Tracko: Ad-hoc Mobile 3D Tracking Using Bluetooth Low Energy and Inaudible Signals for Cross-Device Interaction

Haojian Jin[1], **Christian Holz**[1,2]
[1]Yahoo Labs, Sunnyvale, CA
{haojian, christianh} @ yahoo-inc.com

Kasper Hornbæk[2]
[2]University of Copenhagen, Denmark
kash @ di.ku.dk

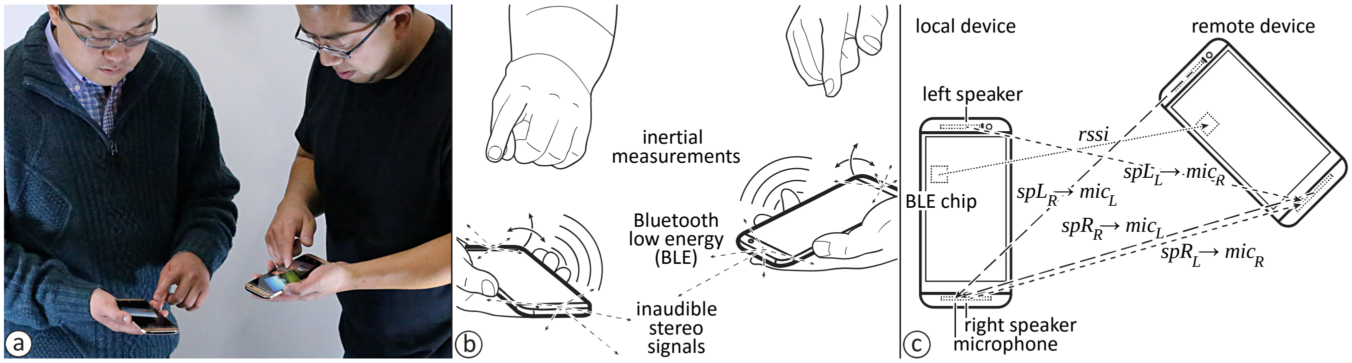
a
b
inertial measurements
Bluetooth low energy (BLE)
inaudible stereo signals
c
local device
remote device
left speaker
rssi
BLE chip
$spL_L \rightarrow mic_R$
$spL_R \rightarrow mic_L$
$spR_R \rightarrow mic_L$
$spR_L \rightarrow mic_R$
right speaker
microphone

Figure 1: *Tracko* provides mobile devices with a *spatial awareness* of surrounding devices in 3D, allowing users to interact across devices in the very space they act in. Tracko readily runs on commodity devices with no added components and requires no calibration or external infrastructure. (a) Here, Tracko enables file sharing across devices simply by swiping items towards the recipient. (b) Tracko detects the *presence* of other devices using Bluetooth low energy, (c) determines their *distance* and *3D direction* from the difference in arrival times of exchanged inaudible stereo signals, and detects quick interactions using inertial sensors.

ABSTRACT

While current mobile devices detect the presence of surrounding devices, they lack a truly *spatial* awareness to bring them into the user's natural 3D space. We present *Tracko*, a 3D tracking system between two or more commodity devices without added components or device synchronization. Tracko achieves this by fusing three signal types. 1) Tracko infers the *presence* of and rough distance to other devices from the strength of Bluetooth low energy signals. 2) Tracko exchanges a series of inaudible stereo sounds and derives a set of accurate distances between devices from the difference in their arrival times. A Kalman filter integrates both signal cues to place collocated devices in a shared *3D space*, combining the robustness of Bluetooth with the accuracy of audio signals for relative 3D tracking. 3) Tracko incorporates inertial sensors to refine 3D estimates and support quick interactions. Tracko robustly tracks devices in 3D with a mean error of 6.5 cm within 0.5 m and a 15.3 cm error within 1 m, which validates Tracko's suitability for cross-device interactions.

ACM Classification Keywords
H.5.2 [Information interfaces and presentation]: User Interfaces. Input devices & strategies.

INTRODUCTION

Interaction with applications increasingly spans multiple devices. Mobile scenarios often involve cross-device file transfers [6], device-specific actions (e.g., Android Wear notifications), or seamless continuation of interaction across devices (e.g., iOS 8 Continuity). In multi-user situations, games and collaborative applications integrate multiple devices into the same workflow and require a notion of collocated users [15] or of space to detect proximate devices [26].

Today's mobile devices typically detect the *presence* of surrounding devices and display them in a list, such as to support file sharing. This conflicts with users' natural way of interacting, which is *spatial* in the physical world. However, current commodity devices do not share this *spatial awareness* of other devices in 3D and require tracking systems that are installed in the environment (e.g., Optitrack).

In this paper, we present Tracko, a relative 3D tracking system for commodity mobile devices. Tracko requires no external infrastructure and readily provides ad-hoc, spatial device tracking without the need for calibration.

TRACKO: 3D TRACKING WITH BLE, SOUND, AND IMUs

Figure 1a shows a file sharing application that uses Tracko to establish a *spatial relation* between two mobile devices. Instead of selecting the recipient from a list, the user on the right simply swipes the image towards the other user's device. Note that all tracking happens ad-hoc; both devices solely need to run Tracko, which *continuously* updates its 3D model of surrounding devices' 3D positions. Tracko runs at interactive rates on all devices with stereo speakers, which spans a wide variety of off-the-shelf phones and tablets.

Tracko combines three types of signals: 1) *Bluetooth low energy* (BLE) serves as a robust tracking component to detect surrounding devices and derive coarse distance to them from BLE signal strengths. 2) *Inaudible acoustic signals* are the key signal for Tracko to calculate 3D positions. Each device repeatedly emits stereo signals that encode the device identifier. Upon receiving such a signal, each device broadcasts arrival timestamps of these signals to surrounding devices. From the differences in arrival times on all devices, Tracko computes the distances between the speakers and microphones across devices, estimates 3D offsets through a residual model, and broadcasts 3D offsets to all other devices. In the case acoustic signals become unavailable, such as when a user accidentally covers the microphone, Tracko falls back to BLE-based distance tracking. 3) *Inertial sensing* detects device motions and rotations at 100 Hz, through which Tracko adjusts its spatial model of previously computed 3D offsets and responds quickly to user input.

CONTRIBUTIONS

Our main contribution is a tracking system that provides ad-hoc relative 3D positions of close-by devices at interactive rates on commodity devices without additional instrumentation. Tracko requires no preconfigured components in the infrastructure and supports moving and resting devices.

Our specific contributions are as follows:

– Robust relative 3D tracking on commodity devices with stereo speakers, a microphone and Bluetooth low energy with an average accuracy of 6.5cm (within 0.5 m) and 15.3 cm (within 1 m) at 3 Hz. Tracko thereby computes distances between speakers and microphones across devices with an accuracy of 2 cm. No cross-device clock synchronization is necessary for Tracko to perform 3D tracking.

– A scalable design of *inaudible* acoustic signal encodings for use with multiple simultaneous devices that is robust to audio interference and concurrent sound signals.

– An on-the-fly calibration of BLE signal strengths to accurate distances, using the measurements Tracko obtains from exchanged sound signals, resulting in BLE-based distance estimations with an average error of 20 cm at 5 Hz, a substantial improvement over current BLE-based approaches.

Finally, Tracko is the first mobile-to-mobile device tracking system that runs at interactive rates, processes all signals and user input in real-time, and dynamically tracks devices' 3D locations and orientations. Our novel *residual model* thereby estimates 3D locations and maintains a low error rate and high accuracy within a typical office environment.

Tracko creates a fundament for user-facing applications similar to other 3D tracking systems. In addition to the applications we show in this paper, Tracko supports the various cross-device interactions found in the literature [6,10,15,].

RELATED WORK

Audio signal design, detection and scalability: Detecting signals encoded in audio is a standard challenge in signal processing. BeepBeep uses audible linear chirps [19]. Qiu et al. sequentially exchanged two modulated pseudo random code signals to avoid overlap [20]. SwordFight instead encodes signals using audible m-sequences, which supports detection during device motions [26]. However, none of these systems explored inaudible signals on commodity devices, which is one of Tracko's contributions to make tracking unobtrusive. While Lazik and Rowe examined the use of inaudible ultrasonic chirps in a *stationary* tracking system, they used standalone audio speakers to emit signals [11].

Bluetooth tracking and signal robustness: BLE is now commonly used for communication across mobile and wearable devices, because it is a robust and reliable signal. The use of BLE for distance prediction is mostly limited to mappings from signal strengths (RSSI) to coarse distances (e.g., close, medium, far, out of range). Since environments constantly change, a one-time calibration is invalid for changing environments. Blumrosen et al. explored the use of a Bluetooth signal after careful calibration and tracked devices with an error of 4.7 cm, using multiple special and stationary BLE beacons [4]. Marquardt et al. placed QSRCT radio nodes in the environment and attached radio modules to all mobile devices to infer device layouts, refined by a depth camera [15].

Video and audio-based mobile tracking: A number of mobile systems track the location of mobile devices relative to each other using the device's camera. For example, TouchProjector observes the screen contents of stationary other devices' screen output to infer their relative positions [5], thus requiring a constant visual connection to other screens for tracking. Orienteer instead requires both mobile devices to observe a shared view for registration, such as users' shoes [7].

Using auditory cues, BeepBeep estimates the distance between two devices without synchronizing audio clocks across devices [19]. Qiu et al. build on BeepBeep's distance sensing to show the feasibility of phone-to-phone 3D tracking using microphone switching in static configurations, which does not support interactive scenarios or update rates [20]. SurfaceLink uses the accelerometers, vibration motors, and microphones on mobile devices to detect surrounding devices on a shared surface [10]. AirLink instead requires in-air gestures above an arrangement of devices to detect Doppler effects and recognize gesture input [6]. Pass-them-Around obtains 2D positions through multi-antenna technology [14]. ENSBox and PANDAA reconstruct device arrangements through external sounds [1,24].

BACKGROUND: INTER-DEVICE DISTANCES FROM AUDIO

Tracko computes centimeter-accurate distances between devices using repeatedly exchanged acoustic signals to predict relative 3D positions. We build on the algorithm proposed by Ganeriwal [8] and Peng et al. [19], which determines 1D distances between devices from the delay of emitted sound signals. The algorithm requires measuring the arrival of incoming signals accurately, because a millisecond-level error will produce a meter-level error in estimated distances between devices. Thus, all computation is performed using the clock of the microphone as the frame counter for each device.

While the algorithm requires no synchronization of clocks across devices, it comes at the expense of obtaining *round-trip distances* between devices, but not single distances between a speaker on one and a microphone on another device.

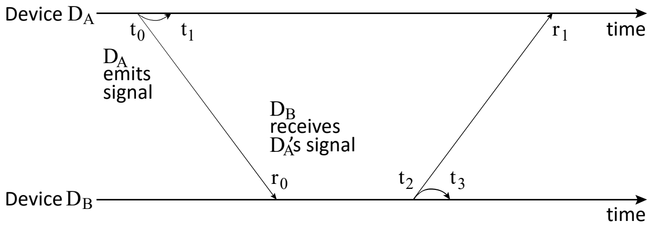

Figure 2: Acoustic ranging between two devices. Device D_A emits a signal at t_0 and records it at t_1. Device D_B receives this signal at r_0. t_2, t_3, r_1 indicate the corresponding time stamps when Device B emits an acoustic signal in response.

This *roundtrip* distance algorithm assumes a single speaker and microphone on each device and works as shown in Figure 2. D_A emits a signal at time t_0 and its own microphone records the signal at t_1. t_0, the time at which the device plays the audio, is unknown, but can be inferred through the known distance d_{SMA} between microphone and speaker. D_B receives the emitted signal at r_0. The distance d_{AB} between D_A's speaker and D_B's microphone results from $(r_0 - t_0 + \sigma)$, where σ is the clock offset between D_A and D_B. While σ is unknown, this offset can be removed by adding the distance d_{BA} between D_B's speaker and D_A's microphone:

$$d_{SMA} = c * (t_1 - t_0), \quad d_{AB} = c * (r_0 - t_0),$$
$$d_{AB} - d_{SMA} = c * (r_0 - t_0 - t_1 + t_0) = c * (r_0 - t_1)$$
$$d_{BA} - d_{SMB} = c * (r_1 - t_2 - t_3 + t_2) = c * (r_1 - t_3)$$

Therefore,

$$d_{AB} + d_{BA} = c * (r_0 - t_1 + r_1 - t_3) + d_{SMA} + d_{SMB}$$
$$= c * ((r_0 - t_3) + (r_1 - t_1)) + d_{SMA} + d_{SMB}$$

r_0, t_3 and r_1, t_1 now represent timestamps on the same clock. As a result, we obtain the roundtrip distance, i.e., the sum of the distance between the speaker and the first device and the microphone on the second plus the distance between the speaker of the second and the microphone on the first device.

TRACKO'S SYSTEM OVERVIEW

To provide ad-hoc 3D tracking information on mobile devices, Tracko comprises three main components (Figure 3):

1. Bluetooth low energy (BLE) manager: This component detects the *presence* of surrounding devices, registers their identifiers, and initially estimates the distance to each device from the signal strength, which we dynamically refine using the accurate distances derived from decoded sound signals.

2. 3D processor: This part estimates relative 3D coordinates of remote devices based on the time-of-flight of incoming inaudible stereo signals and decoded device and speaker identifiers. Tracko's residual model analyzes the series of roundtrip distances from the various speaker-microphone combinations across devices. Our Kalman filter integrates the updates from BLE and audio to determine the final 3D positions, which Tracko then broadcasts to all other devices.

3. IMU manager: This component handles quick device motions and rotations to provide tracking that is responsive to user interaction. Tracko uses inertial sensors updates to adjust the previously computed spatial model of 3D coordinates with low latency. Tracko also detects interactions, such as flipping the device, and forwards them to all apps.

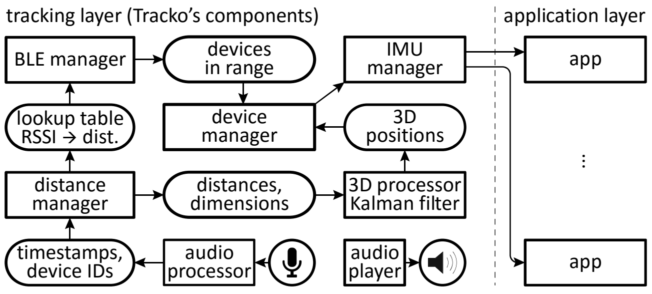

Figure 3: Overview over Tracko's system architecture.

Devices and Requirements

We implemented Tracko to work on Android devices that feature stereo speakers and a microphone. We tested Tracko on HTC One M8 phones and Samsung Nexus 10 tablets. Many other Android devices support stereo output and thus likely run Tracko (e.g., Nexus 6, Nexus 9, Moto G, etc.).

Tracko is implemented in C/C++ using NDK for performance purposes, achieving real-time processing by performing audio processing in NEON. A thin Java layer implements the Kalman filter, handles socket communication, and provides the 3D tracking information to apps using Tracko.

PART 1. BLE SIGNAL ➔ ROBUST DISTANCE ESTIMATION

Tracko's BLE manager continuously scans for available devices with an active BLE signal and updates a list of surrounding devices and their device IDs. For each detected device, Tracko estimates a rough distance from the Received Signal Strength Indicator (RSSI), running an 80th percentile filter to eliminate outlier signal values. Initially, we use a common mapping of signal strengths from −45 to −90 dB to distances between 0 and 100 cm in exponential strides.

Since these distance estimates are coarse [4,15], Tracko calibrates signal strengths on-the-fly with the substantially more accurate distances derived from exchanged audio signals. Tracko maintains a lookup table that for each device maps signal strengths observed by the BLE manager to distances obtained by the position manager and interpolates all subsequent BLE-based distance estimates. This mapping thus becomes more accurate during runtime and reflects potential hardware properties that may affect signal strengths. Our runtime calibration thus provides substantially more accurate BLE-based distance estimations than existing approaches. In addition, BLE-based distances serve as a complementary cue when acoustic signals are unavailable, such as during noise or when covering speakers or microphone when holding.

PART 2. INAUDIBLE STEREO SOUNDS ➔ 3D OFFSETS

We now describe the encoding and decoding of our inaudible stereo signals and how Tracko detects and processes them.

Inaudible acoustic signals that afford precise detection

Tracko obtains its most accurate sense of distances between devices from the time-of-flight of exchanged sound signals. To obtain accurate predictions, we need to reliably determine the precise moment at which sound signals reach a device; every sampling frame Tracko's signal detection is off from the true value adds an error to our distance estimations. To reliably detect incoming signals and compensate for noise, multi-path, low-amplitude and other interference effects, Tracko uses redundant encodings in the inaudible signals.

Design of acoustic codes

Tracko's audio signals consist of two parts: an initial beacon to announce the signal followed by a payload. We designed the signals to support a multitude of devices that potentially play sounds simultaneously when collocated. Figure 4 shows the frequency spectrum of one such signal, which Tracko plays repeatedly. Since the duration of a signal determines the maximum update rate, our audio signals need to be short. Tracko operates audio output and input at 48 kHz, which is the maximum sample rate on current Android devices.

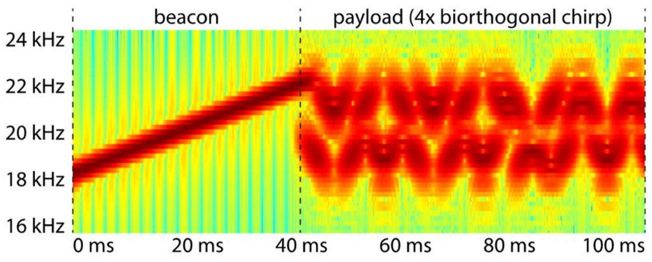

Figure 4: Tracko's inaudible signals comprise an initial beacon and a payload that encodes sender ID, speaker ID, and a counter using biorthogonal chirps for stable audio decoding.

Beacon: Announcing an acoustic signal

The purpose of the beacon in each audio signal is for all receivers to determine the *precise frame* at which the inaudible signal has been received. Tracko uses a linear chirp, spanning the frequency range F_B for a duration of L_B frames. This sweep signal exhibits low cross-correlation and high autocorrelation, which facilitates precise detection and makes it robust to device motion, noise, and ambient sounds [3,9].

Payload: Biorthogonal chirps encode device and speaker ID

Tracko encodes the sender's ID, the speaker ID and a counter into the payload of a signal as shown in Figure 5. This is necessary, because all speakers play the signal simultaneously and receivers need to determine the origin of received signals. Tracko allocates 5 bits to encode the sender ID, which supports up to 32 devices. The sender ID currently results from a hash of the device's BLE identifier. Tracko appends an additional 3-bit counter variable to each message for potential errors of transmission over the audio channel, which is lossy. This counter ensures that the arrival times of one and the same signal are compared across all devices later.

A key part of the payload in Tracko's audio signals is the use of Hamming codes. We encode the concatenated 5+3 bits using two (7,4) Hamming codes, resulting in 14 bits overall. This improves the reliability of signal transmission and aids reliably detecting and discriminating signals that overlap.

Tracko sonifies the 14 bits with 7 biorthogonal chirps as shown in Figure 5. Such chirps represent 2 bits as a series of 4 up or down chirps, each of which is orthogonal to the other three. This makes the recognition of a chirp series robust to superimposed chirps [17]. Tracko plays each biorthogonal chirp as a sound signal across the payload frequency range F_P for a duration of L_P frames per biorthogonal chirp.

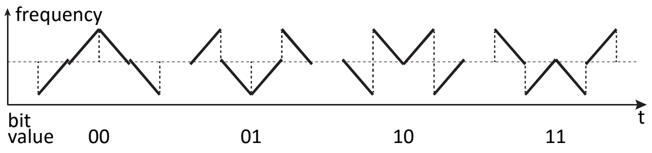

Figure 5: Biorthogonal chirps encode 2 bits as a series of 4 up or down chirps, such that each is orthogonal to each other.

While the duration of the payload is ideally as short as possible, longer durations increase the stability of the signal. In empirical tests, we determined a duration of $L_P = 400$ to be optimal, resulting in 400 * 7 = 2800 frames for the payload.

We had previously experimented with other encoding schemes, such as amplitude, frequency, chirp rate [11, 25], and CDMA. However, they all proved less successful than biorthogonal chirps, mainly because the inaudible frequency range is comparably small and signals needed to be short.

In our evaluation, we examine the design of Tracko's audio signals for various durations L_B and frequency ranges F_B. In particular, we compare the performance of audio signals for shared and separate frequency ranges.

Crossfading signals to prevent sound artifacts

One of Tracko's key features is the use of inaudible sounds. Playing the signal as shown above will produce audible cracking sounds, however, which speakers produce when gain settings change too rapidly [11]. Tracko thus crossfades all individual parts of the sound signal using a fade length of L_F frames during which we interleave signal parts and decrease and increase their amplitudes, respectively, which keeps the total signal duration unaltered as opposed to concatenations of fade-out and fade-ins [11]. $L_F = 80$ frames has proven to be a good length on the devices we tested.

Detecting and decoding audio signals accurately

Tracko continuously records audio through the microphone and analyzes the incoming stream to detect a beacon that announces an encoded payload signal. For all calculations, Tracko solely uses the microphone's frame counter, which represents a timestamp. Tracko requires no synchronization across input and output channels; audio output is independent and may occur at arbitrary times.

To detect a beacon, Tracko first applies a high-pass filter (15th-order Butterworth filter) and cross-correlates the beacon signal starting at each frame using a matched filter. Second, Tracko runs a Hilbert Transform on the correlation scores to identify "thin" peaks [18] and discard those with moderate slopes. The remaining peaks serve as *candidate timestamps* for received beacons. Depending on the power of a received signal, each beacon can produce up to 10 such peaks. We discard spurious candidates by decoding the audio signal *following* each timestamp, through which we verify the Hamming codes and validate each candidate timestamp.

Note that selecting the precise frame is crucial; being off by a single frame directly translates into ~0.5 cm of distance inaccuracy, which propagates through all subsequent steps. Since emitted signals typically result in multiple candidates (e.g., due to multi-path effects), the naïve approach of selecting the largest peak is often error-prone. Tracko thus defers selecting the precise frame until after decoding.

Next, Tracko decodes the payload for *all* candidate timestamps, which starts L_B frames after a timestamp. Correlating the L_P frames of the payload with each of the four possible biorthogonal chirps and selecting the maximum score always results in 14 bits, which represent the Hamming code of the candidate signal. After resolving this code, we obtain 8 bits and an indicator if the Hamming code is flawless. We discard the candidate if there is an error or if the decoded sender ID is not in the list of devices maintained by the BLE manager.

The previous decoding typically eliminates all but one candidate timestamp, granted that all peaks stem from the same signal. If more candidates remain, we compare the decoded sender IDs and speaker IDs and pick the smaller timestamp.

Broadcasting arrival timestamps to other devices

Once a remote signal has been correctly received, i.e., both stereo signals emitted through left and right speaker, respectively, have been correctly decoded, Tracko broadcasts the local timestamps to all surrounding devices through radio. Along with them, Tracko sends the receiver's device ID, the device model, and the local 3D device orientation obtained from the inertial sensors in a receiver packet using radio.

Tracko stores both, the timestamps decoded from audio as well as those from incoming receiver packets. When a new timestamp comes in, either through audio or radio, Tracko invokes the distance calculations we explain below.

Four timestamps ➔ speaker-microphone distances

Tracko computes *roundtrip distances* between the speaker-microphone combinations across devices. Since each device has two speakers and one microphone, Tracko calculates four roundtrip distances altogether as shown in Figure 1c. We denote these distances d_{LL}, d_{LR}, d_{RL}, and d_{RR}. For example, d_{LR} represents the sum of distances from A's *left* speaker to B's microphone plus B's *right* speaker to A's microphone. As mentioned above, this approach requires no cross-device clock synchronization, albeit at the expense of obtaining only the roundtrip distance between two audio components.

Tracko uses the resulting four distances two-fold. First, half of the average of these four distances refines the on-the-fly signal strength-to-distance mapping of Part 1 to improve BLE-based distance estimations. Second, Tracko computes the 3D offsets from A to B from these four distances. Obviously, the quality of the distances we compute determines an upper bound for the quality of the 3D predictions.

4 distances ➔ relative 3D positions using residual models

Tracko predicts relative 3D positions of surrounding devices from the four roundtrip distances of the previous section and the 3D orientations of the two devices. We developed a generic residual model to predict 3D positions in 3 phases.

First, define the coordination system. Both devices have individual coordinates. We thus define vectors from a device's center to its speakers and microphone for the local spL_L, spL_R, mic_L and the remote device spR_L, spR_R, mic_R.

Second, rotate vectors into a shared coordinate system. Since Tracko knows devices' 3D orientations, which share the absolute north due to the magnetometer, we define q_L for the local device's 3D orientation quaternion and q_R for the remote device. The transformation between local and remote device is described through the rotation $q_{rot} = q_R^{-1} q_L$. We can now transform the three remote vectors into the local coordinate system, denoted as $spL_R^*, spR_R^*, mic_R^*$.

Third, predict the 3D offset through residuals. Let $o = (x, y, z)$ denote the 3D offset to the remote device in the local coordinate system. Having transformed the locations of speakers and microphone on the remote device into the local coordinate system, we obtain:

$$d_{XY} = spX_L mic_R + spY_R mic_L\ , \quad \text{where } X, Y \in \{L, R\}$$

and

$$spX_R = spX_R^* + o, mic_R = mic_R^* + o$$

such that we can substitute and define

$$d_{XY} = spX_L(mic_R^* + o) + (spY_R^* + o)mic_L$$

We now compare the predicted and observed distances d_{LL}, d_{LR}, d_{RL}, d_{RR} and compute the squared error as residual:

$$SE = \sum (d_{XY}^* - d_{XY})^2\ , \qquad \text{where } X, Y \in \{L, R\}$$

We then locate the position with minimum squared error in the interaction space using a gradient descent. After evaluating the squared error, Tracko determines two mathematically correct 3D offsets: o_1 and o_2 (Figure 6). We eliminate the inaccurate solution by tracking 3D offsets over time and comparing solutions with previous predictions (similar to [20]).

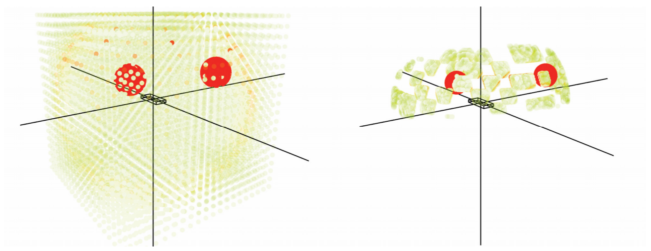

Figure 6: Visualization of the results produced by our residual model. (left) The residual distribution over the whole 3D space. (right) The 1000-point cloud with smallest residuals. The two solutions with lowest residuals are highlighted in red.

Comparison to the performance of a geometric method

Using the residual model described above to predict 3D offsets, Tracko fixes the two main shortcomings of naïve triangulation (proposed by Qiu et al.'s feasibility analysis [20]). First, triangulation relies on a perfect geometry model and requires highly accurate distances and device orientations; otherwise, it produces no real-number solutions for its quadratic equations, which typically occurs during device motion. Our residual model, in contrast, determines the solution through minimizing the square error, which alleviates this problem and thus supports moving devices and user interaction. Second, the geometric model approximates the location of the speaker as the center between two microphones, which requires a large distance between two mobile devices for the geometric model to work. Our residual model, in contrast, accurately works with arbitrary device configurations and scales to any number of speakers and microphones.

Kalman filter: Integrating signals

Tracko's receives tracking information from two sources, the BLE manager and the position manager, potentially at different update intervals. Therefore, Tracko implements a standard linear Kalman Filter to fuse measurements from these multiple sources. From the algorithm we described above, Tracko obtains the acoustic measurement: $\overrightarrow{Z_k} = [x, y, z]$ (i.e., distance and 3D direction). Our BLE ranging algorithm provides the distance measurement d_{BLE} with no directional information. Thus, we reuse the directional information we obtained from the previous audio-based measurement:

$$\overrightarrow{Z_k} = \frac{d_{BLE}}{|\overrightarrow{Z_{k-1}}|} * \overrightarrow{Z_{k-1}}$$

where $\overrightarrow{Z_{k-1}}$ is the last audio measurement.

When Tracko updates the Kalman Filter with a sensor measurement of either type at different sampling rates, BLE or audio, we set the other sensor error covariance to zero since there is an infinite measurement error on the part Tracko does not measure [22]. Our Kalman Filter thus also handles scenarios in which audio-based tracking might fail, such as when a user covers the microphone with their hand, and falls back to BLE tracking.

PART 3. INERTIAL SENSORS DETECT QUICK MOTIONS

Tracko integrates updates from inertial sensors to adjust the previously calculated 3D coordinates of surrounding devices. Upon device rotation, Tracko applies the same rotation matrix to all computed 3D coordinates of remote devices.

To support fast user interaction, Tracko detects device motions and their directions using the accelerometer and gyroscope. Tracko forwards both types of sensor updates (i.e., 3D rotation and quick accelerations) to apps that build on Tracko. In the case of our file-sharing app, a user may rotate the device quickly, but the exchanged audio signals may not respond to this rotation immediately. Therefore, Tracko uses the 3D rotation matrix from the inertial sensors to make tracking responsive. Tracko thereby applies 3D rotations only locally; only after calculating 3D positions using the next available audio signals does Tracko update its Kalman filter and broadcast updates to surrounding devices again.

EVALUATION: AUDIO, DISTANCE, & PREDICTED 3D OFFSET

We demonstrate through our evaluation that Tracko is accurate and robust. These properties make Tracko suitable as an ad-hoc mobile tracking system to support cross-device interaction in 3D space on commodity devices.

We use the following three metrics to evaluate Tracko's performance: 1) Audio recognition accuracy, testing drop-off rates at varying distances and frequency ranges in the audible and inaudible spectrum. 2) Distance accuracy, testing chirp designs for roundtrip distance calculations. 3) 3D offset accuracy, testing the predictions of remote devices' 3D positions and acceptable errors for predictions.

Apparatus

We conducted our evaluation with two HTC One M8 phones. Both devices ran Android 4.4.4 and had two speakers for stereo output and a main microphone. One device was mounted on a tripod and the experimenter held the other one and moved it around. Figure 7 shows our (5 ft)^3 evaluation space.

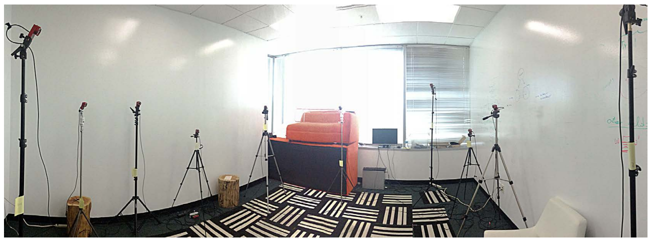

Figure 7: We evaluated Tracko inside (5 ft)^3 space using 11 Optitrack cameras for millimeter 3D tracking ground-truth. (One camera is behind the photographer.)

To obtain ground-truth distances and 3D offsets between devices during our evaluation, we set up an 11-camera Optitrack system inside a conference room. We affixed rigid-body markers to both mobile devices and calibrated the Optitrack to track them with sub-millimeter accuracy.

Conditions and Procedure

We evaluated the metrics listed above in three distance ranges: CLOSE (< 50 cm, representing one user interacting with multiple devices), MEDIUM (50 cm to 1 m, representing multiple users interacting across devices), and FAR (1 m to 1.5 m) to test the limitations of Tracko's audio signals.

Since audio playback and recognition is less reliable in the inaudible spectrum than in the audible spectrum [19], we compared performances in both and evaluated three frequency ranges for F_B (beacon) and F_P (payload): JOINT AUDIBLE (F_B = F_P = [2..6] kHz), JOINT INAUDIBLE (F_B = F_P = [18..22] kHz), and DISJOINT INAUDIBLE (F_B = [18..20] kHz, F_P = [20..22] kHz). DISJOINT INAUDIBLE signals thereby produce less interference, but exhibit a lower signal-to-noise ratio due to their smaller bandwidth. For each range, we tested three beacon lengths L_B: 500 frames (10.4 ms), 1000 frames (20.8 ms), and 2000 frames (41.6 ms). The purpose of these conditions was to measure the effect of interference and decreased signal-to-noise ratio on Tracko's signal recognition.

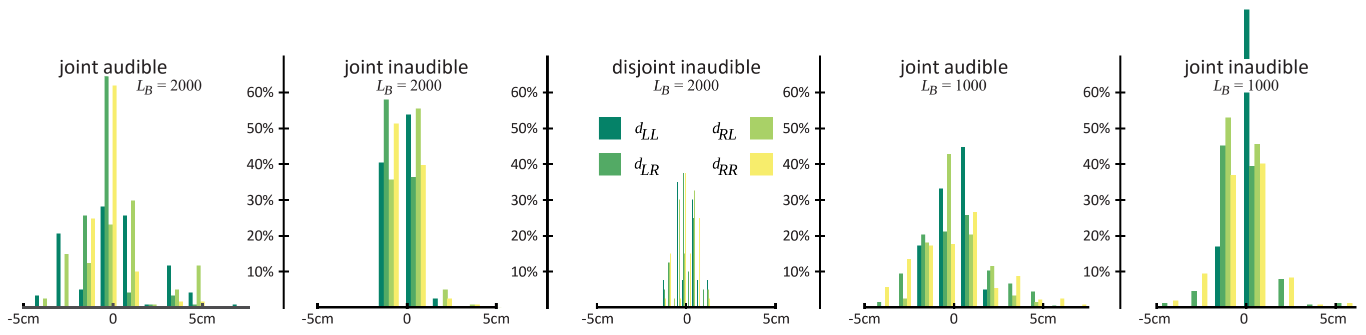

Figure 8: Distance errors for Tracko's predictions of roundtrip distances. Bars present the percentage of error distribution.

During the evaluation, both devices ran Tracko, continuously recorded and processed input audio streams and emitted receiver packets, which we also recorded. We performed all analysis based on the records that the device mounted on the tripod logged. In addition, the workstation powering the Optitrack recorded all locations of both rigid-body markers along with their timestamps for offline comparison.

We evaluated the full combination of all factors with 40 repetitions and computed the average errors for all metrics and their standard deviations to assess stability. Overall, we evaluated 3 distance ranges × 3 frequency ranges × 3 beacon lengths × 40 repetitions = 1080 samples.

Results

Audio Recognition (i.e., chirp detection)

We count audio signals as successfully recognized if and only if Tracko has successfully detected and decoded all *four* chirps that are involved in the audio communication, i.e., the two chirps from the local to the remote device (left speaker and right speaker) and two chirps from the remote device. If Tracko fails to correctly detect or decode a single audio signal, we count this occurrence as not recognized.

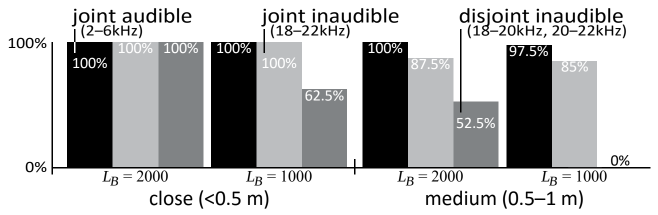

Figure 9: Recognition rates by frequency configuration.

As shown in Figure 9, chirps with $L_B = 2000$ frames produced over 95% recognition rates for CLOSE distances in all three frequency configurations. In MEDIUM distances, the recall rate of DISJOINT INAUDIBLE signals drops to 85% due to the decreased signal-to-noise ratio. This problem is even more severe with a chirp length of only 1000 frames: In no case did Tracko correctly decode *all four* signals in the DISJOINT INAUDIBLE range. We saw a similar effect in the FAR range: at least one of the four detected chirps could not be correctly decoded. We thus discarded this condition from further analysis. Since Tracko requires four successful signal recognitions for each distance estimation, the decreased signal-to-noise ratios have a strong influence on Tracko's recognition rate. For example, 75% for a single detection will result in an overall recognition rate of only $0.75^4 = 31.6\%$.

Distance Accuracy

As shown in Figure 8, Tracko achieves the best distance accuracy with DISJOINT INAUDIBLE signals, resulting in an error of < 2 cm in all cases for CLOSE and MEDIUM ranges.

More specifically, for the beacon length of $L_B = 2000$, both the JOINT INAUDIBLE chirps and DISJOINT INAUDIBLE chirps produced an error of less than 2 cm. The accuracy of JOINT INAUDIBLE signals is a little worse: 6 of the 504 measurements resulted in an error of 11+ cm, which affected the distances d_{LR} and d_{RR} in all 6 cases. This means that one of the devices recognized the peaks from the remote device's right speaker 3 times, which resulted in 3 pairs of distance errors.

For the beacon length of $L_B = 1000$, the accuracy is slightly lower, but reliably under 4 cm. This shows that Tracko's strict decoding guarantees the accuracy of detected signals.

DISJOINT INAUDIBLE signals produced higher distance accuracies than JOINT INAUDIBLE signals, which is due to less interference between beacon and payload chirps. Due to the pattern similarity, the cross-correlation between different chirps are higher than that with random noise. As a result, beacon chirps in DISJOINT INAUDIBLE signals interfere less with payload chirps because of the separate frequencies. For the JOINT AUDIBLE chirps, the distance accuracy is the worst of the three beacon lengths, possibly because of ambient noise in the audible spectrum that affected measurements.

Accuracy of 3D Coordinate Estimation

Since 3D estimation fully depends on the accuracy of distances and we find similar accuracy levels across different chirp settings, we analyze the performance of Tracko's 3D estimation using JOINT INAUDIBLE chirps in this section.

As shown in Figure 10, Tracko overall estimates 3D offsets with an error of 6 cm at CLOSE ranges (0.5 m) and an average error of 15 cm in MEDIUM ranges (up to 1 m). We could not analyze 3D estimates in the FAR range, because Tracko did not decode all four audio signals flawlessly one time.

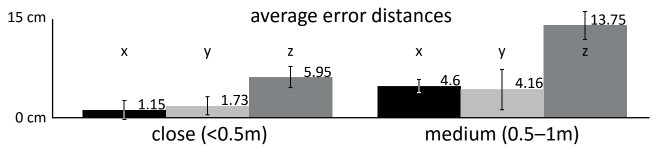

Figure 10: Accuracy of the 3D coordinate estimates calculated by our residual model for the remote device in 3D space.

Since our residual model treats each coordinate in the 3D space independently, we break down the error into three dimensions as shown in Figure 10. The largest error we found in the vertical offsets (z component), which represent the magnitude of the normal vector from the plane defined by the local device (5.95 cm for CLOSE, 13.75 cm for MEDIUM).

Figure 11 plots the error vector of Tracko's 3D estimates in the 2D plane of the device. The estimates offer only a low spread, which indicates that Tracko's residual model produces stable predictions. Overall, we obtained 53 predictions in the CLOSE range and 72 predictions in the MEDIUM range with errors under 10 cm except for five outliers. Due to the discrete nature of the audio sampling, the 3D estimates have fixed granularity, such that many estimates share the same location. Figure 11 also shows the two outliers we saw in the MEDIUM range, which are ~10 cm away from the others.

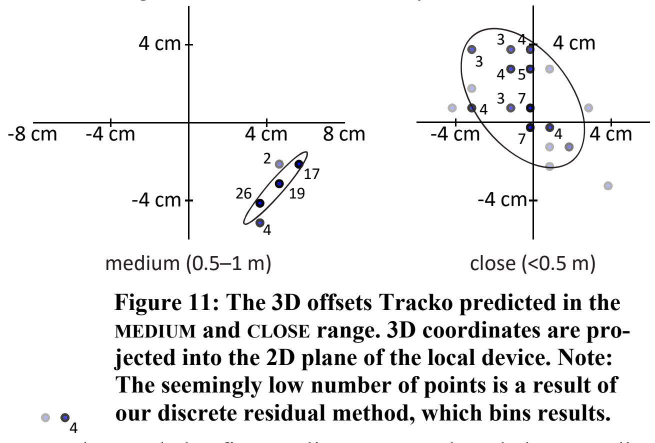

Figure 11: The 3D offsets Tracko predicted in the MEDIUM and CLOSE range. 3D coordinates are projected into the 2D plane of the local device. Note: The seemingly low number of points is a result of our discrete residual method, which bins results.

To understand the five outliers, we analyzed the raw distances of (d_{LL}, d_{LR}, d_{RL}, d_{RR}): (44.45, 32.53, 36.37, 24.45) and (44.45, 33.30, 36.37, 25.22). The ground truth reported by the Optitrack is (44.73, 33.66, 25.39, 14.31), showing that d_{LL}, d_{LR} have been detected correctly, but d_{RL}, d_{RR} have an offset of 10 cm, because the local device recognized the remote device's right speaker with a 14 frames error offset.

DISCUSSION

Tracko achieves comparable distance accuracies in the audible and inaudible spectrum. The JOINT INAUDIBLE range provided better accuracy, but the DISJOINT INAUDIBLE supported larger ranges and better chirp detection. Our final implementation of Tracko thus sonifies chirps and payloads in the joint inaudible range, using a $L_B = 2000$ length to encode the chirp to balance recognition rate and accuracy.

Our evaluation thus shows that Tracko overcomes the limitations of previous systems that require a synchronized playback order for audible signals. Tracko is capable of robustly detecting and decoding simultaneous inaudible signals and thus affords quick interaction with several simultaneous devices at interactive rates even when devices are moving.

Tracko's reliable working range is smaller than that of related systems [20,26], because our restrictive decoding rejects possible candidates in far ranges. As distances increase, the original encoding signal attenuates significantly, which makes decoding more difficult. On the plus side, Tracko's restrictive decoding allows us to detect the true peak and not recognize signals that would lead to erroneous distances.

EVALUATION OF BLUETOOTH LOW ENERGY & DISTANCES

Tracko uses Bluetooth low energy to detect surrounding devices' presence and estimates their distances based on the signal strength. In this section, we evaluate our on-the-fly calibration of signal strength-to-distance mappings and verify our approach with fluctuating RSSI values.

Apparatus and Procedure

The experimenter moved around one HTC One M8 during the evaluation, while the other was mounted on a tripod. Since HTC One M8 devices cannot run in BLE peripheral mode, we attached a Galaxy S5 in peripheral mode to the back of the M8 devices for the purpose of external validity.

We tested our algorithm on 38 locations around the mounted device. In the plane of the mounted device, we tested distances up to 150 cm in steps of 25 cm, both horizontally and vertically offset from the tripod to cover the 2D area (20 points). We additionally tested diagonal distances up to 86 cm at 50 cm below and above the mounted device (9 points for each plane) to cover the z dimension during our evaluation. We collected data for 30 seconds with two repetitions: one for on-the-fly calibration, one for testing. During the collection for on-the-fly calibration, both devices ran Tracko to estimate audio-based distances and record them.

We compare three conditions: TRACKO's on-the-fly calibration, RSSI ranging FORMULA interpolation [2], and "LOCATE BEACONS", a popular debug app in the BLE community [13].

Results

As shown in Figure 12, TRACKO's on-the-fly calibration method estimates distances with an error of less than 20 cm within a range of 1 m, which is significantly more accurate than FORMULA and LOCATE BEACONS. In fact, the errors of the latter two substantially increase with larger distances, rendering their BLE-based distances unusable for Tracko.

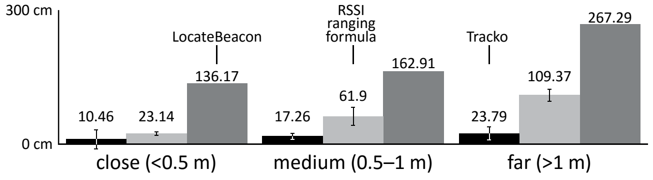

Figure 12: Errors of BLE-based distance estimations. Tracko's on-the-fly calibration yields more accurate distances than existing apps or the typical RSSI-to-distance mapping.

APPLICATIONS

Figure 1a shows an example use case of Tracko: A file-sharing app uses Tracko to detect the 3D location of surrounding mobile devices. The app allows users to share files simply by swiping into the recipient's direction instead of selecting them from a list. The file exchange thereby happens over WiFi. Tracko thereby maintains a 3D model of surrounding devices' locations as users move or rotate their device. Tracko and thus the file-sharing app are also aware of nonexistent devices; when a user swipes a file into a direction where no surrounding device is located, the app denies the transfer as shown in our video figure. Combining 3D positioning with inertial sensors, Tracko supports quick interactions, such as pouring images from one device to another (Figure 13). In general, Tracko can be used as the tracking layer for many of the cross-device systems we discussed in the related work section, allowing them to discard external cameras and work ad-hoc involving only the mobile devices.

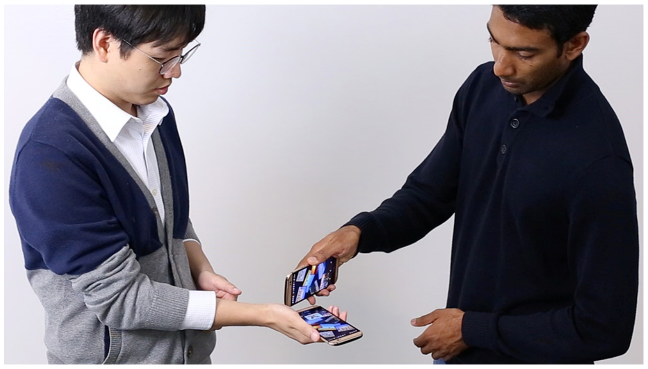

Figure 13: Tracko enables devices to support new interaction techniques, here pouring objects from one device into another.

ONE-WAY SENSING USING TRACKO'S ALGORITHM

Our 3D tracking approach also generalizes to devices that feature only a single audio unit, such as a single microphone on a watch, or two units, such as mobile phones with one speaker and one microphone when two or more surrounding stereo devices are running Tracko for 3D tracking.

Given the established 3D tracking between two stereo devices D_A and D_B, we can infer the location of a 3rd device W as shown in Figure 14. D_A emits an audio signal at an unknown time t_0, which is recorded by D_A and W using their respective local clock times t_1 and w_0. Similarly, D_B emits a signal at t_2, which is recorded by D_B and W at t_3 andw_1.

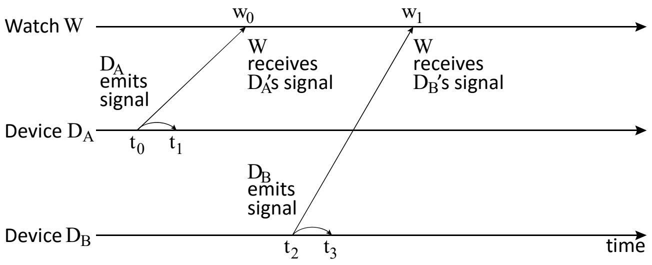

Figure 14 :One-way acoustic ranging. If two devices D_A and D_B have three audio units, a third device W requires only a *single* audio unit for Tracko to calculate its 3D coordinates.

We denote the distances between speaker and microphone on D_A and D_B as d_A, d_B, the distances between W's microphone and the speaker on D_A and D_B as w_A, w_B. Thus,

$$d_A = c * (t_1 - t_0), \qquad d_B = c * (t_3 - t_2)$$
$$w_A = c * (w_0 - t_0), \qquad w_B = c * (w_1 - t_2)$$

Since t_0, t_2 are unknown, we *implicitly* synchronize clocks across devices using known device geometries (i.e., d_A, d_B).

$$\begin{aligned} w_A - w_B &= c(w_0 - t_0) - c(w_1 - t_2) \\ &= c\big(w_0 + (t_1 - t_0) - t_1 - w_1 + t_3 - (t_3 - t_2)\big) \\ &= c(w_0 - t_1 - w_1 + t_3) + w_A - w_B \end{aligned}$$

Although t_1 and t_3 are timestamps on different devices, both can be synchronized using the distance we obtained earlier. We thereby infer the difference in distances between the two speakers on the same phone as described before.

As shown in Figure 15, A_L, A_R, B_L, B_R denote the respective speakers of D_A and D_B; $W = (x,y,z)$ denotes the location of the microphone on the watch. In D_A's coordinate system, D_B's position *(a,b,c)* can be calculated using our algorithm. For any $W = (x,y,z)$, we estimate the geometric distances:

$$A_L W - A_R W = d_1, B_L W - B_R W = d_2, A_L W - B_L W = d_3$$

From our one-way ranging algorithm, we now obtain the observations d_1^*, d_2^*, d_3^*. Comparing the observations and the simulated result, we can now localize the 3D coordinates.

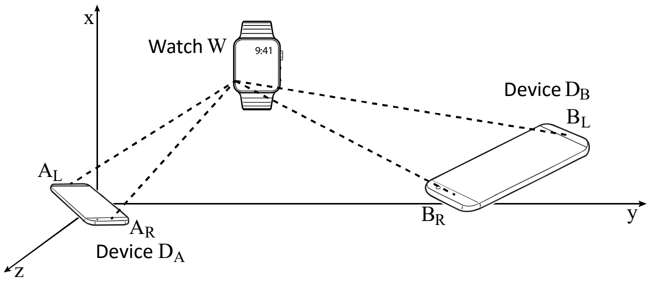

Figure 15: One-way sensing for single-microphone devices.

LIMITATIONS

Acoustic signals enable Tracko to track surrounding devices in 3D, but account for the biggest limitations. Loud sounds in the environment or users obstructing microphones while holding devices will potentially render signals undetectable. In this case, Tracko falls back onto tracking through BLE, which is robust to this kind of interference.

Tracking ranges are currently limited due to the quality of inaudible signals using speakers and microphones in commodity devices [19]. While this currently limits the power of Tracko's signals and thus their range, future device may feature higher-quality audio components and alleviate this.

Dedicated BLE beacons that we attached to all Android devices compensate the lack of BLE in peripheral mode on most Android chipsets. We attached coin-sized Stick'N'Find beacons [23], which also limit our refresh rate. Some device chipsets, however, already support rates of 50 Hz [12].

CONCLUSIONS

We presented Tracko, a mobile-to-mobile 3D tracking system that readily runs on commodity devices. Tracko tracks surrounding devices with an accuracy of 6.5 cm within 0.5 m, bringing a spatial understanding of devices' locations to mobile platforms. Devices running Tracko now understand cross-device interactions with the same spatial notion that users have when using their devices in the physical space.

Tracko accomplishes relative 3D tracking ad-hoc and without the need for installed components in the infrastructure. Tracko's key contribution is the combination of Bluetooth low energy, exchanged inaudible stereo signals, and inertial sensing. While BLE establishes devices' presence and rough distances using the stable Bluetooth signal, the time-of-flight of the inaudible signals devices Tracko yield 3D offsets. Our signal design is scalable and robust enough to handle a number of simultaneous devices. Another key contribution is our on-the-fly calibration of signal strengths to precise distances, which we obtain from audio. Tracko thus addresses a big challenge in BLE-based distance tracking.

We designed Tracko for stereo sound, but our approach would equally work on devices with a single speaker and two microphones (acoustic reciprocity [16]). While one speaker, two mics would reduce the probability of concurrent audio signals, it also doubles the required processing to two audio input streams. Still, most mobile devices now house noise-cancellation microphones. Additional audio components beyond the combinations we used in this paper would benefit Tracko, add extra information, and stability for tracking.

Tracko's design is compatible with emerging BLE indoor tracking (e.g., Apple iBeacon). Detected beacons from the infrastructure can be fused into Tracko's Kalman filter to stabilize the local 3D tracking across devices mobile

Finally, our simple file-sharing application showcases the use of Tracko as a tracking system for applications. Tracko is suitable for many existing interaction techniques in the literature, including those that rely on stationary tracking.

ACKNOWLEDGMENTS

We are indebted to Sven Kratz and FXPAL for letting us borrow their Optitrack system. We thank Sebastian Boring for early discussions and Mayank Goel for feedback on a draft.

REFERENCES

1. Ali, A. M., Yao, K., Collier, T. C., Taylor, C. E., Blumstein, D. T., and Girod, L. Acoustic source localization using the acoustic ENSBox. *Proc. IPSN '07.*
2. Android Beacon Library https://altbeacon.github.io/android-beacon-library/
3. Berni, A.J., Gregg, W.D. On the utility of chirp modulation for digital signaling. *Communications* 21(6), 748–751.
4. Blumrosen, G., Hod, B., Anker, T., Dolev, D., Rubinsky, B. Continuous close-proximity rssi-based tracking in wireless sensor networks. *Proc. BSN '10*, 234–239.
5. Boring, S., Baur, D., Butz, A., Gustafson, S., Baudisch, P. Touch projector: mobile interaction through video. *Proc. CHI '10*, 2287–2296.
6. Chen, K. Y., Ashbrook, D., Goel, M., Lee, S. H., & Patel, S. AirLink: sharing files between multiple devices using in-air gestures. *Proc. Ubicomp '14* 565–569.
7. Dearman, D., Guy, R., Truong, K. Determining the orientation of proximate mobile devices using their back facing camera. *Proc. CHI '12*, 2231–2234.
8. Ganeriwal, S., Kumar, R., Srivastava, M. B. Timing-sync protocol for sensor networks. *Proc. SenSys '03.*
9. Girod, L., Estrin, D. Robust range estimation using acoustic and multimodal sensing. *Proc. IRS '01.*
10. Goel, M., Lee B., Aumi, T., Patel, S., Borriello, G., Hibino, S., Begole, B. SurfaceLink: using inertial and acoustic sensing to enable multi-device interaction on a surface. *Proc. CHI '14*, 1387–1396.
11. Lazik, P., Rowe, A. Indoor pseudo-ranging of mobile devices using ultrasonic chirps. *Proc. SenSys '12.*
12. Liu, J. Chen, C. Energy analysis of neighbor discovery in Bluetooth Low Energy networks. Nokia, 2012.
13. Locate Beacon. https://play.google.com/store/apps /details?id=com.radiusnetworks.locate&hl=en
14. Lucero, A., Holopainen, J., and Jokela, T. Pass-them-around: collaborative use of mobile phones for photo sharing. *Proc. CHI '11,* 1787–1796.
15. Marquardt, N., Hinckley, K. and Greenberg, S. Cross-device interaction via micro-mobility and f-formations. *Proc. UIST '12*, 13–22.
16. Morse, P. M. Theoretical acoustics, McGraw HillBook Co., New York, 37–38, 1968.
17. Othman, M. A. B., Schmid, T., Farhang-Boroujeny, B. Exploiting Quasi-Orthogonal Chirp Signals in Multi-User Access Communication Systems, 2011.
18. Palshikar, G. Palshikar, G. Simple algorithms for peak detection in time-series. *Proc. ADABAI '09.*
19. Peng, C., Shen, G., Zhang, Y., Li, Y., Tan, K. Beep-beep: a high accuracy acoustic ranging system using cots mobile devices. *Proc. SenSys '07*, 1–14.
20. Qiu, J., Chu, D., Meng, X., Moscibroda, T. On the feasibility of real-time phone-to-phone 3d localization. *Proc. SenSys '11*, 190–203.
21. Rädle, R., Jetter, H., Marquardt, N., Reiterer, H., Rogers, Y. HuddleLamp: Spatially-Aware Mobile Displays for Ad-hoc Around-the-Table Collaboration. *Proc. ITS '14.*
22. Smyth, A. and Wu, M. Multi-rate Kalman filtering for the data fusion of displacement and acceleration response measurements in dynamic system monitoring. *MSSP* (21) 2, 2007, 706–723.
23. Stick'N'Find, iBeacon. https://www.sticknfind.com/
24. Sun, Z., Purohit, A., Chen, K., Pan, S., Pering, T., Zhang, P. Pandaa: physical arrangement detection of networked devices through ambient-sound awareness. *Proc. Ubicomp '11*, 425–434.
25. Yang, Z., Pengfei, H., Shasha, J., Luying, Z. Quasi-Orthogonal Chirp Signals Design for Multi-User CSS System. *Proc. ICISE '09*, 631–634.
26. Zhang, Z., Chu, D., Chen, X., Moscibroda, T. Swordfight: enabling a new class of phone-to-phone action games on commodity phones. *Proc. MobiSys '12*, 1–14.