# vTrack: Envisioning a Virtual Trackpad Interface through mm-level Sound Source Localization for Mobile Interaction

Seungeun Chung
North Carolina State University
Raleigh, NC
schung5@ncsu.edu

Injong Rhee
North Carolina State University
Raleigh, NC
rhee@ncsu.edu

## ABSTRACT

Touchscreens on mobile devices allow intuitive interactions through haptic communication, but their limited workspace confines user experiences. In this paper, we envision a virtual trackpad interface that tracks user input on any surface near the mobile device. We adopt acoustic signal as the only medium used for the interaction, which can be handled by lightweight signal processing using inexpensive sensors on mobile devices. In our vTrack prototype, the peripheral device simply emits inaudible acoustic signals through a loudspeaker, while the receiving device performs sound source localization by leveraging a multi-channel microphone array. We build a fingerprint-based localization model using various cues, such as time difference of arrival, angle of arrival, and power spectrum density of the audio signal. The vTrack system integrates the frequency difference of arrival incurred by the Doppler shift to track the sound source in motion. Finally, the position estimations are fed into the extended Kalman filter to reduce errors and smooth the output. We implement our system on Android devices and validate its feasibility. Our extensive experiments show that vTrack achieves millimeter-level accuracy in the moving sound source scenario.

## Keywords

Acoustic signal; Tracking; Mobile interaction; Mobile sensing.

## 1. INTRODUCTION

Although the size of touchscreens on mobile devices is getting bigger [1], the input interface of mobile devices still limits user experiences in mobile interactions. Software keyboards toggled for text input are error-prone due to the fairly small key size compared to human fingers. The major inconvenience incurred by the touch input is the finger itself, which blocks the screen during the interaction. Therefore, we focus on the aspect that any surface near the mobile device can operate as a *virtual trackpad*, if user's movement can be accurately traced. Using this virtual trackpad, users can manipulate the mobile device from outside the touchscreen, without blocking the view of the screen; for example, they can rewind or fast-forward a video while playing it or control characters during the mobile games.

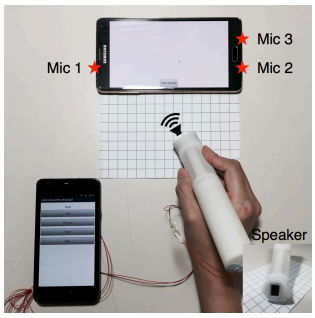



**Figure 1: vTrack operation using two mobile devices. Speaker peripheral is connected to the transmitter device.**

In this paper, we envision vTrack, a virtual trackpad interface that tracks the user's input near the mobile device, which extends the range of mobile interaction over the touchscreen. We build the system based on sound source localization technique using acoustic signal, because it neither requires expensive sensors nor computation-intensive processing on mobile devices. In addition, our system design does not utilize explicit networking between the peripheral and receiving device, because the peripheral does not transmit any data to the receiver; rather, it only emits the sound signal. This excludes any networking module on the peripheral and allows the system to maintain as minimal resource as possible. Low computing and communications on device consequently results in low power consumption, which meets the most significant requisite of peripheral devices.

For the proof-of-concept, we use an off-the-shelf mobile device as the peripheral, which functions as an input device held by a user. Furthermore, we design and build a pen-shaped peripheral using a 3D printer to enhance the usability. In further stages of production, any type of wearable devices with a speaker including smart watches and rings may adopt vTrack to allow ubiquitous interactions with mobile devices. As the peripheral is equipped with a speaker that only costs a few dollars, we expect drastic reduction of the product price when commodified.

The receiving device detects the audio signal and performs sound source localization on-device in real-time. The signal processing and localization procedures are entirely managed by the receiver by taking advantage of its relatively abundant computing resources compared to the peripheral. It uses three-channel built-in microphones that come with a modern smartphone: a primary microphone is for voice recording purposes, while other two are intended to perform stereo recording and cancel the background noise. Figure 1 shows vTrack's operation using two mobile devices.

There exist several challenges in envisioning the vTrack system.

MobiCASE 2016, November 30-December 01, Cambridge, Great Britain

DOI 10.4108/eai.30-11-2016.2267095

The key challenge is to achieve fine-grained localization accuracy with restricted resources on the peripheral side. Our assumption that the peripheral device has an asymmetric capability compared to the receiver device differentiates our work from others in the literature. It limits the system to perform one-way sensing in a synchronization-free manner, with no additional cues for localization other than the sound signal. Second, the audio sampling rate is bounded to $192kHz$ at driver- or operating system-level on mobile devices, which limits the positioning granularity to $1.8mm$ when solely relying on the time difference of audio samples. Finally, the moving sound source increases the uncertainty involved in the acoustic sensing, due to various reasons such as multipath reflections and variance in the signal direction.

In this study, we make contributions by demonstrating that various principles related to the sound source localization technique can be applied in the context of mobile devices and are feasible for positioning the sound source near the mobile device with millimeter-level accuracy. We build a fingerprint-based sound source localization model using the Time Difference of Arrival (TDoA), Angle of Arrival (AoA), and Power Spectrum Density (PSD) of the audio signal. To track the moving sound source with high precision, we leverage Frequency Difference of Arrival (FDoA) incurred by the Doppler shift, and feed the movement direction and velocity measurements to the extended Kalman filter, which reduces the effect of uncertainties in the position estimation. We apply noise reduction technique during the signal processing to prevent the interference from high-frequency noise and improve the accuracy.

We have implemented vTrack using two Android devices. Our experiment verifies that combinations of the aforementioned cues are sufficient to resolve the challenges and envision sound source localization with high accuracy. The AoA of the sound measurements are also noticeable: $0.8^\circ$ of angle estimation error and $1.1mm$ of range estimation error. In general, vTrack achieves $1.1mm$ positioning accuracy in moving sound source scenario, which surpasses the theoretical granularity limited by current audio sampling rate. These promising results demonstrate the potential of vTrack in sound source localization-based mobile interactions.

The paper is organized as follows. Section 2 reviews previous work in sound source localization and mobile interaction literatures that are closely related to our system. Section 3 describes the underlying core ideas of vTrack and validates its feasibility. Section 4 section discusses signal processing and implementation details on Android, and Section 5 presents our extensive experiments of the system. Finally, Section 6 concludes the paper.

# 2. RELATED WORK

## 2.1 Acoustic Localization

### *2.1.1 Large-Scale Scenarios*

There exists significant research on acoustic signal-based positioning techniques. BAT [8] is an indoor localization system, which is based on the time of flight measurement of ultrasound signals. Similarly, Cricket [13] adopts ultrasound in combination with the RF signal. It infers the distance through time difference of arrival of the concurrent transmissions of both signals. These systems achieve a centimeter-level resolution, but require densely deployed infrastructural supports consisting of specially designed ultrasound transceivers. Recently, peer-assisted acoustic ranging scheme [10] is proposed to improve the localization errors induced from the intrinsic limit of the RF signal propagation. Also, PANDAA [15] determines the relative locations of networked sensors measuring the time difference of arrivals of ambient sound in the room. Aforementioned systems aim to localize the object in large-scale indoor scenarios (i.e., building or room-size environments), while our operation space is bounded to tablet-size range.

Other acoustic localization schemes mainly targeted for outdoor environments include ENSBox [7], which is an angle of arrival-based distributed system integrated on ARM platform with four-channel microphones at each node. By virtue of its relatively abundant microphone array that is geometrically arranged in 3D space, it achieves high accuracy in object positioning, up to few centimeters. Whistle [18] leverages the time difference of arrival of the acoustic sound observed at different receiving devices, which serve as the basic infrastructure of the system. Finally, [20] classifies the position of device in a car to detect driver phone use by analyzing the TDoA of acoustic signals emitted from four-channel built-in speakers.

### *2.1.2 Small-Scale Scenarios*

Our work is closely related to proposals that study ranging between nearby devices. BeepBeep [12] is an acoustic ranging mechanism that operates on off-the-shelf mobile devices. Two devices emit a beep sound in turn and simultaneously record both beep sounds. From the (self) audio recording, each device can measure the elapsed time between two beep sound events. By exchanging this time information, devices can obtain the time of flight of the two beeps, and consequently get the distance between two devices. BeepBeep assumes two mobile devices to have equivalent computing capability, and coordinated with wireless communications such as Wi-Fi or Bluetooth. Its ranging accuracy exceeds one centimeter, which is inappropriate for applications that require precision.

Based on the BeepBeep procedure, [14] extends the phone-to-phone localization to three-dimensional space. By leveraging multiple microphones and other inertial sensors on mobile devices, it performs 3D triangulation using time of arrival and signal power of the acoustic signal. Through continuous localization, each device can estimate the other device's relative position and track its movement. However, this work aims for a meter-level operation range, and achieves 3D localization accuracy with several centimeters of position error. Similarly, [22] proposes a high-speed acoustic ranging scheme in 3D space for fast moving mobile devices, which enables phone-to-phone motion games.

## 2.2 Mobile Input Techniques

Various types of mobile input methods have been proposed recently, which refrain from direct interactions with the touchscreen due to the inherent limitations of touch interface. UbiK [17] is a portable text-entry method that requires a keyboard outline printed on a paper. It makes use of the dual-microphone interface on a mobile device to localize the keystroke sound on solid surfaces such as desk. UbiK copes with the acoustic multipath fading through Amplitude Spectrum Density (ASD) of different keystroke sound, and localizes distinct keystroke locations by fingerprinting-based signature matching. This scheme is highly dependent upon the surface environments it works on, and demands of repetitive training process every time when the workspace changes. Similarly, [11] snoops keystrokes behind the keyboard based on the TDoA and other acoustic features.

Okuli [21] takes Visible Light Sensing (VLC) technology into account to let the VLC-capable mobile device to sense the movement of user's finger. It extends the mobile interaction workspace to nearby surfaces and enables a virtual trackpad and keyboard input. Using an LED transmitter and two photodetectors as peripherals, Okuli builds a model-driven framework based on the physical properties of the visible-light channel for finger positioning. It achieves around one-centimeter scale precision, but requires a specially de-

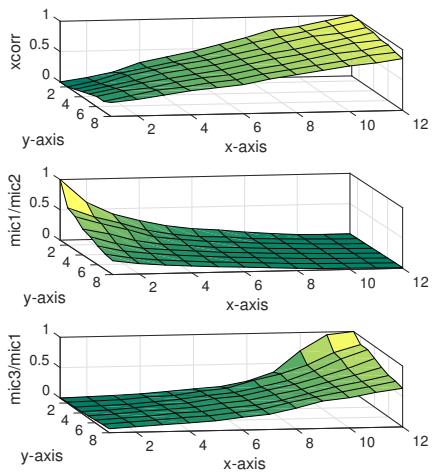



**Figure 2: Normalized cross-correlation value and power ratio of microphones.**

signed peripheral equipped with additional sensors. Other than visible light, the uses of RFID tag and receivers [16] and inertial motion sensors [5, 19] for handwritten text in the air have been proposed for mobile input.

Smartpen (i.e., digital pen) devices available on the market adopt various technologies to capture the user's path of movement. Livescribe [3] smartpen is equipped with an infrared camera at the tip that determines its position on the page when used with the paper pre-printed with a dot pattern. To record the input, the user should write on this special paper, which leverages the dot-positioning system. Equil [2] pairs with its receiver, which uses both ultrasound and infrared to locate the smartpen's position on the paper. By attaching the receiver to the top of the paper, the user can take notes on any paper surface. Phree [4] turns any surface into a virtual canvas by adopting a 3D laser interferometer and an optical sensor, in addition to motion-capturing sensors embedded in the device. Smartpens usually process the sensor readings on-device and send the interpreted data to the paired mobile device over networks such as Bluetooth. The limitation of smartpens is their high price range– from $100 to $200–due to the adoption of various expensive sensors and computing resources to achieve a high level of accuracy when determining the position of the pen.

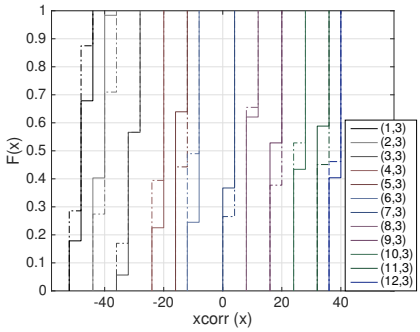
# 3. SOUND SOURCE LOCALIZATION

vTrack is composed of two mobile devices as illustrated in Figure 1. Receiving device is placed horizontally to perform audio signal processing and display the cursor on screen along with the user's movement. The peripheral periodically emits audio signals, and is allowed to move around on the virtual trackpad printed on a piece of paper, which is a $10\times13$ grid with $1cm$ unit. We verify later that the size of the virtual trackpad is not necessarily limited to the distance between microphones, but can be enlarged to twice its current size.

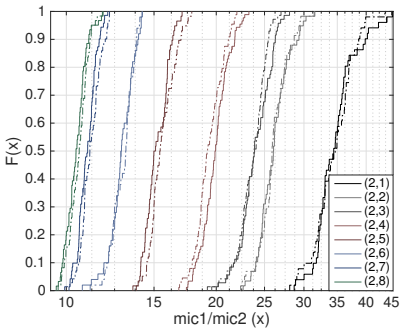
## 3.1 Coordinate Positioning

### *3.1.1 Time Difference of Arrival*

Multilateration is a common navigation technique based on the measurement of the difference in distance to two known locations.




(a) Cross-correlation




(b) Power ratio of two microphones

**Figure 3: Distribution of cross-correlation and power ratio of microphones measured on two different days.**

Because it utilizes the relative difference instead of the absolute distance measures, it is free from clock synchronization between the signal transmitter and its receiver. We benefit from this property to build a synchronization-free one-way sensing system, and use the time difference of arrival of sound for coordinate localization. For TDoA computation, we adopt Generalized Cross Correlation with Phase Transform (GCC-PHAT) algorithm [9], which is known to effectively reduce the noise and reverberation and perform well in actual noisy environments [6]. Index count of the maximum absolute value of GCC-PHAT is considered as the time lag between two audio signals.

The first graph in Figure 2 presents the normalized cross correlation values ($\Delta I$) for audio signals recorded at microphones 1 and 2 illustrated in Figure 1. $\Delta I$ values gradually increase as the sound source moves from left to right. Figure 3(a) shows the cumulative distribution function (CDF) of $\Delta I$ values that are repetitively collected for 50 times on each column of a certain row. Most of the measurement points have cross correlation values oscillating within the range of two to three sample indexes, which shows high stability. Dotted lines represent another measurement results collected on a different day, and we can observe that $\Delta I$ values at the same point are consistent over time and thus reproducible.

Based on this consistency, we first perform linear regression on the measurement data and model the relationship between TDoA ($\Delta I$) and the x-coordinates (i.e., left-right). However, because TDoA is a relative difference measure, locations that have a constant value form a hyperbolic curve, which result in modeling ambiguity. To locate the exact position on the hyperbola, we introduce the third microphone embedded at the bottom of the mobile device. The second measurement taken by a different pair of microphones (i.e.,

microphones 1 and 3) will produce the second curve, which intersects with the first one. When the two curves are compared, a small number of possible locations are considered as candidate positions of the sound source.

Although three-channel microphones reduce the search space, the positioning resolution is limited by the audio sampling rate. TDoA resolution is $0.005ms$ with $192kHz$ sampling rate. When we convert it to distance, the resolution becomes $1.8mm$. Considering the uncertainties included in the measurements, deterioration of positioning accuracy is inevitable. To handle this, we introduce additional cues from audio signal characteristics.

### 3.1.2 Power Spectrum Density

Total power level of the signal tends to increase as the sound source approaches the receiver and decrease during the recession. However, as the microphone is usually located in the middle of edges and embedded perpendicularly to the edge, the change in power level does not show linear relationship with the y-axis (i.e., front-back). When the sound source recesses from the receiving device along the y-axis, there exists a non-monotonic section. In addition, we observe that sensitivity of the power level at two microphones highly differs. Because minor microphones are designed for background noise canceling, it performs better than the major microphone in detecting the sounds with high frequency at a distance. Contrarily, the major microphone is targeted for voice signals in short range, so it cannot sensitively detect high frequency sound signals that are far away from it.

From our extensive experiments, we observe that the amplitude of signal emitted by the transmitter is not consistent throughout the experiment session due to the hardware limitation, although the volume of the sound is set to constant value. Even small variation in signal power can be interpreted as error in modeling the distance to the sound source using the absolute signal amplitude. Therefore, we decide to adopt the ratio of the power level at microphones, because the ratios remain consistent even the absolute power may vary. Lower two plots in Figure 2 present the power ratios of three microphones.

To verify the temporal stability of power ratio at each point of the grid, we measure the power readings on two different days. Figure 3(b) shows the CDF of power ratio of two microphones that are repetitively collected for 50 times on each row of the same column. Solid lines represent results from the first session and dotted lines stand for the second session. Distribution of the power ratio at each point remains the same for different measurement sessions, which shows the temporal stability of power ratio.

After collecting the cross-correlation value and power ratios of three microphones at each point of the grid, we construct a database that consists of five-tuple data:

$$[xcorr12, xcorr13, xcorr23, mic1/mic2, mic3/mic1]$$

where $xcorr12$ stands for the cross-correlation value between microphones 1 and 2. To find the best estimates of x- and y-coordinate of the input data, we perform k-nearest neighbor (KNN) search on the database using the Euclidean distance. To make the matching algorithm robust to abrupt erroneous readings, we use the moving average of input data. We also keep track of the previous states as a reference, and intelligently filter out unexpected errors.

However, some points have overlapping range of power ratio, which results in ambiguity in differentiating two points. For example, two neighboring points (2,2) and (2,3) as well as points (2,7) and (2,8) in Figure 3(b) have power ratio distribution that are very close to each other. This is due to the refraction of sound signal by the corner of the device in addition to the multipath reflections.

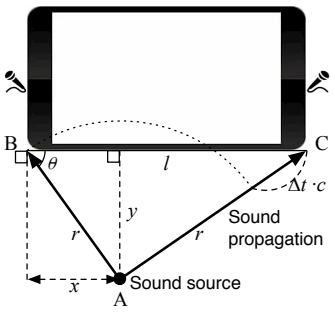



**Figure 4: Angle of arrival computation using two microphones.**

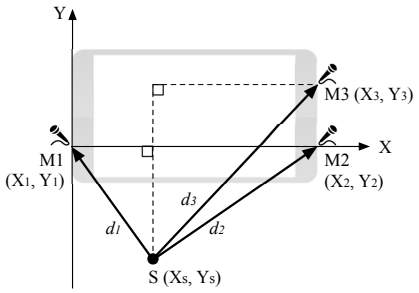



**Figure 5: Coordinate estimation using three-microphone array.**

Thus, we adopt the angle of arrival of the signal and integrate it in the positioning algorithm to correct possible errors.

### 3.1.3 Angle of Arrival

First, we start with a scenario using two microphones. Consider a $\triangle$ABC between the receiving device and the sound source as shown in Figure 4, where AoA is $\theta$ by assuming that the microphone is located at the corner of the device. By applying the cosine rule for $\angle$ABC in $\triangle$ABC, we get

$$\begin{aligned}\cos\theta &= \frac{l^2 + r^2 - (r + \Delta t \cdot c)^2}{2l \cdot r} \\ &= \frac{x}{r}\end{aligned}$$

where:

- $l$ = Distance between two microphones
- $r$ = Shorter distance from sound source to microphone
- $\Delta t$ = Time difference of arrival at two microphones
- $c$ = Speed of sound in dry air at 20 °C ($343.2m/s$)
- $x$ = Offset of the sound source from the left edge.

We can compute the range $r$ and the angle of arrival $\theta$ using TDoA, and consequently get the distance $y$ of the sound source to device as follows.

$$r = \frac{l^2 - (\Delta t \cdot c)^2 - 2l \cdot x}{2\Delta t \cdot c} \tag{1}$$

$$\theta = \arccos\left(\frac{x}{r}\right) \tag{2}$$

$$y = r \cdot \sin\theta \tag{3}$$

### 3.1.4 Multilateration

Based on our x and y coordinate estimation using two microphones, we extend the method by including additional information

from the third microphone, which contributes to improve the positioning accuracy. Because handling a set of nonlinear hyperbolic equations is challenging on resource-constrained mobile devices, we derive the sound source position as follows. By assuming the receiving device is located in a coordinate system as shown in Figure 5, coordinates of three microphones can be represented as M1, M2, and M3, respectively. Then, the distance from the sound source to each microphone is formulated as follows:

$$d_1 = \sqrt{(X_s - X_1)^2 + (Y_s - Y_1)^2}$$
$$d_2 = \sqrt{(X_s - X_2)^2 + (Y_s - Y_2)^2}$$
$$d_3 = \sqrt{(X_s - X_3)^2 + (Y_s - Y_3)^2}$$

By subtracting the square of distances $d_1$ and $d_2$, we get

$$d_1^2 - d_2^2 = X_1^2 - X_2^2 - 2X_s(X_1 - X_2) - 2Y_s(Y_1 - Y_2) + Y_1^2 - Y_2^2$$

which can be rearranged as follows:

$$2X_s(X_1 - X_2) + 2Y_s(Y_1 - Y_2) = X_1^2 - X_2^2 + Y_1^2 - Y_2^2 - d_1^2 + d_2^2$$

Similarly, relationship between $d_1$ and $d_3$ becomes

$$2X_s(X_1 - X_3) + 2Y_s(Y_1 - Y_3) = X_1^2 - X_3^2 + Y_1^2 - Y_3^2 - d_1^2 + d_3^2$$

Then, these relationships can be formulated in matrix form

$$2A \begin{bmatrix} X_s \\ Y_s \end{bmatrix} = B$$

where:

$$A = \begin{bmatrix} X_1 - X_2 & Y_1 - Y_2 \\ X_1 - X_3 & Y_1 - Y_3 \end{bmatrix}$$
$$B = \begin{bmatrix} X_1^2 - X_2^2 + Y_1^2 - Y_2^2 - d_1^2 + d_2^2 \\ X_1^2 - X_3^2 + Y_1^2 - Y_3^2 - d_1^2 + d_3^2 \end{bmatrix}$$

The distance difference between microphones $\Delta d_{21}$ and $\Delta d_{31}$ can be represented using the TDoA measures.

$$\Delta d_{21} = d_2 - d_1 = t_2 \cdot c - t_1 \cdot c = \Delta t_{21} \cdot c$$
$$\Delta d_{31} = d_3 - d_1 = t_3 \cdot c - t_1 \cdot c = \Delta t_{31} \cdot c$$

As the range $d_1$ is known from the Equation 1, the distances $d_2$ and $d_3$ are obtained. Therefore, the matrix B becomes

$$B = \begin{bmatrix} X_1^2 - X_2^2 + Y_1^2 - Y_2^2 + \Delta d_{21}^2 + 2\Delta d_{21} d_1 \\ X_1^2 - X_3^2 + Y_1^2 - Y_3^2 + \Delta d_{31}^2 + 2\Delta d_{31} d_1 \end{bmatrix}$$

Finally, by projecting the microphone coordinates on the Cartesian coordinate system, we substitute the coordinates $(X_1, Y_1) = (0,0), (X_2, Y_2) = (l,0)$, and $(X_3, Y_3) = (l,h)$, and get matrices A and B,

$$A = \begin{bmatrix} l & 0 \\ l & h \end{bmatrix}$$
$$B = \begin{bmatrix} l^2 - \Delta d_{21}^2 - 2\Delta d_{21} d_1 \\ l^2 + h^2 - \Delta d_{31}^2 - 2\Delta d_{31} d_1 \end{bmatrix}$$

where:

$l$ = Distance between microphones 1 and 2
$h$ = Distance between microphones 2 and 3.

Now, we can infer the position of the sound source using the TDoA measurements with three-microphone array. However, in the moving sound source scenario, the result may include errors due to the unexpected deviation in the direction and/or angle of the speaker. Therefore, we leverage additional information that can be obtained from the moving sound source.

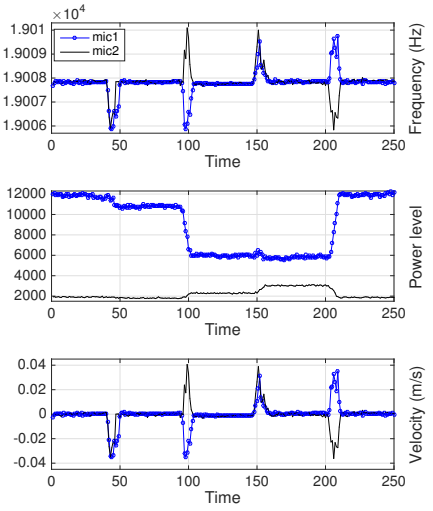



**Figure 6: Frequency and power level measured at microphones 1 and 2 and the estimated velocity of the moving sound source.**

## 3.2 Movement Detection

### *3.2.1 Frequency Difference of Arrival*

Doppler effect refers to the phenomenon where the frequency of a sound wave increases as the sound source approaches observer, and decreases during the recession. To detect the movement of the sound source relative to the observer, we leverage the change in frequency of the sound wave. When the sound source approaches the observer with velocity $v_s$ during the period $T$ of the sound wave, the wavelength becomes $\lambda' = \lambda - v_s T$ as the sound source gets closer as $v_s T$. By substituting the equation with properties $\lambda' = v/f'$, $\lambda = v/f$, $T = 1/f$, and $v$ with speed of sound $c$, we get the observed frequency $f'$.

$$f' = \left(\frac{c}{c - v_s}\right) f$$

where $f$ is the original frequency emitted by the transmitter. Thus, we can compute the velocity of the sound source $v_s$:

$$v_s = c\left(1 - \frac{f}{f'}\right)$$

where $v_s$ is positive when the sound source approaches the receiver and negative in the other direction. As we know the velocity of the sound source, we can also derive the amount of movement by multiplying $\Delta t$, which can be known from the timestamp.

As an example, we measure the change in frequency and received power level at each microphone when the sound source moves along the path connecting points $(x,y) = (4,3) \rightarrow (4,8) \rightarrow (9,8) \rightarrow (9,3) \rightarrow (4,3)$ in the grid sequentially, drawing a rectangle clockwise. Because the first movement $(4,3) \rightarrow (4,8)$ puts the sound source away from both microphones, frequency readings show abrupt decrease at both microphones in Figure 6. Similarly, the third movement where the sound source approaches both microphones makes the frequency increase at both microphones. We can observe that when the sound source moves along the y-axis, the rise and fall of frequency at both microphones coincide. Contrarily, frequency changes at two microphones are opposite when the sound source follows the x-axis direction. By leveraging this frequency

difference of arrival, we can infer the direction of the sound source movement.

The change in power level at each microphone can also assist the movement detection. As the sound source moves toward the microphone, the power level increases and decreases in the other direction. In Figure 6, we can observe the asymmetric performance of two microphones. Microphone at the bottom of the device (mic 2) shows linear power level transition as a function of distance. However, microphone at the top (mic 1) responses with high sensitivity to the sound in short distance, while the response sensitivity sharply decreases after some distance. Although there exists ambiguity in long distance, each movement to different direction is differentiable with the increase and decrease of power level as follows: $\downarrow = [decrease, decrease]$, $\rightarrow = [decrease, increase]$, $\uparrow = [increase, increase]$, and $\leftarrow = [increase, decrease]$, where the arrows represent the movement direction and the tuples stand for the power level change of $[mic1, mic2]$.

Finally, we can now use the aforementioned cues regarding the direction of the sound source to assist our decision making process. We can compute the velocity of the moving sound source by using the frequency difference as shown in Figure 6, and consequently calculate the distance of movement by taking the integral of the plot. In this specific example, each movement involves 7 samples each, where the sample interval is set to $300ms$. Thus, the amount of displacement can be approximated to $4.2cm$, which is close to the ground truth.

### *3.2.2 Extended Kalman Filter*

Due to the noisy measurements induced by the movement, the uncertainties in inferring x and y-coordinates are unavoidable. To filter out erroneous readings and smooth the output, we introduce extended Kalman filter, which is a Markov model that assumes dependence on the previous state only. The filter deals with uncertain measurement about the dynamic system that is continuously changing, and makes an educated estimation about what the next state will be. It is widely used in technology such as navigation and control of vehicles, time series analysis in signal processing, and robotic motion control and trajectory optimization. In the prediction step, the filter produces estimates of the current state taking uncertainties into account. Once new measurement is observed, estimates are updated using a weighted average, with more weight being given to estimates with higher certainty.

The state vector $x_k = \begin{bmatrix} p_x & v_x & p_y & v_y \end{bmatrix}^\top$ in our model involves estimating not only the $x$ and $y$ coordinates but also its $x$ and $y$ velocities. These four states must be estimated given only noisy measurements of range and angle. The measurement vector $z_k = [r\ \theta]^\top$ contains the actual range and angle readings as illustrated in Figure 4. The state transition and measurement models are represented as follows:

$$x_k = f(x_{k-1}) + w_k$$
$$z_k = h(x_k) + v_k$$

where $w_k \sim \mathcal{N}(0, Q_k)$ and $v_k \sim \mathcal{N}(0, R_k)$ are the process and measurement noises, which are both assumed to be zero mean multivariate Gaussian noises with covariance $Q_k$ and $R_k$, respectively.

The process noise matrix $Q_k$ measures the variability of the input signal away from the ideal transitions defined in the state transition matrix. Larger values in this matrix mean that the input signal has greater variance and the filter needs to be more adaptable. Smaller values result in a smoother output, but the filter is not as adaptable to large changes. In our model, we define $Q_k$ through some fine-tuning process. The measurement noise matrix $R_k$ defines the error of the measuring device. We determine this accuracy empirically as well. Decreasing the values in this matrix means we are optimistically assuming our measurements are more accurate, so the filter performs less smoothing and the predicted signal will follow the observed signal more closely. Conversely, increasing the values means less confidence in the accuracy of the measurements, so more smoothing is performed.

The state transition function $f$ computes the predicted state from the previous estimate. Similarly, the measurement function $h$ computes the predicted measurement from the predicted state. As the displacements and velocities are non-linearly related to the range and angle, the filter algorithm requires calculation of a matrix of partial derivatives (the Jacobian) for the state and measurement equations. So, in predict stage, predicted state $\hat{x}_k$ and predicted covariance $P_k$ are represented as follows. $F_k$ is the Jacobian matrix of state transition function.

$$\hat{x}_k = F_k \hat{x}_{k-1}$$
$$P_k = F_k P_{k-1} {F_k}^\top + Q_k$$

The measurement update equation uses the range and angle, which are related to the $x$ and $y$ displacements as shown in Equations (1)-(3). The Jacobian matrix $H_k$ for the measurement equations is computed as follows.

$$h(x_k) = \begin{bmatrix} r_k \\ \theta_k \end{bmatrix}$$

$$H_k = \left. \frac{\partial h(x_k)}{\partial x_k} \right|_{\hat{x}_k}$$

The update stage takes the measurement $z_k$ and computes the measurement residual $\tilde{y}$ and covariance $S_k$, and gets the Kalman gain matrix $G_k$.

$$\tilde{y} = z_k - h(\hat{x}_k)$$
$$S_k = H_k P_k {H_k}^\top + R_k$$
$$G_k = P_k {H_k}^\top {S_k}^{-1}$$

Finally, we get our new best estimates for state $\hat{x}'_k$ and covariance ${P_k}'$.

$$\hat{x}'_k = \hat{x}_k + G_k \tilde{y}$$
$${P_k}' = P_k - G_k H_k P_k$$

The filter recursively iterates these two stages and updates the position estimation of the moving sound source.

# 4. IMPLEMENTATION

We implement the proposed model using Android devices and run the sound source localization algorithm in real-time. Samsung Galaxy Note 4 is used as a receiver, while Samsung Galaxy S II operated as the audio signal transmitter. As the Android OS does not provide APIs that can handle three-channel audio streams by default, we modify the Android audio framework system to deliver raw PCM data generated at the audio hardware to the application layer. The process of vTrack system is illustrated in Figure 7 as a flowchart.

## 4.1 Audio Signal Design

As we neither use time synchronization method to synchronize two devices nor exchange any informative data between them, the audio signal is the only mean that can control our system. Thus, the signal and its protocol should be selected with care. According to our experiment, Samsung Galaxy Note 4 has high quality frequency

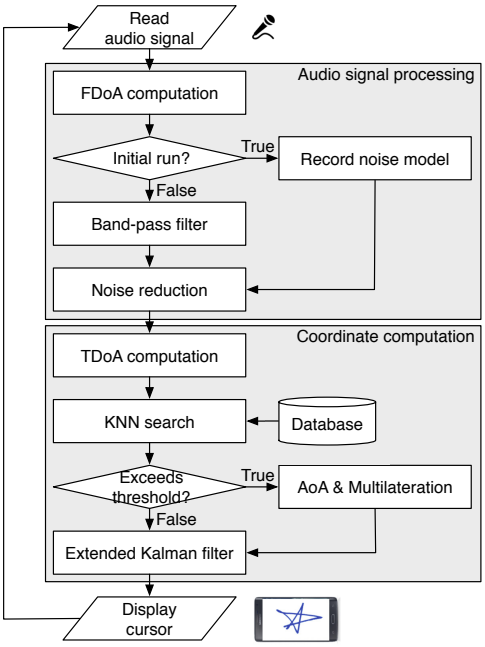



**Figure 7: vTrack's operation flowchart.**

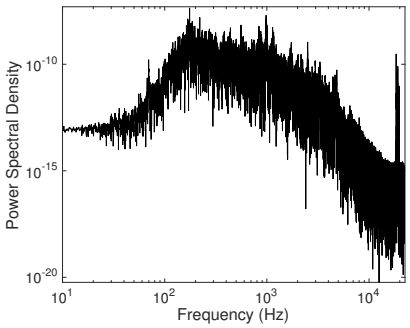



**Figure 8: Ambient noise capture at a local Starbucks cafe.**

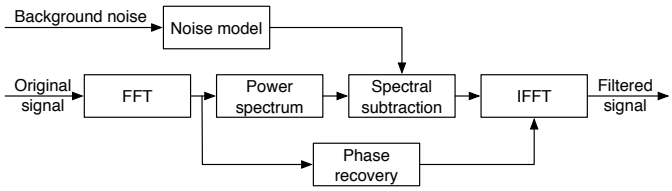



**Figure 9: Noise reduction procedure of vTrack.**

response ranging from $20Hz$ up to $20kHz$. We choose the frequency range of the audio signal to be $19kHz$, which is almost inaudible. It does not generate any disturbing sound during the operation, and also can be used in usual environment without interference because it does not overlap with the frequency range of voice speech.

The receiving device continuously reads audio data from its audio hardware in chunks, so the signal emitted by the transmitter should fit into the buffer size with good cross-correlation property. An audio signal that is too short may not be clearly distinguishable with the ambient noise. On the other hand, a longer signal consumes excessive time in reading and processing the audio data, which incurs delay in the update interval. Based on our extensive experiments, we decide to generate a $2ms$ long sine wave every $21ms$, which gives success rate over 98%.

## 4.2 Signal Processing

We choose the size of buffer that system periodically reads audio data in and the period of audio sound to be the same, so that each audio chunk contains one audio sample. Aligning the buffer size and the audio signal period does not support perfect synchronization between the transmitter and receiver. Although the system issues a command to play an audio out at a certain time, there exists some delay in time when the signal is actually emitted. This latency in audio playback is still an open issue in Android OS. Thus, the clock drift between two devices is inevitable. To guarantee the signal detection in every chunk of audio data, we force shift the starting index of buffer. When there is more than one peak that is larger than the threshold value, we read the next signal data from the buffer starting from index increased by half of the buffer size.

To detect FDoA at two microphones, the received audio signal should be transformed from time domain into frequency domain. Representing the given signal in frequency domain is done via Fast Fourier Transform (FFT), which implements Discrete Fourier Transform (DFT) in an efficient manner. Usually, power spectrum is desired for analysis in frequency domain. In a power spectrum, power of each frequency component of the given signal is plotted against their respective frequency. We compute $P_x(f) = X(f)X^*(f)$, where $X(f)$ is the frequency domain representation of the signal $x(t)$, and $X^*(f)$ is the complex conjugate of $X(f)$. We implement FFT procedure that returns a Power Spectrum Density (PSD) array of the received signal. From the output PSD array, we extract the maximum value where its frequency is within the frequency threshold. In order to get the TDoA between two signals $x_i(t)$ and $x_j(t)$ at each microphone, the GCC-PHAT is computed as follows:

$$G_{PHAT}(f) = \frac{X_i(f)X_j^*(f)}{|X_i(f)X_j^*(f)|}$$

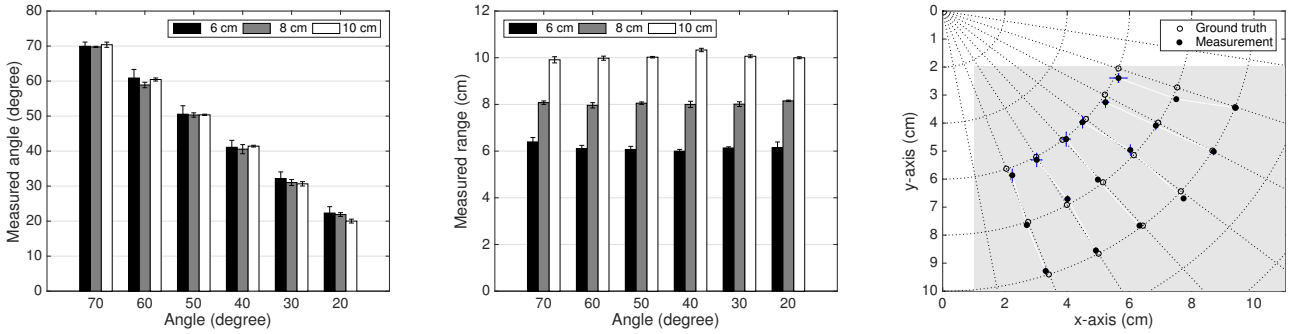
and the time difference for these two microphones is estimated:

$$d_{PHAT}(i,j) = \underset{d}{\operatorname{argmax}}(R_{PHAT}(d))$$

where $R_{PHAT}(d)$ represents the inverse Fourier transform of the function $G_{PHAT}(f)$. The audio sampling rate at the receiver is set to $f_s = 192kHz$, which is the highest value supported by modern Android devices. By multiplying the cross correlation value with the period of sample, we can compute TDoA of sound signals captured at two microphones.

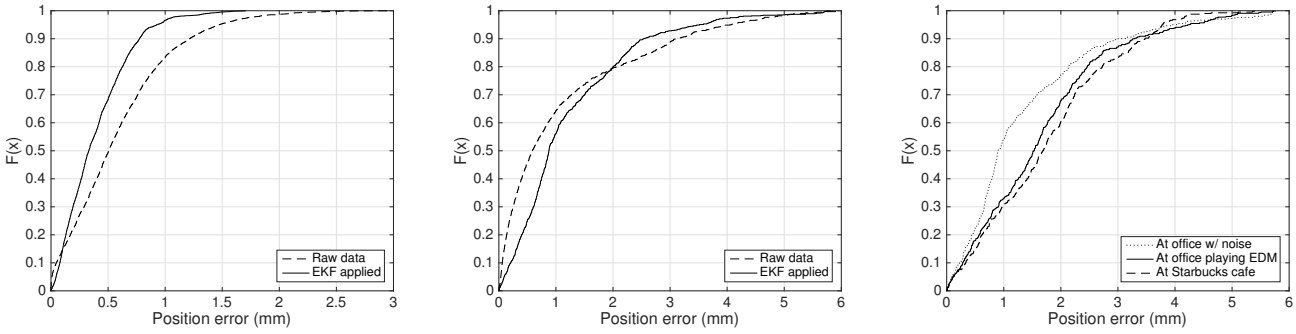
## 4.3 Noise Reduction

To adaptively handle the background noise and improve the positioning accuracy, we adopt noise reduction technique. First, we apply band-pass filter to the frequency-domain audio signal to remove noise at unintended frequencies. As we use the audio signal at $19kHz$ frequency, we pass the frequency band between $18.5kHz$ and $19.5kHz$, leaving upper and lower $500Hz$ band for FDoA scheme. Applying the band-pass filter effectively blocks most of the background noise and leaves the frequency band sent by the transmitter. However, when there are high frequency noises such as machine sound and fan noise, the band-pass filter performs poorly and degrades the positioning performance. Figure 8 illustrates the frequency-domain audio captured at a cafe. It contains high frequency noise around $19kHz$, which significantly interferes




(a) Measured angle (b) Measured range (c) Position error

**Figure 10: Summary of (a) angle and (b) range estimation results. (c) Angle and range measurements presented using the polar coordinate systems. All error bars stand for standard deviation.**




(a) Static sound source (b) Moving sound source (c) Moving with background noise

**Figure 11: CDF of position errors in various operation scenarios.**

the detection of our audio signal from the peripheral. Therefore, we implement a noise reduction technique to extract clean original signal.

When the system starts, we record the ambient sound for a short period of time and build a noise model for current background noise. At this time, the peripheral remains silent. Using the noise model, the receiver performs spectral subtraction to the incoming audio when the peripheral starts to emit the signal. Finally, the original audio signal is reproduced after the phase recovery and transforming the frequency-domain signal to time-domain signal. The noise reduction procedure is shown in Figure 9.

# 5. PERFORMANCE EVALUATION

## 5.1 Experiment Setup

We use two off the shelf mobile devices without any modification on hardware for evaluation. We adopt a device with large screen with high screen-to-body ratio as a receiver: a Samsung Galaxy Note 4 which has 5.7 inch display with 1440×2560 pixels on a 153.5×78.6×8.5*mm* body. Microphones 1 and 2 are aligned horizontally, while microphones 2 and 3 are aligned vertically along the bottom edge with 33*mm* spacing as shown in Figure 1. The mobile device continuously displays a cursor on the screen, which tracks the sound source.

The audio-signal transmitter, Samsung Galaxy S II, has a speaker at the backside of device. To enhance the usability of the transmitter, we design a pen-type peripheral device that embeds a small speaker at the tip of the pen, and prototype it using a 3D printer. It leverages a 16×12*mm* speaker with 1.2*W* rated output power, which is adopted by modern devices including Samsung Galaxy S7. We redirect the output wire of an embedded speaker of the mobile device, which repeatedly generates audio signals and physically connect the output with the peripheral.

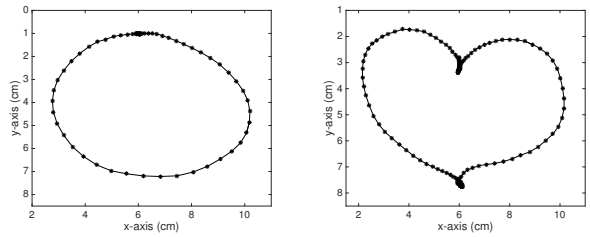
We first run experiments in a quiet office room and study the impact of background noise later. We perform operations on a spacious desk with any objects on it in order to minimize the multi-path effects. Our evaluation covers tracking over 5000 of movements composed of straight lines and curves.

## 5.2 Static Sound Source Localization

We analyze the localization performance when the sound source is static. Position error is defined as the Euclidean distance between the estimation result and the ground truth position of the grid.

We first measure the angle and range error of the estimated value. $\theta$ denotes the angle between the receiving device and the sound source, as illustrated in Figure 4. This angle increases from 20° to 70° in 10° intervals. Range $r$ signifies the displacement of the sound source from microphone 1, assuming that the microphone is located at the corner of the device. The range varies from 6*cm* to 10*cm*, in 2*cm* intervals.

Figure 10(a) plots the measured angle on the designated angle and the range position in the measurement space, where the error bars stand for the standard deviation of the angle error. The average angle




(a) Circle (b) Heart

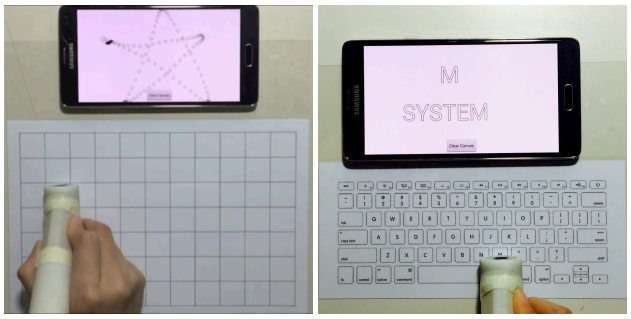
(c) Ribbon (d) Star

**Figure 12: Movement trajectories of vTrack.**

error is 0.8°, with a standard deviation of 1.0°. The angle error as well as the standard deviation tends to increase when the angle gets smaller. This is because the effect of refraction on sound propagation increases when the sound source is located at grid points with a small angle. With regard to the measurement range, the average angle error is high (1.2°) in the short range and decreases as low as 0.5° in a longer measurement range. Similarly, Figure 10(b) shows the measured range. The range error also tends to increase with small angles, but the estimation is quite accurate, which shows only a 1.1*mm* error on average with a standard deviation of 1.0*mm*.

Figure 10(c) illustrates the angle and range estimations using the polar coordinate system, where the reference point (0,0) is assumed to be the location of microphone 1 in our scenario. The Cartesian coordinates are also transcribed in parallel for reference purposes. The gray area represents the space where the virtual trackpad targets its operations. We design the dimension of the trackpad to be identical to the actual screen of the receiving device. Due to the bezel on the device's short edges, there is a 1*cm* space on the x-axis. The measurement result in the black circle is illustrated with its ground truth measurement point in the white circle, and measurements on the same angle are connected with a white solid line. The error bars stand for the standard deviation of the position error. The measurement result becomes more accurate as the range increases. In the short range, the space between each ground truth is narrower than in the longer range, which accentuates the position error.

Figure 11(a) shows the distribution of the position error on the grid: the average position error is 0.7*mm* before applying the extended Kalman filter. Specifically, 83% of raw measurements have less than a 1.0*mm* error. We also measure the position estimation using the extended Kalman filter algorithm and compare the result with the raw data. Because the filter reflects possible errors that may exist in measurements and applies more weight to values with more certainty, it contributes to reducing the position error to 0.3*mm* on average, with small standard deviations of 0.2*mm* as well, as shown in the CDF. Over 96% of estimations have errors smaller than 1.0*mm*, which surpasses the theoretical granularity 1.8*mm*. By applying the filter, we achieve a 57% of improvement in the localization performance.




(a) 2× size trackpad (b) Keyboard input

**Figure 13: Two application scenarios using vTrack.**

## 5.3 Moving Sound Source Localization

We next measure the performance of localization under the moving sound source scenario. We collect the estimation of the x- and y-coordinates while drawing various paths on the grid space. Four example results are illustrated in Figures 12. To derive the position estimation error when tracking the moving source, we compute the shortest distance from each estimation point to the corresponding ground truth path.

Figure 11(b) presents the error distribution of the moving source. The average position error for the raw data case is 1.2*mm*, with a standard deviation of 0.8*mm*. It is a slight decrease in the performance compared to the static scenario, which is expected. After applying the extended Kalman filter, the position error decreases to 1.1*mm* on average with a smaller standard deviation. Overall, the filter improves the accuracy with measurements that have large error, but slightly degrades the measurements with small error as well. Our experiments demonstrate that the proposed moving source localization algorithm can accomplish millimeter-level accuracy, which is sufficient for tracking the minute movements.

To evaluate the usability of vTrack in an environment with background noise, we perform experiments in various locations: an office room with general noise, an office room playing Electronic Dance Music (EDM), and a Starbucks cafe. General noise in an office room includes human voice and ceiling fan noise, while playing EDM adds high-frequency electronic music sound on it. We find that a cafe has all the aforementioned conditions including high-frequency machine sound, which overlaps with our audio signal and interferes the signal detection. Figure 11(c) shows the position error distribution with different background noise. Compared to the previous result performed in a quiet office room, the performance difference is marginal. Average error remains 1.3*mm*, 1.6*mm*, and 1.7*mm*, respectively. We confirm that our noise reduction algorithm performs well in environments with high-frequency noise.

**2× trackpad:** To evaluate the impact of the size of workspace on the tracking accuracy, we expand the size of virtual trackpad to 2 times larger (i.e., 20×26*cm* grid) than the one used in the previous experiments. The width of the workspace exceeds the distance between microphones 1 and 2 as shown in Figure 13(a). Average accuracy of vTrack with 2× trackpad is 1.5*mm*, where the degradation of it performance is negligible. The results indicate that vTrack can be used on a large surface near the receiving device, and the workspace is not limited to the distance between two microphones.

**Keyboard input application:** To evaluate the usability of vTrack as an input interface, we design a keyboard application that tracks the sound source and recognizes the key a user selects. We use

the Apple keyboard printed on $18\times8cm$ paper, which is 70% of its original size. Each key is a $1cm\times1cm$ square with $2mm$ spacing exists between each other. A short pause on a key generates repeated position of the key and delivers the user's intention. We perform 100 trials of experiment with a large number of words with both densely and sparsely populated keys, and achieve 95% of key recognition accuracy using vTrack.

# 6. CONCLUSION

In this paper, we propose a virtual trackpad interface by applying the sound source localization technique in the context of mobile systems. vTrack utilizes a minimal set of low-cost sensors on the off-the-shelf mobile devices, including a speaker on the peripheral side and a multi-channel microphone array on the receiving device. Our experiments show that vTrack achieves a reasonable positioning accuracy in both static and moving sound source scenario. We believe that vTrack's positioning performance is promising enough to demonstrate the potential of acoustic-based mobile interaction interfaces. Our ongoing work is to extend vTrack to a three-dimensional motion tracking system using the acoustic signal.

# 7. REFERENCES


[1] A comprehensive look at smartphone screen size statistics. https://goo.gl/7mX2DR.

[2] Equil. http://www.myequil.com/home/.

[3] Livescribe. http://www.livescribe.com.

[4] Phree. http://otmtech.com.

[5] S. Agrawal, I. Constandache, S. Gaonkar, R. Roy Choudhury, K. Caves, and F. DeRuyter. Using mobile phones to write in air. In *Proceedings of the 9th International Conference on Mobile Systems, Applications, and Services*, MobiSys '11, pages 15–28, New York, NY, USA, 2011. ACM.

[6] M. Brandstein and H. Silverman. A robust method for speech signal time-delay estimation in reverberant rooms. In *Acoustics, Speech, and Signal Processing, 1997. ICASSP-97., 1997 IEEE International Conference on*, volume 1, pages 375–378 vol.1, Apr 1997.

[7] L. Girod, M. Lukac, V. Trifa, and D. Estrin. The design and implementation of a self-calibrating distributed acoustic sensing platform. In *Proceedings of the 4th International Conference on Embedded Networked Sensor Systems*, SenSys '06, pages 71–84, New York, NY, USA, 2006. ACM.

[8] A. Harter, A. Hopper, P. Steggles, A. Ward, and P. Webster. The anatomy of a context-aware application. In *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking*, MobiCom '99, pages 59–68, New York, NY, USA, 1999. ACM.

[9] C. Knapp and G. Carter. The generalized correlation method for estimation of time delay. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 24(4):320–327, Aug 1976.

[10] H. Liu, Y. Gan, J. Yang, S. Sidhom, Y. Wang, Y. Chen, and F. Ye. Push the limit of wifi based localization for smartphones. In *Proceedings of the 18th Annual International Conference on Mobile Computing and Networking*, Mobicom '12, pages 305–316, New York, NY, USA, 2012. ACM.

[11] J. Liu, Y. Wang, G. Kar, Y. Chen, J. Yang, and M. Gruteser. Snooping keystrokes with mm-level audio ranging on a single phone. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, MobiCom '15, pages 142–154, New York, NY, USA, 2015. ACM.

[12] C. Peng, G. Shen, Y. Zhang, Y. Li, and K. Tan. Beepbeep: A high accuracy acoustic ranging system using cots mobile devices. In *Proceedings of the 5th International Conference on Embedded Networked Sensor Systems*, SenSys '07, pages 1–14, New York, NY, USA, 2007. ACM.

[13] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. The cricket location-support system. In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, MobiCom '00, pages 32–43, New York, NY, USA, 2000. ACM.

[14] J. Qiu, D. Chu, X. Meng, and T. Moscibroda. On the feasibility of real-time phone-to-phone 3d localization. In *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems*, SenSys '11, pages 190–203, New York, NY, USA, 2011. ACM.

[15] Z. Sun, A. Purohit, K. Chen, S. Pan, T. Pering, and P. Zhang. Pandaa: physical arrangement detection of networked devices through ambient-sound awareness. In *Proceedings of the 13th international conference on Ubiquitous computing*, pages 425–434. ACM, 2011.

[16] J. Wang, D. Vasisht, and D. Katabi. Rf-idraw: Virtual touch screen in the air using rf signals. In *Proceedings of the 2014 ACM Conference on SIGCOMM*, SIGCOMM '14, pages 235–246, New York, NY, USA, 2014. ACM.

[17] J. Wang, K. Zhao, X. Zhang, and C. Peng. Ubiquitous keyboard for small mobile devices: Harnessing multipath fading for fine-grained keystroke localization. In *Proceedings of the 12th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '14, pages 14–27, New York, NY, USA, 2014. ACM.

[18] B. Xu, R. Yu, G. Sun, and Z. Yang. Whistle: Synchronization-free tdoa for localization. In *Distributed Computing Systems (ICDCS), 2011 31st International Conference on*, pages 760–769, June 2011.

[19] C. Xu, P. H. Pathak, and P. Mohapatra. Finger-writing with smartwatch: A case for finger and hand gesture recognition using smartwatch. In *Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications*, pages 9–14. ACM, 2015.

[20] J. Yang, S. Sidhom, G. Chandrasekaran, T. Vu, H. Liu, N. Cecan, Y. Chen, M. Gruteser, and R. P. Martin. Detecting driver phone use leveraging car speakers. In *Proceedings of the 17th Annual International Conference on Mobile Computing and Networking*, MobiCom '11, pages 97–108, New York, NY, USA, 2011. ACM.

[21] C. Zhang, J. Tabor, J. Zhang, and X. Zhang. Extending mobile interaction through near-field visible light sensing. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, MobiCom '15, pages 345–357, New York, NY, USA, 2015. ACM.

[22] Z. Zhang, D. Chu, X. Chen, and T. Moscibroda. Swordfight: Enabling a new class of phone-to-phone action games on commodity phones. In *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services*, MobiSys '12, pages 1–14, New York, NY, USA, 2012. ACM.