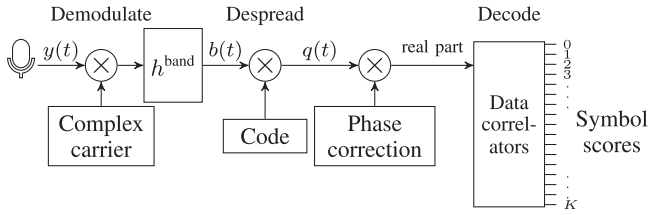

Fig. 3. System model: decoding.

$$b = h^{\text{band}} * b^{\text{raw}}, \tag{9}$$

where h^{band} is a complex filter with approximately unit gain over $[0, 1/(2T_c)]$ and attenuating negative frequencies $f < 0$ and higher frequencies $f > 1/(2T_c)$. More details are given in Section VI-A.

Before decoding, the baseband signal must be time synchronized, finding a frame start time τ such that $b(\tau + t), t \in [0, T_s)$, is aligned with a symbol frame in the signal. We defer synchronization to Section V and describe here how to decode a frame that has already been synchronized. First, the frame is despread by multiplying with the code signal,

$$q(t) = c(t)b(\tau + t). \tag{10}$$

Despreading approximately cancels the code signal component so that $q(t) \approx \big(d_k(t) + 1\big)\mathrm{e}^{i\psi_0}$, where $\mathrm{e}^{i\psi_0}$ is the unknown complex phase offset due to the carrier. Since the data signal plus pedestal $\big(d_k(t) + 1\big)$ is nonnegative, we estimate ψ_0 by

$$\psi = \angle \frac{1}{T_s}\int_0^{T_s} q(t)\,dt, \tag{11}$$

which maximizes the real part of the average of $\mathrm{e}^{-i\psi}q$. Inner products with the data waves $d_0, \ldots, d_K$ are then computed to determine a symbol score for each k,

$$s_k = \langle \mathrm{Re}\{\mathrm{e}^{-i\psi}q\}, d_k \rangle, \quad k = 0, \ldots, K. \tag{12}$$

A simple decoding rule is to select the symbol with the largest score as the decoded symbol, $\arg\max_k s_k$.

III. Code Signal

This section develops in further detail the discrete $N = 127$-periodic pseudorandom code sequence $c[n]$. We generate a $\{0, 1\}$-valued sequence $b[n]$ from the LFSR recurrence

$$\begin{cases} b[n] = b[n-4] \oplus b[n-5] \oplus b[n-6] \oplus b[n-7], \\ b[-7] = b[-6] = \cdots = b[-1] = 1, \end{cases} \tag{13}$$

where $\oplus$ denotes XOR, then map to $\{-1, +1\}$ values by $c[n] = 2b[n] - 1$. The sequence $b[n]$ is an example of a maximum length sequence [28]. The sequence $c[n]$ has the attractive property that

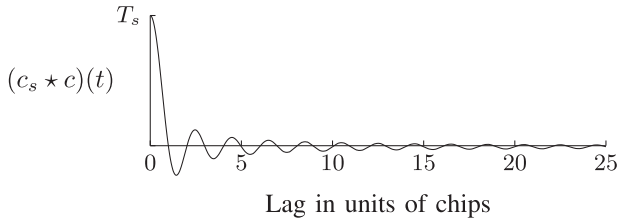

Fig. 4. Autocorrelation of the interpolated code signal $(c_s \star c)(t)$.

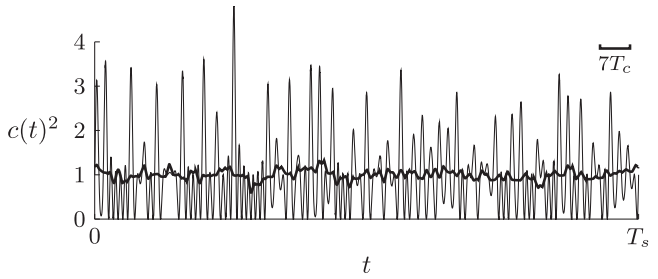

Fig. 5. Plot of $c(t)^2$ (thin line) and its moving average over a window of size $7T_c$ (thick line).

it is nearly orthogonal to all shifts of itself where the shift is nonzero modulo the period,

$$\sum_{m=0}^{N-1} c[m]c[m+n] = \begin{cases} N & \text{if } n = 0 \mod N, \\ -1 & \text{if } n \neq 0 \mod N. \end{cases} \tag{14}$$

A design consideration is that a longer code period improves noise robustness, but increases the receiver's cost in computing correlations for synchronization. Another limitation is that maximum length sequences only exist for periods $N = 2^k - 1$, $k \in \mathbb{N}$, so there are only a few practical choices for the period.

The code sequence is sinc-interpolated to obtain a T_s-periodic function $c(t)$ (Fig. 2),

$$c(t) = \sum_{n=-\infty}^{\infty} c[n] \frac{\sin\big(\pi(t/T_c - n)\big)}{\pi(t/T_c - n)}. \tag{15}$$

To minimize the bandwidth used for transmission, we use sinc interpolation (as opposed to say zero-order hold) so that the code signal is bandlimited to $|f| \leq 1/(2T_c)$.

Next, we investigate two properties of $c(t)$ that we use later on: its autocorrelation and its pointwise square.

A. Autocorrelation

The code signal's autocorrelation is important because it determines the precision of our time synchronization scheme in Section V.

For notational convenience, we define $c_s(t)$ to be $c(t)$ restricted to one symbol frame $[0, T_s)$,

$$c_s(t) := \begin{cases} c(t) & \text{if } 0 \leq t < T_s, \\ 0 & \text{otherwise,} \end{cases} \tag{16}$$

and note that periodic autocorrelation of c on $[0, T_s)$ is the same as the nonperiodic correlation $c_s \star c$.

We calculate the code signal autocorrelation to be

$$(c_s \star c)(t) = T_c\Big((N+1)\frac{\sin(\pi t/T_c)}{N\sin(\pi t/T_s)} - 1\Big). \tag{17}$$

Interpolation has an effect on the autocorrelation, changing it from nearly a discrete impulse (14) to a sinc-like function (Fig. 4). The autocorrelation value is maximal at $t = 0$ and has its main lobe over $|t| < T_c$. This near-ideal autocorrelation enables a DSSS receiver to determine the time synchronization with fine precision, on the order of one chip.

B. Pointwise Square

We show in this section that local averages of the pointwise square $c(t)^2$ are approximately equal to one, so that spreading and despreading are possible in continuous time with pointwise multiplication by $c(t)$.

Since $(\pm 1)^2 = 1$, the operation of pointwise multiplication by the discrete sequence $c[n]$ is its own inverse. This property is preserved in the interpolated code signal $c(t)$ in an average sense. The Fourier series coefficients (C_k) of $c(t)$ have flat power $|C_k|^2 = (N+1)/N^2$, $1 \leq |k| \leq \lfloor N/2 \rfloor$, with an exception $|C_0|^2 = 1/N^2$ at DC. So by Parseval's equality,

$$\frac{1}{T_s}\int_0^{T_s} c(t)^2\, dt \overset{\text{Parseval}}{=} \sum_{k=-\lfloor N/2 \rfloor}^{\lfloor N/2 \rfloor} |C_k|^2 = 1. \tag{18}$$

Thus the squared code signal $c(t)^2$ equals one on average. Fig. 5 plots $c(t)^2$ and its zero-phase moving average over a window of size $7T_c$. The function $c(t)^2$ itself is not equal to one pointwise and oscillates rapidly around it, but its moving average is near one.

We could have made $c(t)^2$ pointwise equal to one by replacing sinc interpolation in (15) with zero-order hold, however, this would increase the bandwidth of the signal. Moreover, pointwise equality is unnecessary since the despread signal $q(t) = c(t)b(\tau + t)$ (10) is needed only in inner products with the data waves, $\langle \mathrm{Re}\{\mathrm{e}^{-i\psi} q\}, d_k \rangle$. This average-sense approximation is sufficient provided that the data waves are slowly varying relative to the chip duration.

IV. Data Signal

Data is encoded by modulating the code with a data signal. The data signal $d(t)$ is constructed by concatenating data wave frames $d_k(t)$ of length T_s, where each frame encodes one symbol. This data signal is then multiplied with the code signal and carrier to construct the transmitted signal (5). The data wave functions $d_0, \ldots, d_K$ are orthogonal sinusoids

$$d_k(t) = \sin\left(\frac{2\pi}{T_s}(4+k)t\right), \quad k = 0, \ldots, K, \ t \in [0, T_s]. \tag{19}$$

This choice of data waves is attractive to limit the used spectral bandwidth and for continuity at the frame endpoints. A minimum of 4 cycles per frame is used so that data waves are less confusable with slow amplitude changes caused by channel

fading, automatic gain control, or from incorrect estimations of the Doppler shift. More generally, $\{d_k\}$ could be any set of slowly-varying zero-mean orthogonal functions.

The data signal is added to the pedestal, a constant offset of one, then multiplied pointwise with the code to *spread* it to form the baseband signal

$$b(t) = c(t)\big(d(t) + 1\big). \tag{20}$$

In the frequency domain, (20) corresponds by the convolution-multiplication property to $B(f) = (C * D)(f) + C(f)$ (these spectra should be interpreted as distributional or coming from finite-duration signals). The data signal's spectrum $D(f)$ is spread by the spectrum $C(f)$; this is the "spread spectrum" characteristic for which DSSS is named.

When correlated with one period of the code signal, the baseband signal (20) becomes

$$(c_s \star b)(t) = \big(c_s \star (c \cdot d)\big)(t) + (c_s \star c)(t), \tag{21}$$

where $\cdot$ denotes pointwise multiplication. There are two terms, one corresponding to the data signal and the other to the pedestal. The pedestal term is the code autocorrelation $(c_s \star c)(t)$ (17), having a large peak at lag $t = 0$ (Fig. 4).

Compared to the pedestal term, the data term $\big(c_s \star (c \cdot d)\big)(t)$ in (21) is smaller and contributes little. This is essential since it enables us to perform time synchronization using only the code correlation, and not requiring computation of correlations for each possible data symbol.

To assess the data term of the correlation, we approximate by summing chip midpoints,

$$\begin{aligned}\sum_{m=0}^{N-1} \int_{(m-1/2)T_c}^{(m+1/2)T_c} & c(s)c(s+t)d(s+t)\,ds \\ &\approx T_c \sum_{m=0}^{N-1} c[m]c[m+n]d[m+n],\end{aligned} \tag{22}$$

where $t = nT_c$. For $n = 0$, since the data waves are zero mean over the frame, (22) is

$$T_c \sum_{m=0}^{N-1} c[m]^2 d[m] = T_c \sum_{m=0}^{N-1} d[m] = 0. \tag{23}$$

For $n \neq 0$, lags t off by more than a chip, suppose that $c[m]$ is a random nonperiodic sequence of i.i.d. zero-mean $+1$ and -1 values, then (22) has zero mean and variance

$$\begin{aligned}&\mathbf{E}\left[\left(T_c \sum_{m=0}^{N-1} c[m]c[m+n]d[m+n]\right)^2\right] \\ &= {T_c}^2 \sum_{m=0}^{N-1} d[m+n]^2 = \frac{{T_s}^2}{N^2} \sum_{m=0}^{N-1} d[m+n]^2.\end{aligned} \tag{24}$$

Therefore, the magnitude $\big|\big(c_s \star (c \cdot d)\big)(t)\big|/T_s$ is on the order of $1/\sqrt{N}$ times the root-mean-square value of $d(t)$ over $[t, t + T_s]$, whereas $\big|(c_s \star c)(t)\big|/T_s$ has a peak of 1. The root-mean-square value of each data wave (19) is $\sqrt{1/2}$, suggesting that the standard deviation is close to $1/\sqrt{2N} \approx 0.063$, and by normal approximation, 95% of $\big|c_s \star (c \cdot d)\big|/T_s$ is below 0.123.

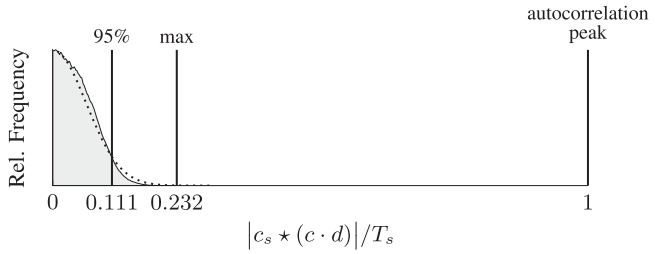

Fig. 6. Histogram of correlation $\big|\big(c_s \star (c \cdot d)\big)(t)\big|/T_s$ sampled over t and all pairs of symbols. 95% of the correlation values have magnitudes below 0.111. The dotted line shows the normal approximation based on (24), for which the 95th percentile is 0.123.

To verify the correlation magnitude numerically, we construct each possible data signal $d(t)$ that alternates between two symbol values $(k_1, k_2, k_1, k_2, \ldots)$, and compute the histogram of $\big|\big(c_s \star (c \cdot d)\big)(t)\big|/T_s$ aggregated over each such data signal (Fig. 6). For 95% of the correlation values, the magnitude is below 0.111, close to our normal approximation estimate of 0.123 and well below the autocorrelation peak.

Therefore, the correlation $c_s \star b$ is approximately $c_s \star c$, the autocorrelation of c. The autocorrelation is T_s at $t = 0$ mod T_s and small otherwise, allowing to synchronize the signal by locating large values.

V. Synchronization

The central challenge in developing the protocol is synchronizing the locations of the frames on the receiving side. The time when the signal begins is unknown to the receiver. In addition to an unknown offset in time, relative motion (which must be tolerated with handheld devices) introduces Doppler shift, scaling observed frequencies as

$$\tilde{f} = \left(1 + \frac{v}{340\ \text{m/s}}\right) f, \tag{25}$$

where 340 m/s is the approximate speed of sound and v is the relative velocity, in which positive velocity means moving closer together. Since the carrier frequency f_c is large compared to the signal bandwidth, motion approximately has the effect of shifting the received signal in frequency by $\delta = f_c v/340$. At a relaxed walking pace [29] of $v = 1$ m/s, Doppler shift over the band $[f_c, f_c + 1/(2T_c)]$ varies from 54.4 Hz at the lower edge to 58.8 Hz at the upper edge.

Therefore, we approach synchronization as a search for an offset in time and an offset in frequency. We assume Doppler shift is approximately constant over a frame, but estimate the Doppler shift separately for each frame, so that a slowly varying relative velocity between the devices is tolerated.

Searching over all combinations of time and frequency offsets is a big space of possibilities to consider. Conventional DSSS receivers use serial search techniques to acquire the code phase, for example by testing the correlation with the received signal at one code phase $\int c(s)b(\tau + s)\,ds$, and slowly varying the code

phase τ until a large correlation is found. These algorithms track efficiently once locked, but may take some time to acquire the correct code phase [27], [30], which would not be suitable for short messages. Low latency is critical for our application, so we make a brute force search to acquire as early as possible. We discuss computation cost in Section VI-B.

A. Code Correlation

Let $y(t)$ denote the received signal. Ignoring modulation by the data signal and filtering, we model $y(t)$ as

$$y(t) = \sin\big(2\pi(f_c - \delta_0)(t - \tau_0)\big)c(t - \tau_0), \tag{26}$$

where our goal is to estimate τ_0 and δ_0, the time of arrival and Doppler shift of the carrier frequency. The code factor in this model is approximate because it incorporates Doppler as a shift in frequency, rather than a scaling, and it ignores time dilation, $c(t) \approx c(t/(1 + \frac{\delta}{f_c}))$.

After downconversion to baseband and filtering to remove the negative alias, the signal is approximately

$$b(t) = \frac{1}{2i}\mathrm{e}^{i\psi_0}\,\mathrm{e}^{-i2\pi\delta_0 t}c(t - \tau_0), \quad \psi_0 = -2\pi(f_c - \delta_0)\tau_0. \tag{27}$$

In addition to unknown time and frequency offsets τ_0 and δ_0, there is a complex phase offset ψ_0 due to the carrier. We ignore ψ_0 for now. It will be estimated and canceled in a later step.

To determine the time of arrival τ_0 and Doppler offset δ_0, define the *acquisition score*, the square magnitude of modulated correlation,

$$a^{\text{raw}}(\tau, \delta) = \left|\frac{1}{T_s}\int_0^{T_s} \mathrm{e}^{i2\pi\delta s}c(s)b(\tau + s)\,ds\right|^2, \tag{28}$$

which measures how well the code signal matches with the baseband signal at a symbol frame beginning at time τ and having Doppler shift δ. We seek to maximize this function over (τ, δ), which is large when synchronized with the code at $\tau \approx \tau_0 \mod T_s$, $\delta \approx \delta_0$, and small otherwise.

We evaluate all possibilities as a brute force search for (τ_0, δ_0), sampling $a(\tau, \delta)$ exhaustively over a grid of τ and δ combinations. This allows synchronization to be achieved as early as possible to minimize latency of our protocol. Direct computation of correlation with so many possible signals would be too expensive, but an approximation allows us to do this with moderate computation (in Section VI-B).

Substituting (27) into (28) obtains the ambiguity function

$$a^{\text{raw}}(\tau, \delta) = \left|\frac{1}{T_s}\int_0^{T_s} \mathrm{e}^{i2\pi(\delta - \delta_0)s}c(s)c(\tau - \tau_0 + s)\,ds\right|^2. \tag{29}$$

Fig. 7 shows this ambiguity function. The acquisition score is maximal with value 1.0 at $\tau = \tau_0$, $\delta = \delta_0$, where the width of the main peak is about T_c (≈ 333 μs) in time and one cycle per frame $1/T_s$ (≈ 23.6 Hz) in frequency. The highest sidelobe peak is 0.0515. When $\delta = \delta_0$, it reduces to the square of (17),

$$a^{\text{raw}}(\tau, \delta_0) = \left|\frac{1}{T_s}(c_s \star c)(\tau - \tau_0)\right|^2, \tag{30}$$

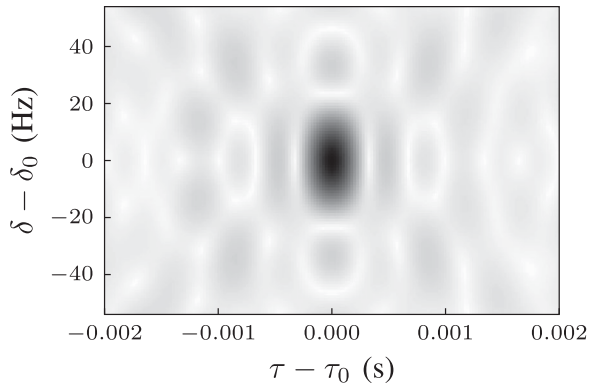

Fig. 7. $a^{\text{raw}}(\tau, \delta)$, the ambiguity function (29), over frame start time τ and Doppler offset δ. The maximal value is 1.0, and the highest sidelobe peak is 0.0515.

which is maximal when $\tau = \tau_0 + nT_s$. In the other dimension, setting $\tau = \tau_0$ and using $c^2(s) \approx 1$ shows that $a^{\text{raw}}(\tau_0, \delta)$ is approximately a squared sinc with peak at $\delta = \delta_0$,

$$\left|\frac{1}{T_s}\int_0^{T_s} \mathrm{e}^{i2\pi(\delta - \delta_0)s}\,ds\right|^2 = \left|\frac{\sin\big(\pi(\delta - \delta_0)T_s\big)}{\pi(\delta - \delta_0)T_s}\right|^2. \tag{31}$$

In our receiver, the frame start time τ is sampled uniformly at sample rate F_b, and δ is sampled over a fixed set Δ of 17 test frequencies. Specifying a maximum motion tolerance of 1 m/s, the maximum possible Doppler shift to the carrier frequency is about 54 Hz, or 2.3 cycles per frame. We sample these test frequencies uniformly with a spacing of 0.3 cycles per frame (or reciprocally a Doppler shift of one cycle per frame corresponds to $1/0.3 \approx 3.3$ test frequencies), $\Delta = \{0, \pm\frac{0.3}{T_s}, \pm\frac{0.6}{T_s}, \ldots, \pm\frac{2.4}{T_s}\}$.

Although the primary effect of Doppler shift is an offset in the received carrier frequency, it also dilates (or compresses) in time the received duration of a frame. This time dilation effect is ignored in (26). Suppose that $c(t)$ is transmitted with motion, then the receiver observes a dilated (or compressed) version $\tilde{c}(t) = c\big(t/(1 + v/340)\big)$. For motion $v = \pm 1$ m/s, the difference between the correlation $c_s \star \tilde{c}$ and the autocorrelation $c_s \star c$ is small, with correlation at lag 0 several times greater than any lag more than one chip away. Over one length-T_s symbol frame, the misalignment between c and $\tilde{c}$ is $\frac{1}{2}T_s/|340/v + 1|$ at each end, which is less than 0.2 chips for $|v| \le 1$ m/s. Thus synchronization with the code signal can be robustly and precisely detected even under moderate motion.

B. Normalization

Real environments often have significant nonstationary noise sources. Additionally, the code signal is not exactly orthogonal to itself, having a small amount of off-peak correlation, which appears in the acquisition scores as an unwanted noise-like component proportional to the signal.

To mitigate these effects, acquisition scores are normalized by an estimate of the local noise standard deviation. Normalization also has the benefit that subsequent processing is invariant to the scaling of the received signal, since scaling the input proportionally scales the noise standard deviation.

We derive a rule to normalize the noise level by supposing the baseband signal $b(t)$ is complex-valued white noise with spectral density N_0. This simple noise model is unrealistic, but makes analysis tractable, and we find that the resulting rule works well in practice.

Assuming that the real and imaginary parts are independent and identically distributed, shifts in frequency do not change the distribution of complex white noise, so it is sufficient to consider the distribution of $a^{\text{raw}}(t,\delta)$ for $\delta = 0$, which reduces to a square of filtered white noise $a^{\text{raw}}(t,0) = \left|(\frac{1}{T_s}c_s \star b)(t)\right|^2$. At fixed time t, the cross-correlation $(\frac{1}{T_s}c_s \star b)(t)$ has complex Gaussian distribution with zero mean and variance

$$\mathbf{E}\left[\left|\left(\frac{1}{T_s}c_s \star b\right)(t)\right|^2\right] = \frac{N_0}{2{T_s}^2}\int_0^{T_s} |c(s)|^2\, ds. \tag{32}$$

It follows that $a^{\text{raw}}(t,\delta)$ is exponentially distributed, and (32) is its mean and standard deviation. Therefore, $a^{\text{raw}}(t,\delta)$ divided by its mean has exponential distribution with unit rate,

$$\frac{a^{\text{raw}}(t,\delta)}{\mathbf{E}[a^{\text{raw}}(t,\delta)]} = \frac{2{T_s}^2}{N_0\|c_s\|^2}a^{\text{raw}}(t,\delta) \sim \text{Exp}(1), \tag{33}$$

which no longer depends on the noise level N_0.

We implement (33) by estimating the expected value $\mathbf{E}[a^{\text{raw}}(t,\delta)]$ with a local average and normalize as

$$a(t,\delta) = \frac{a^{\text{raw}}(t,\delta)}{\mu(t)}, \quad \mu = h^{\text{normalize}} * \frac{1}{|\Delta|}\sum_{\delta\in\Delta} a^{\text{raw}}(\cdot,\delta), \tag{34}$$

where $h^{\text{normalize}}$ is a weighted moving average filter (explained further in Section VI-C). Supposing the temporal dependence in a^{raw} is small, $\mu(t)$ approximates $\mathbf{E}[a^{\text{raw}}(t,\delta)]$.

If there is signal and not only noise, each propagation path of the signal appears sparsely in a^{raw} as correlation peaks spaced in time about every T_s and small values in between. If the weighted moving average $\mu(t)$ is sufficiently wide, these peaks have little effect on the normalization.

C. Acquisition Image

Observing the T_s-periodic structure of the signal, a useful representation of acquisition is to maximize over frequency, define a function of time t, and wrap the result into an image

$$a(t) = \max_{\delta\in\Delta} a(t,\delta),$$
$$A(\rho,\varphi) = a(\rho T_s + \varphi), \quad \rho \in \mathbb{Z},\ \varphi \in [0, T_s), \tag{35}$$

where horizontally ρ is the frame index and vertically φ is phase within the frame. The top and bottom image boundaries are arbitrary due to circular phase ambiguity and wrap according to one dimensional order (Fig. 8).

T_s-periodic structures appear as horizontal features in this image. If $a(t)$ is large, then it is likely that $a(t+\rho T_s)$ is also large, appearing in A as a ridge of large values (ρ,φ_0), $\varphi_0 = t \mod T_s$. Thanks to the code signal's finely localized autocorrelation, the peak of this ridge is sharply defined with a main lobe width of T_c ($\approx 333\ \mu$s, or 4 pixels in our implementation where the baseband sample rate is 12 kHz).

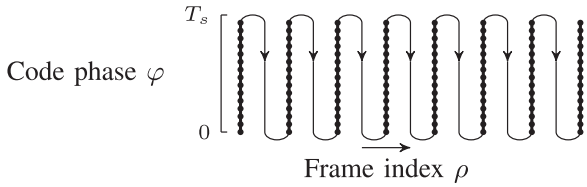

Fig. 8. Wrapping the acquisition image (35).

Each horizontally-oriented ridge in the acquisition image corresponds to a propagation path from the transmitter to the receiver. The acquisition image distinguishes different paths as distinct ridges, enabling reliable decoding in real environments with multiple paths. This multipath capability is the main advantage of DSSS over other modulation schemes.

The acquisition image is denoised by applying a 9-tap box blur filter along each row, taking advantage of the signal's periodic structure. Filtering suppresses noise and bridges small gaps to form connected ridges of large values.

After denoising, image values below a threshold are set to zero. The resulting image is sparse, with surviving nonzeros corresponding to the signal paths as horizontally-oriented connected components.

Fig. 9 shows an example of acquisition. The top plot is a spectrogram of the received signal, in which the transmission is visible as a slightly darker rectangle around time 2.5–4 s and frequency 18.5–20 kHz. The two horizontal structures indicate that two distinct propagation paths are present.

VI. Implementation

Our implementation is in C++, executed as a single thread with most processing done in 32-bit floating-point arithmetic. We use the C++ Eigen linear algebra library [31] for SIMD-optimized dot products and other vector operations. Our receiver operates in a streaming manner, processing an incoming recording stream in blocks of 100 ms at a time.

Our implementation is fast enough on most recent mobile devices and computers to keep up with recording, running several times faster than real time. Computation time to process one 100 ms block, recording at 44.1 kHz:

Cherry Mobile Android One	18.769 ms
Nexus 5X	6.233 ms
Nexus 6P	5.148 ms

A. Downconversion

We require that the sine carrier has an integer number of cycles per symbol frame. Given a desired frequency of 18500 Hz, the closest such carrier frequency is $f_c = 783/T_s \approx 18496$ Hz.

We suppose that the received audio signal $y[n]$ has sample rate F_y. The first step is to demodulate the carrier,

$$b^{\text{raw}}[n] = y[n]\mathrm{e}^{-i2\pi f_c n/F_y}, \tag{36}$$

which has the effect in the frequency domain of shifting down by f_c so that the signal spectrum is in $[0, 1/(2T_c)]$. The raw

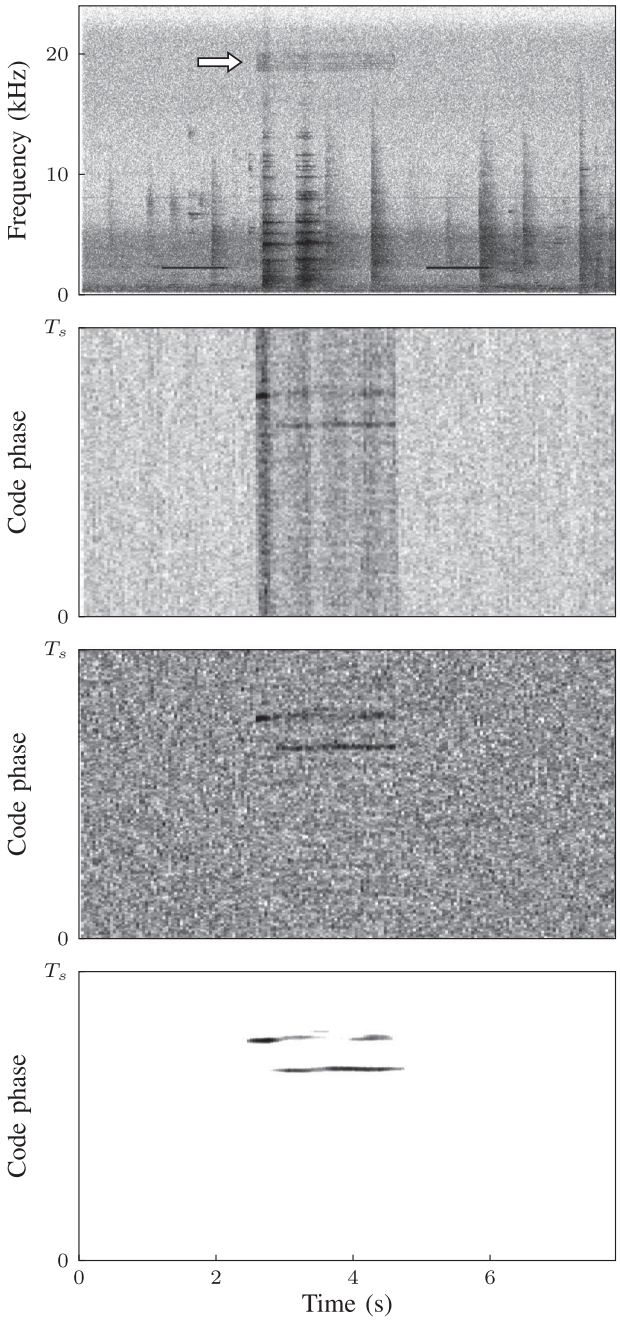

Fig. 9. An example of acquisition. From top to bottom: received signal spectrogram; raw acquisition $a^{\text{raw}}(t,\delta)$; normalized acquisition $a(t,\delta)$; acquisition after blurring and thresholding.

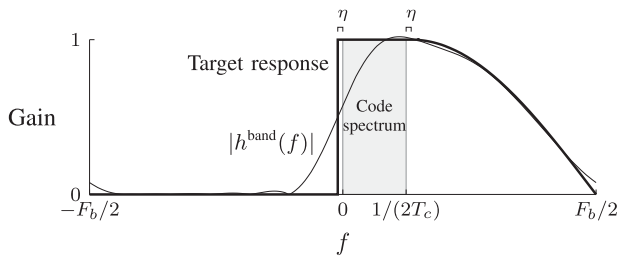

Fig. 10. Response of h^{band} (thin line) and target response (thick line).

baseband signal also has a negative alias of the signal in $[-2f_c - 1/(2T_c), -2f_c]$ (possibly exceeding $-F_y/2$ and wrapping to $[F_y - 2f_c - 1/(2T_c), F_y - 2f_c]$).

g) h^{band} filter: Next, out-of-band noise is filtered from the raw baseband signal. We divide this filtering into two steps.

First, the raw baseband signal is resampled to a lower sample rate $F_b = 12$ kHz to reduce the cost of subsequent processing. We implement resampling as a polyphase FIR decimator with anti-alias filtering to suppress the negative alias of the signal and other out-of-band energy.

Second, the raw baseband signal is filtered to suppress noise in the lower sideband $[-1/(2T_c), 0]$,

$$b = h^{\text{band}} * b^{\text{raw}}, \tag{37}$$

where h^{band} is an 11-tap complex FIR filter with approximately unit gain over $[0, 1/(2T_c)]$ and attenuating negative frequencies $f < 0$ and higher frequencies $f > 1/(2T_c)$ (see Fig. 10). The filter is designed by truncating the impulse response of the following target frequency response with a sine window:

$$H^{\text{band}}(f) = \begin{cases} 0 & -F_b/2 < f < -\eta, \\ 1 & -\eta \le f \le f^{\text{knot}}, \\ \cos\left(\frac{\pi}{2}\frac{f - f^{\text{knot}}}{F_b/2 - f^{\text{knot}}}\right) & f^{\text{knot}} < f < F_b/2, \end{cases} \tag{38}$$

where $f^{\text{knot}} = 1/(2T_c) + \eta$ and η is a small positive parameter.

Algorithm 1: Downconversion

Input: Received signal (y_n) with sample rate $F_y \ge 44.1$ kHz

Output: Baseband signal (b_n) with sample rate F_b

1: $b^{\text{raw}}[n] \leftarrow y[n]\mathrm{e}^{i2\pi f_c n/F_s}$
2: Resample b^{raw} from F_y to F_b
3: $b[n] \leftarrow \sum_k h^{\text{band}}[k] b^{\text{raw}}[n-k]$

B. Synchronization

Having the baseband signal, the next step is to identify the time and Doppler frequency offsets of frames within the signal by computing the modulated correlation

$$a^{\text{raw}}(t,\delta) = \left| \frac{1}{T_s} \int_0^{T_s} \mathrm{e}^{i2\pi\delta s} c(s) b(s+t)\, ds \right|^2. \tag{39}$$

A direct discretization of (39) would be

$$a^{\text{raw}}[n,\delta] = \left| \frac{1}{T_s} \sum_{m=0}^{M_s-1} \mathrm{e}^{i2\pi\delta m/F_b} c(m/F_b) b[n+m] \right|^2, \tag{40}$$

where $M_s = F_b T_s$ is the number of baseband samples per symbol. However, (40) would be prohibitively expensive, costing $2M_s|\Delta|$ complex multiplies for each time offset n.

We approximate (39) to make decoding tractable on device. The interval $[0, T_s]$ is partitioned into N_p panels of length $T_p = T_s/N_p$, and the factor $\mathrm{e}^{i2\pi\delta t}$ is approximated as having constant

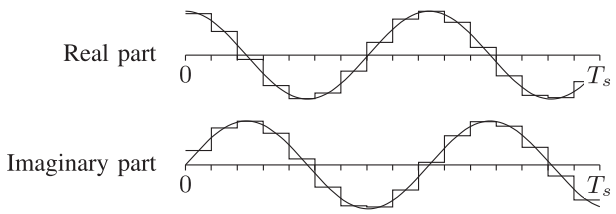

Fig. 11. Piecewise approximation of (42). The modulation factor $\exp(i2\pi\delta t)$ is approximated as piecewise constant with $N_p = 16$ pieces.

phase over each panel (Fig. 11),

$$\exp(i2\pi\delta t) \approx \exp\left(i2\pi\delta T_p\left(k+\frac{1}{2}\right)\right),\quad k=\lfloor t/T_p\rfloor. \quad (41)$$

This approximation is reasonable if N_p is large compared to δ so that the phase changes little over each panel.

Substituting the piecewise constant modulation into (39), we approximate modulated correlation $a^{\text{raw}}(t,\delta)$ as

$$\left|\frac{1}{T_s}\sum_{k=0}^{N_p-1} \mathrm{e}^{i2\pi\delta T_p(k+\frac{1}{2})}\int_{kT_p}^{(k+1)T_p} c(s)b(s+t)\,ds\right|^2. \quad (42)$$

This may be viewed as sum-and-dump decimation of the despread baseband signal $c(s)b(s+t)$ before the modulation, and it factors such that the integrals no longer depend on the hypothesized Doppler shift δ.

This approximation accurately maintains the correlation peak. Suppose that the despread baseband signal is $c(s)b(s+t) = \mathrm{e}^{-i2\pi\delta s}$, corresponding to $b(s+t) \approx \mathrm{e}^{-i2\pi\delta s}c(s)$. Then

$$\frac{1}{T_s}\int_0^{T_s} \mathrm{e}^{i2\pi\delta s}c(s)b(s+t)\,ds = \frac{1}{T_s}\int_0^{T_s} ds = 1, \quad (43)$$

and approximation (42) computes

$$\frac{1}{T_s}\sum_{k=0}^{N_p-1} \mathrm{e}^{i2\pi\delta T_p(k+\frac{1}{2})}\int_{kT_p}^{(k+1)T_p} c(s)b(s+t)\,ds = \frac{\sin(\pi|\delta|T_p)}{\pi|\delta|T_p}. \quad (44)$$

Using $\sin(x) \geq x - \frac{1}{6}x^3$ for $x \geq 0$, the difference between (43) and (44) is bounded as

$$1 - \frac{\sin(\pi|\delta|T_p)}{\pi|\delta|T_p} \leq \frac{1}{6}(\pi\delta T_s/N_p)^2. \quad (45)$$

In our implementation, the number of panels is $N_p = 16$ and the number of Doppler cycles per frame for up to 1 m/s of motion is bounded as $|\delta|T_s \leq 2.4$, for which

$$\frac{1}{6}(\pi\delta T_s/N_p)^2 \leq \frac{1}{6}(\pi 2.4/16)^2 \approx 0.04. \quad (46)$$

Therefore, less than 4% of the correlation peak is lost. This approximation reduces the cost and is sufficiently accurate to make brute-force search feasible.

We discretize (42) with $M_p = M_s/N_p$ samples per panel and compute code correlations $p[k]$ over the panels,

$$p[k] = \sum_{m=0}^{M_p-1} c\left(\frac{kM_p+m}{F_b}\right) b[kM_p+m+n]. \quad (47)$$

Implementation amounts to N_p correlations over M_s/N_p-sample panels and a N_p-term sum for each δ. The cost is $(M_s + N_p|\Delta|)$ complex multiplies for each n vs. $2M_s|\Delta|$. This allows to obtain a significant reduction in computational load. We combine panels to compute raw acquisition scores

$$a^{\text{raw}}[n,\delta] = \left|\frac{1}{T_s}\sum_{k=0}^{N_p-1} \mathrm{e}^{i2\pi\delta T_p(k+\frac{1}{2})}p[k]\right|^2. \quad (48)$$

The $+1/2$ in the exponent may be omitted without changing the complex magnitude.

Algorithm 2: Synchronization

Input: Baseband signal (b_n) with sample rate F_b
Output: Raw acquisition scores $a^{\text{raw}}[n,\delta]$
1: Precompute $c(n/F_b)$ for $n = 0,\ldots,M_s-1$
2: **for** $n = 0, 1, \ldots$ **do**
3: Compute $p[k]$ with (47)
4: $a^{\text{raw}}[n,\delta] \leftarrow |\sum_{k=0}^{N_p-1}\exp(i2\pi\delta T_p k)p[k]|^2$ for $\delta \in \Delta$
5: **end for**

A minor adjustment is needed to account for that M_s is not evenly divisible by N_p. All but the last panel have $M_p = \lceil M_s/N_p\rceil$ samples per panel. The last panel has the remaining $M_p^{\text{last}} := M_s - (N_p-1)M_p$ samples, and in (47), the summation is over $m = 0,\ldots,M_p^{\text{last}}-1$ for $k = N_p - 1$.

C. Normalization

To normalize out variations in noise level, raw acquisition scores are scaled in a temporally-adaptive manner,

$$a[n,\delta] = \frac{a^{\text{raw}}[n,\delta]}{(h^{\text{normalize}} * s)_n},\quad s_n = \sum_{\delta\in\Delta} a^{\text{raw}}[n,\delta]. \quad (49)$$

For a cheap filter with roughly Gaussian-like impulse response, $h^{\text{normalize}}$ is the recursive 2-pole filter with Z-transform

$$H^{\text{normalize}}(z) = z^d\left(\frac{1-p}{1-pz^{-1}}\right)^2,\quad d = \left\lceil\frac{2p}{1-p}\right\rceil, \quad (50)$$

where $0 < p < 1$ is a parameter. This filter is attractive since it can be efficiently implemented with two multiplies per sample.

The z^d factor is an advance so that the center of mass of the impulse response is at lag 0. Its impulse response is the probability mass function of a negative binomial (Pascal) distribution with parameters $r = 2$ and p,

$$h_n^{\text{normalize}} = (n+1)(1-p)^2p^n \quad n \geq 0. \quad (51)$$

Fig. 12 shows, for sake of visualizing its discrete nature, the response for $p = 0.83$, where the center of mass is $2p/(1-p) \approx 9.76$ samples. We use $p = 0.990805$, where the center of mass is about 215.51 baseband samples.

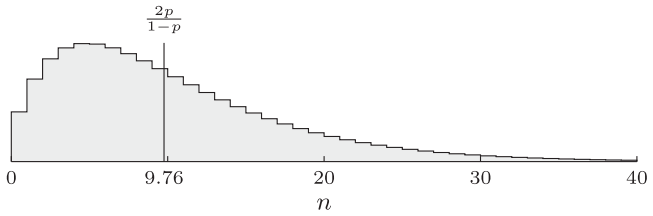

Fig. 12. Impulse response of $h^{\text{normalize}}$ with $p = 0.83$. Its center of mass is $\frac{2p}{1-p} \approx 9.76$ samples.

Algorithm 3: Normalization

Input: Raw acquisition scores $a^{\text{raw}}[n, \delta]$, filter parameter p
Output: Normalized acquisition scores $a[n, \delta]$

$d \leftarrow \left\lceil \frac{2p}{1-p} \right\rceil$
$\mu^{\text{init}} \leftarrow \text{mean}\{s_0, \ldots, s_{d-1}\}$
$z_0 \leftarrow \mu^{\text{init}}, z_1 \leftarrow \mu^{\text{init}}$
for $n = 0, 1, \ldots$ **do**
  $s_n \leftarrow \sum_{\delta \in \Delta} a^{\text{raw}}[n, \delta]$
  $z_0 \leftarrow z_0 + (1-p)(s_n - z_0)$
  $z_1 \leftarrow z_1 + (1-p)(z_0 - z_1)$
  $a[n-d, \delta] \leftarrow a^{\text{raw}}[n-d, \delta]/z_1$ for $\delta \in \Delta$
end for

We estimate the frequency shift δ with a simple temporally noncoherent selection as the frequency with the largest score, and define a summary score at sample n by

$$a[n] := \max_{\delta \in \Delta} a[n, \delta_n]. \tag{52}$$

VII. Tokens

The preceding sections described how to encode a sequence of symbols $k[m]$. We now turn to the higher level goal of transmitting a *token*, a small fixed-length amount of data.

h) Protocol: First, the beginning of the token is marked with a "spacer" having symbol value $k[0] = K$, which is reserved for this use. The spacer is followed by the data payload converted base-K to symbol values $k[1], \ldots, k[L] \in \{0, \ldots, K-1\}$. Last, as a limited form of error detection coding, a parity symbol is appended $k[L+1]$ whose value in $\{0, \ldots, K-1\}$ is such that the sum of all token symbols modulo K is zero. We call the encoding of this sequence of symbols $k[0], \ldots, k[L+1]$ one "token repetition" of the signal.

SSB highpass filtering (Section II-A) is implemented by transforming one such token repetition and setting FFT coefficients below f_c to zero. The SSB-filtered token repetition is then transmitted repeatedly. When starting and stopping transmission, we taper the ends with 5 ms linear ramps to limit spectral splatter. Relative to the signal, residual frequency components below f_c are 58 dB down at 18.25 kHz and at least 82 dB down for frequencies below 17 kHz, below the threshold of hearing for most humans.

The number of data symbols L must be known to the decoder. We find that under typical circumstances, broadcasting the token three times is enough for successful transmission.

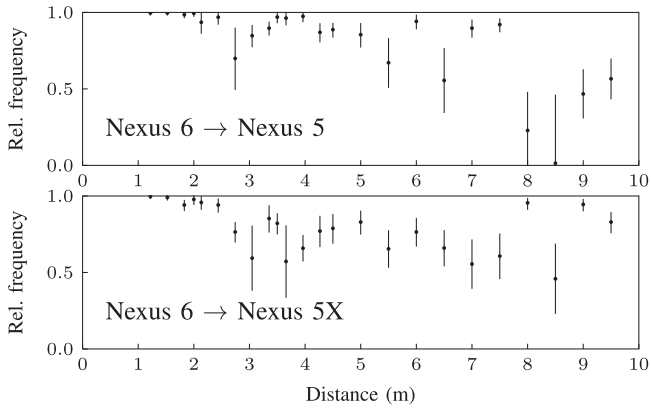

Fig. 13. Reliability vs. distance from real recordings in an office environment. A transmission is considered successful if the decoder produces the correct token. Error bars show two-sided 90% confidence intervals. Top: Nexus 6 transmitting, Nexus 5 receiving. Bottom: Nexus 6 transmitting, Nexus 5X receiving.

Repeating the token signal allows receivers that are slow to initialize or duty-cycled to begin receiving partway into transmission and still decode the data, provided that at least one token period is received. The spacer symbol is used to resolve circular ambiguity.

Token repetition also serves as very simple form of error correction, giving multiple chances to observe each symbol.

i) Symbol selection with token repetitions: The symbol scores (12) s_k are tabulated per symbol value at each decoding time. Scores at positions modulo $(L+2)$ are added to aggregate over token repetitions. The (i, j)th entry in the resulting table represents the confidence that the jth symbol in the token has symbol value i. Scores are thresholded and normalized such that each column has unit sum. The candidate having the largest summed score and valid spacer and parity is selected as the most likely decoding.

VIII. Results

A. Real-World Experiments

We test the protocol with real-world recordings at various distances, using a Nexus 6 as the transmitter and a Nexus 5 or Nexus 5X as receiver. The phones are placed flat on movable filing cabinets in a quiet office environment with ends facing each other. In each trial, a random 64-bit token is encoded, and three token repetitions are transmitted. To avoid boundary effects, encoded signals are padded with 0.25 s of silence on each end. A transmission is considered successful if the decoder produced the correct token.

Fig. 13 shows the observed relative frequency of successful transmissions vs. distance. Each point in the plot represents at least 120 transmission trials. At distance 2 m or closer, 94% or more transmissions succeeded. Transmission was often successful at further distances, even past 9 m. Reliability is similar with either the Nexus 5 or Nexus 5X as the receiver.

Although the devices have line of sight, sound still reflects from the room walls and other surrounding objects and produces nontrivial multipath effects. We believe transmission reliability to be sensitive to the environment, especially at further dis-

tances, since small changes to environment geometry can substantially change how different propagation paths interfere with one another. This might explain the non-monotonic decrease in reliability with distance in Fig. 13, although the overall trend decreases as expected. Sound absorption properties of materials in the room have an effect as well, for example, carpet absorbs high frequency sound much more than flat rigid surfaces do.

This experiment was done in a quiet office environment. In noisier settings, reliability is worse, depending on the background noise in the signal band. We observe in moderately-noisy open office spaces, transmission between various recent smartphone devices still usually succeeds at distances up to 2 m. We note also that our protocol is used successfully in practice as part of the Google Nearby platform and other products. The next sections investigate how reliability varies with different kinds of distortion.

B. Noise Robustness

We use simulated signals to test the robustness of the protocol with respect to background interference. The signal is a random 64-bit token. Noise is added to the encoded signal. We then attempt to decode the noisy signal using our C++ receiver implementation. The transmission is a success if the top candidate is the correct decoding.

The strength of the signal relative to noise is quantified by the in-band signal-to-noise ratio (SNR),

$$\text{in-band SNR} := 10 \log_{10} \frac{\int |S(f)|^2 \, df}{\int |N(f)|^2 \, df} \tag{53}$$

where $S(f)$ is the clean signal spectrum, $N(f)$ is the noise spectrum, and the domain of integration is the signal band $[f_c, f_c + 1/(2T_c)]$ (approximately 18.5–20 kHz).

We perform this test with additive white Gaussian noise (AWGN) for various SNR levels. Fig. 14 shows results, scaled as probability of successful transmission vs. in-band SNR and as bit error rate vs. E_b/N_0. We compare also how repetitions of the token signal affect reliability and show results with broadcasting the token once, twice, or three times. Each point is estimated from 10^6 simulated transmissions.

The protocol shows good overall robustness to noise, succeeding at low SNR levels thanks to the DSSS modulation. More repeats of the token improves the probability of successful transmission, but is less energy efficient.

For sound, AWGN is a poor model of background interference. Many natural sounds are short explosions that are not sustained over time, such as clicks and bumps as objects come into contact with one another. This motivates us to examine robustness to impulsive noise. Comparing with the AWGN noise tests in Fig. 14, these impulsive tests illustrate how reliability varies with noise distribution.

In Fig. 15, we repeat the experiment with impulsive noise for comparison, broadcasting the token three times. The impulsive

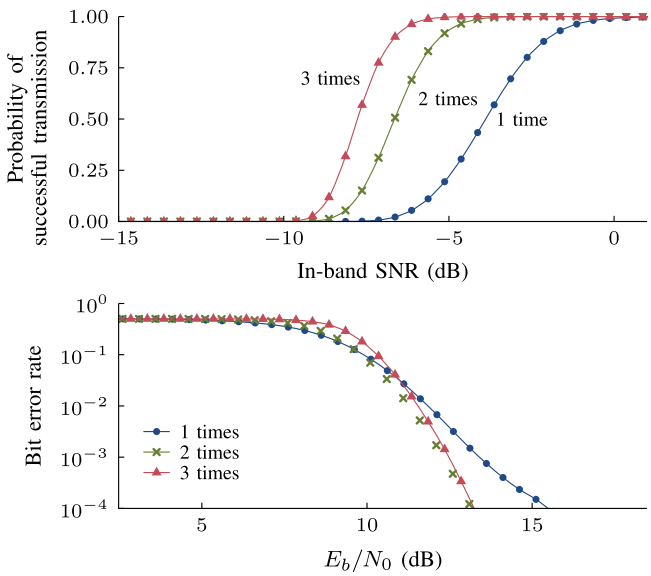

Fig. 14. Robustness to AWGN in simulated transmissions, broadcasting the token 1, 2, or 3 times. Top: probability of successful transmission vs. in-band SNR. A transmission is considered successful if the decoder produces the correct token. Bottom: bit error rate vs. E_b/N_0.

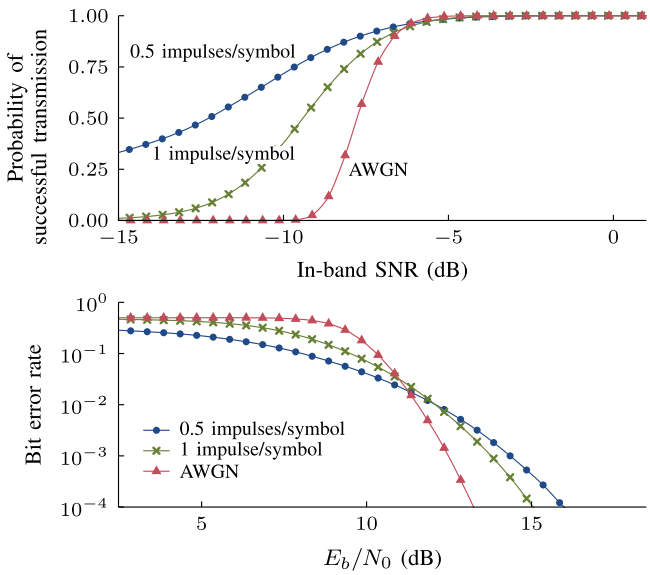

Fig. 15. Robustness to different types of noise in simulated transmissions, broadcasting the token 3 times. Top: probability of successful transmission vs. in-band SNR. A transmission is considered successful if the decoder produces the correct token. Bottom: bit error rate vs. E_b/N_0.

noise has i.i.d. samples

$$\begin{cases} +h & \text{with probability } p/2, \\ -h & \text{with probability } p/2, \\ 0 & \text{otherwise,} \end{cases} \tag{54}$$

where h and p are parameters determining the height of the impulses and the rate at which impulses occur. We set $p = 1/(T_s F_y)$ for one expected impulse per symbol and vary h to

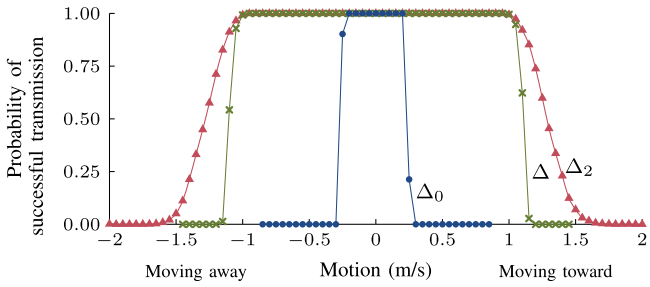

Fig. 16. Probability of successful transmission vs. simulated motion with 0 dB in-band SNR with different Doppler shift search sets Δ_0, Δ, and Δ_2. A transmission is considered successful if the decoder produces the correct token.

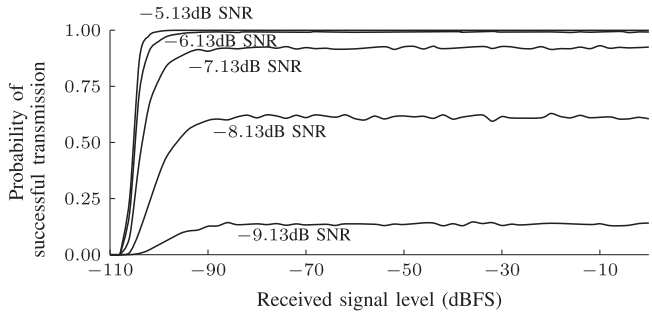

Fig. 17. Probability of successful transmission with simulated 16-bit quantization vs. input level, with AWGN at several in-band SNR levels.

obtain different SNR levels. We also show impulsive noise with p set for 0.5 expected impulses per symbol.

Impulsive noise is tolerable since the token is transmitted 3 times. When a symbol is obscured by an impulse in one repetition, the decoder has two other chances.

C. Motion Robustness

Next we test the robustness of the protocol with respect to motion. The operable range of motion depends on Δ, the set of Doppler shifts δ considered in acquisition and demodulation. We compare three choices for the Doppler shift search set:

1) $\Delta = \{0, \pm\frac{0.3}{T_s}, \pm\frac{0.6}{T_s}, \ldots, \pm\frac{2.4}{T_s}\}$, our default search set of Doppler shifts up to ± 56.7 Hz, which is large enough to include motion up to ± 1 m/s.
2) $\Delta_2 = \{0, \pm\frac{0.3}{T_s}, \pm\frac{0.6}{T_s}, \ldots, \pm\frac{4.8}{T_s}\}$, a search set with twice the range, with Doppler shifts up to ± 113.4 Hz, including motion up to ± 2 m/s.
3) $\Delta_0 = \{0\}$, a minimal search set where motion is not considered. Raw acquisition scores are then simply correlation with the code, $a^{\text{raw}}(t, 0) = (c_s \star b)(t)$.

Fig. 16 shows the results following the testing procedure as in the previous section, but resampling the encoded signal by factor $(1 + v/(340\,\text{m/s}))$ to simulate Doppler shift for linear motion with velocity v. After resampling, AWGN is added with 0 dB in-band SNR. Each point is estimated from 10^4 simulated transmissions.

With our default search set Δ, transmission nearly always succeeds for $|v| \leq 1$ m/s, then drops sharply to nearly always fail for $|v| \geq 1.2$ m/s. This operating range agrees with the design intention of supporting motion up to ± 1 m/s.

With Δ_2, greater motion is tolerated, but reliability is limited for $|v| > 1$ m/s, probably by our approximate model of motion. We model the effect of motion as a frequency shift, when more accurately it is a scaling, and time dilation of the code is ignored.

With Δ_0, Doppler shift is not considered. Nevertheless, some small motion $|v| \leq 0.2$ m/s is tolerated.

D. Quantization Robustness

In this test, we vary the level of the simulated received signal and quantize it with a fixed 16-bit quantizer. Measuring level as decibels relative to full scale, that is, with 0 dBFS meaning maximum amplitude of 1.0 and a quantization of 2^{-15}, we assess the effect of quantization down to very low signal levels, at several in-band SNR levels.

Fig. 17 shows how reliability varies with signal level. Reliability is nearly constant above -90.3 dBFS, showing that the protocol is basically immune to quantization. Below $20\log_{10}(2^{-15}) \approx -90.3$ dBFS, the signal amplitude is less than one quantization step. Successful transmission continues to be possible slightly below this point, since adding noise followed by quantization has a dithering effect.

IX. Conclusion

We have presented a protocol for near-ultrasonic sound communication using direct-sequence spread-spectrum (DSSS) modulation. Rather than conventional lock-and-track acquisition, a brute force correlation is computed over a grid of combinations of Doppler shifts and time offsets to synchronize the system quickly. The protocol is robust to multiple propagation paths, background noise, and motion, and works reliably at 2 m distance in real indoor environments.

Though we have designed a system that is practical, reliable, and efficient, there remain unexplored alternatives that might perform better. Of all aspects in the design of this system, the pedestal added to the data waves is perhaps the biggest trade-off: it enables robust and efficient synchronization, but spends more than half the energy without contributing to distinguishing symbol values.

Acknowledgment

Many of our colleagues have helped us with the ideas presented here. We specifically thank Andrew Bunner, Brian Duff, Ami Patel, Arunesh Mishra, and Jeremy Thorpe for their support and encouragement.

References

[1] O. Riva and J. Kangasharju, "Challenges and lessons in developing middleware on smart phones," *Computer*, vol. 41, no. 10, pp. 23–31, Oct. 2008.
[2] K. Ashihara, K. Kurakata, T. Mizunami, and K. Matsushita, "Hearing threshold for pure tones above 20 kHz," *Acoust. Sci. Technol.*, vol. 27, no. 1, pp. 12–19, 2006.
[3] G. Ballou, *Handbook for Sound Engineers*, 5th ed. New York, NY, USA: Taylor & Francis, 2015.
[4] J. P. Cowan, *Handbook of Environmental Acoustics*. Hoboken, NJ, USA: Wiley, 1993.

[5] C. Lopes and P. Aguiar, "Aerial acoustic communications," in *Proc. IEEE Workshop Appl. Signal Process. Audio Acoust.*, 2001, pp. 219–222.
[6] M. Hazas and A. Ward, "A novel broadband ultrasonic location system," in *UbiComp 2002: Ubiquitous Computing* (Lecture Notes in Computer Science, vol. 2498), G. Borriello and L. E. Holmquist, Eds. Berlin, Germany: Springer-Verlag, 2002, pp. 264–280. [Online]. Available: http://dx.doi.org/10.1007/3-540-45809-3_21
[7] M. Alloulah and M. Hazas, "An efficient CDMA core for indoor acoustic position sensing," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat.*, Sep. 2010, pp. 1–5.
[8] N. M. Vallidis, "WHISPER: A spread spectrum approach to occlusion in acoustic tracking," Ph.D. dissertation, Dept. Comput. Sci., Univ. North Carolina at Chapel Hill, NC, USA, 2002.
[9] L. Galluccio, T. Melodia, S. Palazzo, and G. E. Santagati, "Challenges and implications of using ultrasonic communications in intra-body area networks," in *Proc. 9th Annu. Conf. Wireless On-demand Netw. Syst. Serv.*, Jan. 2012, pp. 182–189.
[10] G. E. Santagati, T. Melodia, L. Galluccio, and S. Palazzo, "Ultrasonic networking for e-health applications," *IEEE Wireless Commun.*, vol. 20, no. 4, pp. 74–81, Aug. 2013.
[11] G. E. Santagati and T. Melodia, "U-wear: Software-defined ultrasonic networking for wearable devices," in *Proc. 13th ACM Annu. Int. Conf. Mobile Syst. Appl. Serv.*, 2015, pp. 241–256.
[12] G. E. Santagati, T. Melodia, L. Galluccio, and S. Palazzo, "Medium access control and rate adaptation for ultrasonic intrabody sensor networks," *IEEE/ACM Trans. Netw.*, vol. 23, no. 4, pp. 1121–1134, Aug. 2015.
[13] M. Hanspach and M. Goetz, "On covert acoustical mesh networks in air," *CoRR*, vol. abs/1406.1213, 2014. [Online]. Available: http://arxiv.org/abs/1406.1213
[14] M. Hanspach and M. Goetz, "Recent developments in covert acoustical communications," in *Sicherheit*, 2014, pp. 243–254.
[15] L. Deshotels, "Inaudible sound as a covert channel in mobile devices," in *Proc. 8th USENIX Workshop Offensive Technol.*, 2014.
[16] Q. Do, B. Martini, and K.-K. R. Choo, "Exfiltrating data from Android devices," *Comput. Security*, vol. 48, pp. 74–91, 2015.
[17] B. Carrara, "Air-gap covert channels," Ph.D. dissertation, School Elect. Eng. Comput. Sci., Univ. Ottawa, Ottawa, ON, Canada, 2016.
[18] H. Matsuoka, Y. Nakashima, and T. Yoshimura, "Acoustic communication system using mobile terminal microphones," *NTT DoCoMo Tech. J.*, vol. 8, no. 2, pp. 2–12, 2006.
[19] H. S. Yun, K. Cho, and N. S. Kim, "Acoustic data transmission based on modulated complex lapped transform," *IEEE Signal Process. Lett.*, vol. 17, no. 1, pp. 67–70, Jan. 2010.
[20] R. Nandakumar, K. K. Chintalapudi, V. Padmanabhan, and R. Venkatesan, "Dhwani: Secure peer-to-peer acoustic NFC," in *Proc. ACM SIGCOMM Comput. Commun. Rev.*, 2013, vol. 43, no. 4, pp. 63–74.
[21] H. Lee, T. H. Kim, J. W. Choi, and S. Choi, "Chirp signal-based aerial acoustic communication for smart devices," in *Proc. IEEE Conf. Comput. Commun.*, 2015, pp. 2407–2415.
[22] R. C. Dixon, *Spread Spectrum Techniques*. Piscataway, NJ, USA: IEEE PRESS, 1976.
[23] D. Kirovski and H. Malvar, "Robust covert communication over a public audio channel using spread spectrum," in *Proc. 4th Int. Workshop Inf. Hiding*, 2001, pp. 354–368.
[24] N. Lazic and P. Aarabi, "Communication over an acoustic channel using data hiding techniques," *IEEE Trans. Multimedia*, vol. 8, no. 8, pp. 918–924, Oct. 2006.
[25] X. Kang, R. Yang, and J. Huang, "Geometric invariant audio watermarking based on an LCM feature," *IEEE Trans. Multimedia*, vol. 13, no. 2, pp. 181–190, Apr. 2011.
[26] E. Sozer, J. Proakis, M. Stojanovic, J. Rice, A. Benson, and M. Hatch, "Direct sequence spread spectrum based modem for under water acoustic communication and channel measurements," in *Proc. OCEANS'99 MTS/IEEE. Riding Crest21st Century,* 1999, pp. 228–233.
[27] M. K. Simon, J. K. Omura, R. A. Scholtz, and B. K. Levitt, *Spread Spectrum Communications Handbook*. New York, NY, USA: McGraw-Hill, 1994.
[28] R. Gold, "Optimal binary sequences for spread spectrum multiplexing," *IEEE Trans. Inf. Theory*, vol. IT-13, no. 4, pp. 619–621, Oct. 1967.
[29] H. J. Ralston, "Energy-speed relation and optimal speed during level walking," *Internationale Zeitschrift für Angewandte Physiologie Einschliesslich Arbeitsphysiologie*, vol. 17, no. 4, pp. 277–283, 1958.
[30] R. G. Winch, *Telecommunication Transmission Systems*. New York, NY, USA: McGraw-Hill, 1998.
[31] G. Guennebaud *et al.*, "Eigen: A C++ linear algebra library," libraries for scientific computing, École Polytechnique, 2013. [Online]. Available: http://eigen.tuxfamily.org

Pascal Getreuer received the B.S. and M.S. degrees in applied mathematics from the University of Colorado, Boulder, CO, USA, and the Ph.D. degree in applied mathematics from the University of California, Los Angeles, CA, USA. He is currently at Google Research on the Computational Imaging team. His research interests include image and audio processing, inverse problems, and scientific computing.

Chet Gnegy received the B.S. degree in electrical engineering from the University of Pittsburgh, Pittsburgh, PA, USA, and the M.S. degree in music technology from Stanford's Center for Computer Research in Music and Acoustics, Stanford University, Stanford, CA, USA. He is a musician and electrical engineer also he is currently at Google Research on the Sound Understanding team, where among other things, he is developing multimicrophone technology.

Richard F. Lyon (LF'17) grew up in El Paso, TX, USA, and received the B.S. degree from Caltech, Pasadena, CA, USA, and the M.S. degree from Stanford University, Stanford, CA, USA, in electrical engineering. He has worked in research in Silicon Valley for more than 40 years, in various aspects of signal processing, VLSI design, machine perception, electronic photography, sound, and hearing. He is currently a Principal Research Scientist at Google, where he leads the Sound Understanding team, and previously led the team developing camera systems for Street View. Cambridge University Press recently published his book *Human and Machine Hearing: Extracting Meaning from Sound.*

Rif A. Saurous received the S.B. degree in mathematics with computer science and the M.S. and Ph.D. degrees in operations research from Massachusetts Institute of Technology, Cambridge, MA, USA. He conducts research in the fields of machine learning and audio and signal processing.