AMIL: Localizing Neighboring Mobile Devices Through a Simple Gesture

Hao Han, Shanhe Yi†, Qun Li†, Guobin Shen‡, Yunxin Liu‡, Ed Novak†
Intelligent Automation, Inc., †College of William and Mary, ‡Microsoft Research Asia
hhan@i-a-i.com, †{syi,liqun,ejnovak}@cs.wm.edu, ‡{jackysh,yunliu}@microsoft.com

***Abstract*—Smartphone users are often grouped to exchange files or perform collaborative tasks when meeting together. We argue that the location information of group members is critical to many mobile applications. Existing localization solutions mostly rely on anchor nodes or infrastructures to perform ranging and positioning. These approaches are inefficient for ad hoc scenarios. In this paper, we propose AMIL, an Acoustic Mobility-Induced TDoA (Time-Difference-of-Arrival)-based Localization scheme for smartphones. In AMIL, a smartphone user can use simple gestures (e.g., hold the phone and draw a triangle in the air) to quickly obtain the relative coordinates of neighboring mobile devices. We have implemented and evaluated AMIL on off-the-shelf smartphones. The field tests have shown that our scheme can achieve less than three degree orientation errors and can successfully build a simple map of 12 people in an office room with average error of 50cm.**

I. Introduction

Smartphone users are often grouped to exchange files when meeting together. They may not know each other in advance, so have no prior knowledge about each other's names. They would like, however, to exchange information such as contact information or share electronic documents during the meeting. To achieve that goal, mobile users often use their smartphones to set up local networks (e.g., Wi-Fi or Bluetooth), where inter-connected devices can communicate with each other. In such networks, the ad hoc pattern increases the demand for more intuitive ways to identify communication parties. For example, Bluetooth can list neighboring devices and display their names. Yet the user may not easily link the name to individual communication party, because there exists a perception gap between the digital world and the physical world. We argue that location information can help bridge this gap and enable a more intuitive method of sharing. Suppose all neighboring devices can be displayed on a map according to their relative positions, a user can easily share information with selected targets (see Fig.1). It should be noticed that existing NFC technology has provided an intuitive way to share files. However, in order to share with many users, touching each other's devices is inefficient.

In this paper we design and implement *AMIL*: an Acoustic Mobility-Induced Localization scheme for smartphones with a requirement of only a set of common hardware: a speaker, a microphone, and Inertial Measurement Units (IMUs). AMIL allows a mobile user to quickly locate other devices in proximity with little configuration overhead. Unlike conventional localization approaches, AMIL offers a fast *one-to-many* localization scheme: a device can perform AMIL to obtain the relative coordinates of all other devices without measuring the distance between them. Furthermore, AMIL is a pure software solution that does not rely on any dedicated hardware or modification to the operation system. It thus can be adopted across different embedded platforms with little deployment effort. Note that the intention of this work is to develop a working system that allows a single mobile user to obtain the basic positions of others so as to distinguish them efficiently Our work does not aim at inventing a more accuracy localization algorithm.

In the literature, researchers from Microsoft are among the first to use speaker/microphone for acoustic ranging on mobile devices with BeepBeep [1]. BeepBeep uses a TDoA (Time Difference of Arrival)-based approach to measure the distance between two devices only. Although mutual distances can be further used to determine the relative positions, it requires at least three anchor nodes to determine the location of a device. Aiming to support localization between two devices, recent work [2] extends BeepBeep, where each device must have at least two speakers and two microphones. By measuring the distances for all speaker-to-microphone pairs, two devices can thus locate each other. The major drawback of both approaches is that every device has to transmit beeps, so that the localization overhead is increased significantly with the number of devices. However, no matter how many devices are nearby, a fixed number of beeps are emitted by a single device in AMIL. AMIL induces the mobility of a single device to create arbitrary "virtual" anchors. In such a way, a single device can locate others more efficiently.

The basic idea of *AMIL* is as follows: A device (player) who attempts to locate other devices is moved in the air. During the movement, the internal accelerometer and gyroscope track the displacement at every instant. The player, meanwhile, emits audio beeps with a specific pattern. Other devices (listeners) passively use their microphones to listen each beep, digitize the sound signal, and compare it with the known pattern to confirm the arrival time. Due to the movement of the player, a listener at difference place is expected to measure different time intervals between beeps. The player can use these differences coupled with its motion trail to determine each listener's location. Again, AMIL does not intend to improve the accuracy of the state-of-the-art localization algorithm. Instead, AMIL offers a lightweight way to obtain the direction and distance of all the neighboring mobile devices.

978-1-4673-9953-1/16/$31.00 ©2016 IEEE

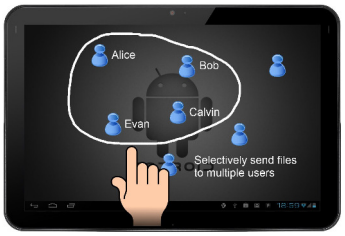

Fig. 1: Motivation scenario

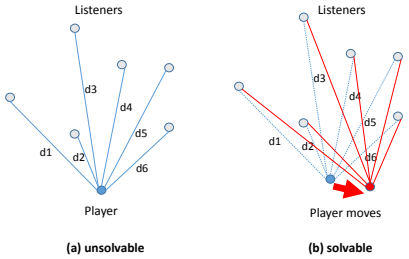

Fig. 2: Intuition of AMIL localization scheme

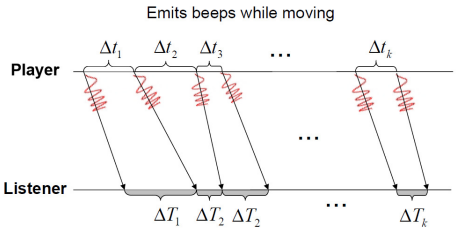

Fig. 3: Intuition of the difference of beep intervals between a player and a listener

The idea of mobility-induced TDoA in AMIL is not totally new, it has been proposed for localization in wireless sensor networks in [3]. This paper, however, has demonstrated for the first time that mobile devices can adopt this technique for localization with a set of common hardware and pure software implementation. Since the mobility of a phone is limited by a small area covered by (human) arm motion, it is very challenging to leverage phone's internal sensors to deliver an accurate localization when the mutual distance between phones are far away. Furthermore, the moving strategy has effects on the localization accuracy. Little work focused on this problem before. It is non-trivial to design a better moving gesture for practical use.

To the best of our knowledge, we are the first to combine internal motion sensors and acoustic techniques on smartphones for localization. The main contributions of this work are:

- AMIL is the first working system that allows a off-the-shelf smartphone to locate neighboring mobile devices easily. We have conducted extensive experiments to evaluate our system in a real-world environment.
- We characterize several new challenges of applying acoustic mobility-induced localization on smartphones, such as microphone's sample rate drift and moving strategy. We also give solutions to address each challenge.

II. System Design and Algorithm

In this section, we present the system design and algorithms of AMIL. We first describe the intuition behind why a single user (player) can locate others (listeners) and then elaborate the system architecture and each component.

A. Intuition

The intuition of our localization scheme is illustrated in Fig. 2. We assume that the player who attempts to locate several listeners is able to measure the distance towards each of them (e.g., d_1 to d_6). Yet only knowing the distance is not enough to locate each listener, because there is only one anchor point (shown in Fig. 2(a)). The listener can reside in any point on a circle centered on the player with the radius equal to the measured distance d_j. If we could create multiple anchor points, the problem is then solvable. To achieve this, we move the player and rely on IMUs to track the displacement (shown in Fig. 2(b)). Thus, multiple "virtual" anchors are created, and the position of each listener can be uniquely determined.

Actually, measuring the mutual distance (i.e., d_1 to d_6) is not necessary here. In our solution, the player can locate multiple receivers simultaneously by emitting a series of audio beeps without the necessity for each listener to reply the beep. Suppose the player emits $k+1$ beeps at intervals Δt_i for $i \in [1, k]$. After propagation, each listener will measure a set of beep intervals ΔT_i shown in Fig. 3. If the player and the listener are both stationary, we have $\Delta t_i = \Delta T_i$. Due to the spatial change when the player emits beeps, each listener will capture different beep intervals depending on its own location. For example, if the player moves towards a listener, the beep interval measured by this listener is less than that captured by the player. Based on the interval difference, we can derive the *delta distance* using the speed of sound as $c \cdot (\Delta T_i - \Delta t_i)$. After $k+1$ beeps, we have k delta distances. The player can finally use them to determine the position for each listener; the details of this will be elaborated later. The advantage of this approach is that only the player emits beeps, while listeners only passively listen for beeps. This approach is easy for any mobile user who wants to locate others to perform in practice.

B. Architecture

Fig. 4 depicts the system architecture of AMIL. In this architecture, a player uses its internal speaker to emit beeps with a pre-defined pattern, and its microphone to pick up the acoustic signals for measuring the beep intervals. The player also relies on IMUs (i.e., accelerometer and gyroscope), to estimate the displacement of the phone during the movement. At the same time, the listener just passively records sound from its microphone, measures the beep intervals, and exchanges this information with the player through other available communication channels such as Wi-Fi or Bluetooth.

There are three algorithms in AMIL – the *movement tracking algorithm*, the *interval calculation algorithm*, and the *positioning algorithm*. The movement tracking algorithm is used only on the player side to track the motion trail of the player and determine the displacement when each beep is played. The interval calculation algorithm is performed on both the player and the listener sides, aiming to detect the beeps and measure beep intervals. After that, each listener sends back the calculated beep intervals to the player. Once receiving the beep intervals from the listeners, the player will use the positioning algorithm to compute the relative coordinates for each listener. The detail of each algorithm is presented in the following subsections.

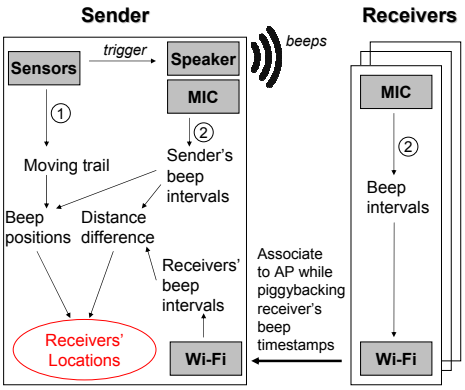

Fig. 4: System architecture

C. Movement tracking algorithm

A player leverages accelerometer and gyroscope to compute its displacement at every beep. The basic idea is to double integrate the accelerometer readings to derive displacement, where the first integration converts the acceleration into velocity, and the second integration converts the velocity into displacement. However, on-board motion sensors are not designed for precise tracking. Any small errors caused by noise, gravity and rotation will be accumulated into a significant drift. To minimize these errors, we apply the following methods.

Motion detection. In order to reduce the drift, it is important to decrease the integration period. Thus, we only integrate on the period when motion is detected. To detect motion, we separate sensor readings into bins and compute the *standard deviation (std)* of each bin. If the std for a bin is greater than a pre-defined threshold, the first reading in that bin is conservatively regarded as the start point of a movement. Given the start point, we marks the end point by checking if the deviations of two subsequent consecutive bins are less than the threshold.

Rotation transformation. An accelerometer records acceleration readings along the axes of phone's frame. During moving, the phone's frame may be rotated. Thus, sensor readings actually come from different reference coordinate systems. To address this problem, we leverage gyroscope data to perform rotation transformation. The gyroscope measures the instantaneous angular speed around phone's x, y, and z axes. We define the initial phone's frame is the reference frame. The output of the gyroscope is integrated over time to calculate the angle of rotation from the sampling instance to the initial orientation. Suppose at timestamp t_1, the angle changes on x, y, z axes are roll (ϕ), pitch (θ), and yaw (ψ) respectively. The rotation matrix is calculated as follows,

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\phi & -sin\phi \\ 0 & sin\phi & cos\phi \end{bmatrix} \begin{bmatrix} cos\theta & 0 & sin\theta \\ 0 & 1 & 0 \\ -sin\theta & 0 & cos\theta \end{bmatrix} \begin{bmatrix} cos\psi & -sin\psi & 0 \\ sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

The product of the rotation matrix and the accelerometer readings in the phone's frame yields the transformed acceleration.

Velocity compensation. When the phone is moved, it is natural to pause for a short time somewhere. For instance, if a user draws a triangle, natural pauses shall occur at three corners. During the pause, the phone is stationary and its speed is zero. However, the acceleration residues caused by gravity and the misalignment of internal sensors typically are integrated to a non-zero velocity at a stop. We need to compensate the velocity. First, we detect the pauses using the method mentioned in motion detection. At each pause, if the integrated velocity is not zero, all velocities integrated during the last movement are adjusted to the following value:

$$v'(t_j) = v(t_j) - v(t_k)\frac{t_j - t_i}{t_k - t_i},\ i \in [i, k]$$

where $v'(t_j)$ denotes the adjusted velocity at t_j, $v_(t_j)$ refers to the originally integrated velocity at t_j, and t_i and t_k represent the starting and ending time of motion. Using this method, constant gravity and sensor misalignment offset can be eliminated.

D. Interval calculation algorithm

Due to the uncertain delay between the time when a command is issued to emit a tone, and the instant when the tone is physically emitted by the hardware, the actual interval between two beeps on the sender side cannot be accurately determined directly from the timestamp in the software when the command was issued. To deal with this problem, we adopt a "self-recording" method similar to work [1], [2], [4], where the sender records itself using its own microphone when beeping. Since the distance between the sender's speaker and microphone never changes, the time interval between the two beeps captured by the microphone is exactly the same as they are physically emitted by the speaker. In practice, the beep signal should be designed carefully to cope with the following issues. First, the sound signal will be attenuated and distorted through the communication channel, and negatively affected by the ambient noise. Thus, the signal should be designed to have a good Signal-to-Noise Ratio (SNR) when reaches the receiver. Second, the signal should have a better resistance to *multi-path* and *non-line-of-sight (NLOS)* effects. Based on existing work [1], [2], [4], [5], we choose a linear chirp waveform to overcome these challenges.

To detect the beep signal and its arrival time, we adopt a common cross-correlation-based method used in previous mentioned work, where the emitted waveform is correlated with the received signal to determine when the beep is present in the received signals. Let the sample sequence $\{u_i\}$ for $i = 1, 2, \cdots, n$ denote the received samples from microphone, and the sample sequence $\{v_i\}$ for $i = 1, 2, \cdots, m$ represent the known, emitted waveform, where $n >> m$. In matched filtering, a sliding window with the length equal to m is extracted from $\{u_i\}$, and the sample correlation coefficient r is computed as follows:

$$r = \frac{\sum_{i=1}^{m}(u_i - \overline{u})(v_i - \overline{v})}{\sqrt{\sum_{i=1}^{n}(u_i - \overline{u})^2}\sqrt{\sum_{i=1}^{n}(v_i - \overline{v})^2}},$$

where $\overline{u}$ and $\overline{v}$ are the sample means of the sliding window and $\{v_i\}$ respectively. For each r, the computational complexity

$\Theta(m)$ can be achieved by Welford's one-pass algorithm. As the sliding windows moves from the begin to the end of $\{v_i\}$ sample by sample, each r is computed with the total computation complexity equal to $\Theta(n \cdot m)$. A large value of r means a high similarity between the two sequences. The arrival instance of each beep (in terms of sample index) can be found at the maximal peak of these coefficients. After obtaining the index of two beeps, the beep interval is then calculated by the number of samples between two beeps divided by the sampling rate (44.1 kHz in our implementation). It is worth noting that owing to the multiplicity of paths, the sound signal arriving via the shortest path may be weaker than that of the reflections. This situation is even worse in NLOS scenarios because the direct path signal has to traverse through obstructions. As a result, the maximum correlation peak may not represent the Line-of-Sight distance. To address this problem, we adopt an existing technique, in [6], that finds the first sidelobe that exceeds a threshold ratio to the maximum peak.

FFT-based acceleration for beep detection. The correlation is computationally expensive especially when n is large. Some works(e.g., [1] and [6]) offload the computation to a powerful cloud server or leverage parallelism of GPU hardware to accelerate the process. Such approaches work but at the expense of extra communication overhead or power consumption, because the computational complexity is still $\Theta(n \cdot m)$. In recent work, Qiu et al [2] applied an energy threshold to reduce the search space in the sample sequence. However, setting such a threshold is challenging due to the dynamics of ambient noise. If set too low, the computational overhead won't save much. In contrast, if set too high, some beeps may be lost. To overcome this issue, Zhang et al. [4] used auto-correlation to quickly estimate the rough position of a beep and then apply cross-correlation to identify the exact position. In their approach, a beep is composed of two same sequences: one followed by the other. If a sequence of audio samples is found to be very similar to its half-lag (-shift) sequence, a beep is detected. Auto-correlation can be computed fast but inherently has wide peak, so it is difficult to detect the position of a beep accurately. That is why cross-correlation is used in the second stage. However, this method cannot be applied to any scenario with highly self-correlated ambient noise, such as air conditioner. In AMIL, we propose a fast Fourier transform (FFT) based filter to reduce the search space. First, the emitted chirp sound is converted to the frequency domain by FFT. At the same time, the spanning frequency is recorded. Next, the received sequence is divided into equal blocks with size equal to that of the chirps. In each block, an FFT is computed to find whether the similar spanning frequency is detected with high energy. If true, a beep may reside in the block and cross-correlation is calculated on the samples in recent two blocks. Otherwise, we skip it to save the computation of cross correlation. We include the previous block conservatively because a beep sequence may be present in two consecutive blocks and we want to find the exact start point, which may reside in the first block. Suppose N beeps are emitted, each with length equal to m, and the received signal has n samples in total. To detect the position of every beep, our approach needs about $\Theta(n \log m)$ time to process FFT plus $\Theta(m^2)$ time to calculate cross-correlation. Later, in the evaluation section, we will use experiments to show the efficiency of this method.

E. Positioning Algorithm

Since three beeps are the minimal requirement to position a device in a 2D plane, we first present the positioning algorithm for three beeps and then generalize it to more than three beeps.

Positioning algorithm for three beeps. Suppose a receiver is located at the coordinates (x, y). The coordinates of three beeps are $(0, 0)$, (x_1, y_1), and (x_2, y_2) respectively. The distance difference derived from beep intervals on both the sender and receiver side are denoted as dd_1 and dd_2. Thus, we have the following equations:

$$\begin{aligned} dd_1 &= \sqrt{(x - x_1)^2 + (y - y_1)^2} - \sqrt{x^2 + y^2} \\ dd_2 &= \sqrt{(x - x_2)^2 + (y - y_2)^2} - \sqrt{x^2 + y^2} \end{aligned} \quad (1)$$

Combining the above two equations produces a system of linear equations for x and y in terms of $Ax + By = C$, where A, B and C are all constant variables given (x_1, y_1), (x_2, y_2), dd_1 and dd_2. Either x or y can be expressed by the other variable and substituted back into Eq.(1) to derive the closed-form for x and y using (x_1, y_1), (x_2, y_2), dd_1 and dd_2.

The *triangle inequality* states that the difference of lengths of any two sides must be less than the length of remaining side. That means the absolute values of dd_1 and dd_2 must be less than the moving lengths $\sqrt{x_1^2 + y_1^2}$ and $\sqrt{x_2^2 + y_2^2}$ respectively. In practice, due to measurement errors dd_i may be slightly larger the $\sqrt{x_i^2 + y_i^2}$ or less than $-\sqrt{x_i^2 + y_i^2}$ where $i \in 1, 2$. In both cases, we cannot solve the equations, though the approximate direction of receivers can be inferred. In the former case, the receiver(s) is likely to be located somewhere in the movement direction. In the latter case, the receiver(s) may be located in the opposite of movement direction.

Solving Eq.1 may produce one, two or zero solutions in practice. If two solutions exist, additional information is needed to select one point. For example, in a meeting room where all people are sitting around a table and everybody is in front of the player, the solution that indicates the receiver is behind the player should be ignored. For the zero solution case, more beeps are necessary.

Positioning algorithm for more beeps. In general, extra beeps can improve the positioning accuracy. Let the coordinates of the first beep be $(0, 0)$ and (x_i, y_i) $i = 1, 2, 3, \cdots$ for other beeps. We have an equation for each delta distance as: $dd_i = \sqrt{(x - x_i)^2 + (y - y_i)^2} - \sqrt{x^2 + y^2}$. Similar to Eq. 1, combing every dd_i and dd_1 yields a system of linear equations in two variables x and y. Suppose there are n beeps. All of $n - 2$ linear equations can be expressed in matrix form

$$\begin{bmatrix} A_1 & B_1 \\ A_2 & B_2 \\ \cdots & \\ A_{n-2} & B_{n-2} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} C_1 \\ C_2 \\ \cdots \\ C_{n-2} \end{bmatrix}.$$

As long as $n \geq 4$, variables x and y can be solved without substituting them back into the equation of dd_i. The least squares method can be used to find an approximate solution to the overdetermined system. The point derived by such a method has minimal distance to all the curves. However, according to our experience, least squares method does not work well in practice, since some curves are noisy, due to measurement errors. Including these curves for calculation may degrade the overall accuracy. Our method is to select a set of three beeps from n beeps. The total number of sets is equal to $\binom{n}{3}$. For each set, the positioning algorithm for three beeps is applied to extract possible solutions. The direction of those points to the origin is then calculated. The majority of points within a certain angle (10° is used in the current implementation) are kept and others are filtered out as outliers. Finally, the geometric centroid of all the remaining points is used as the location of the target. Although the localization accuracy may be slightly deteriorated when a very accurate solution is averaged by several less accurate solutions, this method can output a good solution for general cases and is therefore more robust.

III. Moving Strategy

There are two error sources that may affect the accuracy of AMIL. The first source is in the acoustic subsystem. Due to the effects of multi-path and non-light-of-sight (NLOS) in practice, the arrival of a beep may not be precisely detected. Thus, the beep interval is calculated with errors. The second source of errors are from the IMU subsystem. As mentioned before, sensor noise, gravity and phone's rotation will lead to several centimeter errors of estimating the displacement of the phone. All these measurement errors will contribute to the inaccuracy of positioning results. In previous sections, we use various methods to minimize these errors. This section will focus on the impact of moving strategy on the accuracy.

To simplify the analysis, we assume a sender simply moves along a segment and beeps at two endpoints. The angle between a listener's direction and the direction of movement is defined as θ. Due to the measurement errors, the estimated location of the listener will depart from the actual place. According to our simulation, with bounded value of errors, all the estimated locations will reside in a sector which also includes the actual location. We define α as the angle of this sector. The smaller α, the better. We seek to find a better moving strategy (i.e., θ) that can yield a small α. In simulation, we enumerate every possible errors within the bound for a θ. Fig. 5 depicts the relation between θ and α, where the value of θ ranges from -180° to 180°. The positive angles represent clockwise rotation of θ, while the negative angles refer to counter-clockwise rotation. It is seen that if the player is moving towards or away from the listener, the corresponding α is the largest. However, if the direction of movement is perpendicular to the direction of the listener, the smallest α can be achieved. Therefore, our first guideline for a better moving strategy is that *it is better to move perpendicular than parallel to the direction of the target.*

From Fig. 5, it seems that the positive θ and negative θ have the same impact on α. In fact, if we further divide the sector into the left and right parts based on the actual location of the listener (i.e., the estimated location falling in the left part is defined as negative error, and the estimated locations in the right part refers to positive error), we can tell the difference. Fig. 6 shows the percentage of positive and negative errors against θ. It is seen that when θ is negative, errors are prone to be positive; otherwise, they are prone to be negative. If overlapping the positive and negative errors, the final localization can be improved. Therefore, our second guideline for movements is *if moving more than once, the directions of the movements should flank the the target.*

Based on our findings, if the location of the target is known in advance, the best movement strategy is shown in Fig. 7(a). However, that strategy may result in a poor estimation for a target whose location is approximately parallel to the direction of movement like Fig. 7(b). Therefore, without knowledge of target's location, a triangle is the best moving strategy to achieve good performance on average. It ensures that we can find at least two sides of the triangle that are not parallel to the direction of target (see Fig. 7).

The performance of AMIL is not very sensitive to the shape of triangle according to our real experiments. Users are free to draw any triangle. To further improve the robustness of our localization algorithm, we design a new moving strategy to draw a triangle: starting from the middle of the bottom side, moving to the left corner, to the top corner, to right corner, and then back to the origin. This strategy combines the benefits of both the line and triangle gestures. We evaluate this triangle gesture in comparison to the line gesture in Section V.

IV. Implementation

We have implemented AMIL on Android devices, including Galaxy Nexus, HTC EVO 3D, Nexus S and Galaxy S2 with Android version 4.x. According to previous research [1], [5], [6], we choose a *linear chirp* as the ranging signal, because linear chirps offer good pulse compressibility and increased signal-to-noise ratio (SNR). To minimize audible artifacts, we set the frequency band of the linear chirp as [18kHz,22kHz], and modified the waveform with fade-in and fade-out pattern. The duration of the chirp is 50 ms. The sampling rate of microphone is configured to 44.1kHz. During the implementation of AMIL, we encountered two major difficulties as follows.

Sampling drift. Previous work always assumes that the microphone hardware could generate samples at the given sampling rate. According to our observations, *sampling drift*, however, occurs among different smartphones. To demonstrate the drift, we investigated the smartphones listed in Table I. A Galaxy Nexus phone was set up as a player (i.e., the reference), and other phones acted as listeners. The player emitted two beeps with intervals ranging from 0.5s to 2.5s. Meanwhile, the listeners measured the beep intervals in terms of microphone samples. It is expected that the samples captured on the listener side should be equal to that on the player side, because no one moved. However, we indeed captured different number

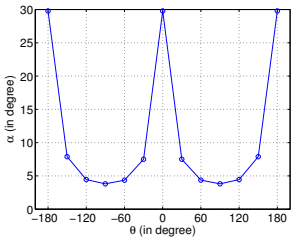

Fig. 5: Relation between θ and α

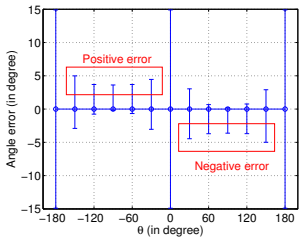

Fig. 6: Distribution of angle errors against θ

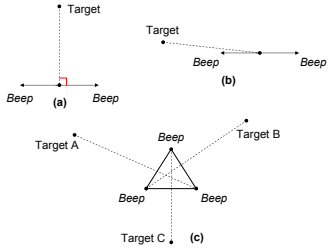

Fig. 7: Exemplar moving strategies

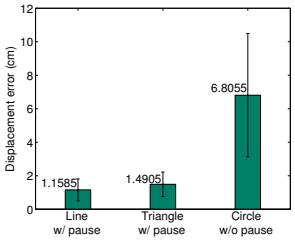

Fig. 8: Comparison of different gestures with or without pause

of samples on these devices. Table I presents the results, where the positive values means more samples were captured by listener than the player, and the negative values means the opposite. It is seen that 1) the same model platforms (two Galaxy Nexus phones) barely have sampling drift, and 2) the drift on different models increases linearly over time. The experiment results are quite stable for 10 tests and also exist when we changed the role for each phone. It should be noticed that each sample difference will lead to around 0.8cm ranging error given the sampling rate is 44.1kHz. In that case, 2.5s will bring about 16cm ranging error between Galaxy S2 and Galaxy Nexus. To cope with this problem, we build a linear model to calibrate each phone based on our extensive experiment results. During the process of localization, this model was used to compensate the sample drift for the second algorithm shown in Figure 4.

TABLE I: Sampling drift under different durations

Galaxy Nexus (ref)	0.5s	1.0s	1.5s	2.0s	2.5s
Galaxy Nexus	0	0	0	0	0
HTC EVO 3D	3	5	7	9	11
Nexus S	2	3	5	7	9
Galaxy S2	-4	-8	-12	-16	-20

Beep timing. It is critical to decide when to beep. Through study we found it is best to play a beep when the movement pauses. The reasons are in two aspects. First, Doppler effects will shift the frequency band of the emitted beeps and make the detection inaccurate. Playing beeps when the phone is stationary can mitigate this effect. Second, it is difficult to synchronize the sensor and microphone. In other words, it is not easy to find the precise location of the phone when a beep is physically emitted. However, if the beep is only emitted when the phone is stationary, we have more constraints to improve the accuracy.

V. Evaluation

In this section, we evaluate the performance of AMIL by answering the following questions: 1) What is the accuracy of our motion tracking algorithm; 2) What is the accuracy of our localization algorithm with respect to both direction and ranging errors; and 3) What is the on average computation time to locate other devices.

A. Accuracy of IMU sensors

We investigated the accuracy of different gestures including *line*, *triangle* and *circle* with or without pauses during the movement. To obtain the ground truth, we first drew a trail (i.e., line, triangle and circle) on the desk as the reference, and then moved the phone exactly following the trail. For a line, the estimated location of the ending point was compared to the reference. For a triangle and a circle, we compared three corners and quadrant points, respectively. The experiment was repeated 30 times. Fig. 8 show the averaged results with the standard deviation (std) as error bars. It is seen that the line gesture has the smallest estimation error which is slightly less than the triangle gesture and significantly less than the circle gesture. It concludes that pausing can significantly improve the displacement estimation. Therefore, we use simple gestures with natural pause, namely line and triangle for rest experiments.

B. Accuracy of determining direction

Line gesture. A line gesture with the length equal to 40cm was first tested for localization. Given the starting point at (0,0), we drew the line from the middle of the line, first to the left endpoint (-20,0) and then to the right endpoint (20,0). The reason why we drew the line in such a way is to make three beeps at each point. Receivers were placed along a line which is perpendicular to the moving line with 41cm away from each other. Experiment at each location was repeated five times. Fig. 9 shows the results. We found that the average direction errors were within 2.5° and the accuracy decreased when the receivers departed from the center. This is consistent with the the analysis in Section 4. Note that we also tested the cases of angles greater than 40 degree, but sometimes we could not find any solution, hence the results were not plotted in the figure. It confirms our guidelines in Section III.

Triangle gesture. Next, a triangle gesture with five beeps was tested. The motion trail was kept the same for all experiments from the coordinates (0,0) to (-20,0) to (0,20) to (20,0) and then back to (0,0). We use Cartesian coordinate to mark locations of player and listeners. The unit of distance is centimeter (cm). We investigated several locations where a line gesture cannot work well, where receivers were set to the coordinates (-264,-366), (-264,-244), (-264,-122), (-264,0), (-264,122), (-264,244) and (-264,-366) respectively. Again, each location was tested for 5 times. Fig. 10 shows the results. It is seen that the average direction errors are less than 6 degree. It confirms that the triangle gesture performs well for some places that the line gesture cannot work.

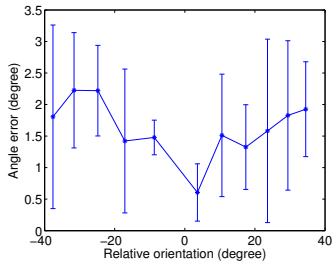

Fig. 9: Angle errors with line gesture given targets are located in the left region

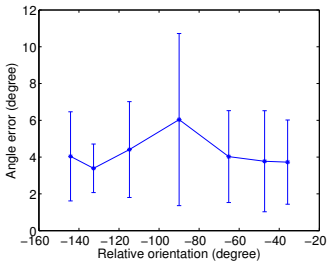

Fig. 10: Angle errors with triangle gesture given targets are located in the left region

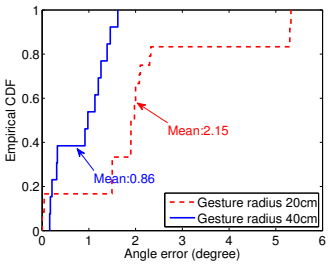

Fig. 11: Angle errors against different moving radius

Moving radius. We also tested the same triangle gesture but with different radius. In this experiments, receivers were placed in the center of a sender. First, the player is moved the coordinates (0,0) to (-20,0) to (0,20) to (20,0) and back to (0,0). Next, the moving radius increase from (0,0) to (-40,0) to (0,40) to (40,0) and back to (0,0). Both experiments were repeated 10 times. Fig. 11 depicts the CDF of angle errors, in which a gesture with large radius has better accuracy.

C. Accuracy of determining position

To evaluate the accuracy of determining positions, we conducted the following experiments in the hallway of our department building. The player was moved from (0,0) to (-30.5,0), to (0,30.5), to (30.5,0) and back to (0,0), and four listeners were placed from (0, 122) to (0, 488) each spaced by 122cm. The experiments were tested for 10 times. Since there are two types of error sources as mentioned in Section III, we decided to investigate them separately. First, we assumed that the motion trail was given in advance. In this way, we can ignore the sensor errors and only estimate how much the acoustic errors affect the localization accuracy. Next, we redid the experiment with inertial sensors enabled to estimate the displacement of our gesture. Last, we performed the same test by drawing three triangles to improve the performance.

The results are shown in Fig. 12. From this figure, we find that both errors of inertial sensor and beep detection contribute to the localization errors. And it shows that the displacement error adds more variations to the results. As the distance between sender and receiver increases, both the mean error and std becomes larger. For example, in (0,122) case, the biggest error is within 30cm while in (0,488) we have an error of almost 1 meter. By using three triangles, we can limit the error to less than 50cm for all cases. Note that, except for the 488cm case, all other three have an error of 30cm.

D. Field test

To measure the overall accuracy of our system, we conducted field tests in a 3m x 5m hallway. One player and twelve targets (listeners) are located in a 2D plane. All the listeners formed a grid with its cell size equal to 122 cm. The origin of the coordinate system is the player. The listeners are marked as in Fig. 13. Based on the previous discussion, we tested a small-sized triangle starting from the origin and ending to the same point. The side length of the triangle is 30.5cm.

Fig. 13 shows the visualized localization results. In this figure, the actual location of each listener is denoted by a unique solid and bold symbol. The estimated location is plotted in the same symbol to indicate its relationship with the ground-truth location. It is seen that when the listener is near the player, all the estimations are close to the ground-truth locations. It confirms our previous measurement observations that AMIL can estimate angle within 3 degree errors and differentiate targets accurately. However, when the distance between listeners and the player are far (e.g., listeners on the third row and fourth row), the accuracy decreases. There are more overlapping results, but we can still differentiate targets excepts that in the (-122,488) case.

To further improve the accuracy, we can use our localization scheme in multiple rounds. Fig. 14 depicts the results when a triangle was drawn three times. The maximal error is approximately 50cm. From this figure, we can easily differentiate all the targets, because there are no overlapping results.

E. Computation time

The finishing time of the localization process consists of two main parts: the duration of movement and computation time. When phone is moving, samples collected from microphone and IMU sensors are stored in memory or disk (when memory is full). The entire duration varies according to the shape and radius of the gesture, but it typically can be finished within 5s. The computation process is to calculate the moving trail from sensor readings, beep intervals from microphone readings, and the coordinates of receivers afterwards. To compare with full cross-correlation method, we only investigated the duration for computing beep intervals. Given all samples are in memory, the full cross-correlation method is first tested then followed by our method. Table II lists the results. It is seen that the FFT-based correlation method used in AMIL can reduce the finishing time by more than 90%.

F. Other considerations

Due to space limit, we briefly discuss other practical considerations in our system implementation.

Noise: We tested if our scheme can resist to ambient noise through playing different videos in the process of localization. AMIL can give accurate results most of time. However, in some very noisy cases, the starting point of the beep can be buried in random noise and results in big errors even though

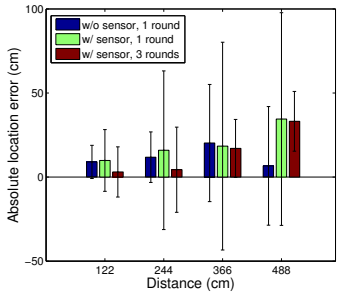

Fig. 12: Localization error using triangle gesture of 30.5cm radius. "w/o" means the exact beep locations are given, while "w/" means using sensor-estimated locations. The number of rounds counts the gesture repetition.

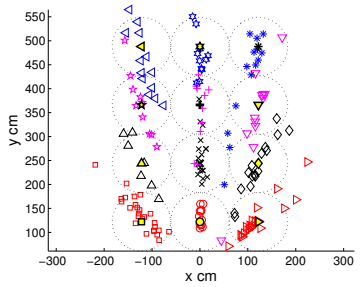

Fig. 13: Field test localization map. The dotted circle has a reference radius as 61cm. Each listener is located at the center of each circle. Results are denoted using same maker as the target's ground truth location.

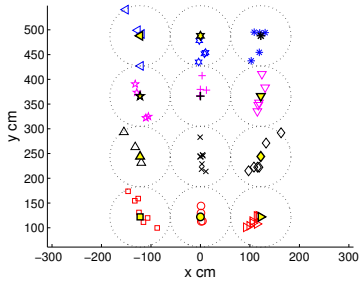

Fig. 14: Field test using three triangles. The dotted circle has a reference radius as 61cm. Each listener is located at the center of each circle. Results are denoted using same maker as the target's ground truth location.

TABLE II: Computation time of cross-correlation

Gesture	Samples		Finishing time (s)			
	Mean	Std	cross-corr. (mean)	cross-corr. (std)	AMIL(mean)	AMIL(std)
Line (4 beeps)	389734	16249	46.06	1.78	3.09	0.07
Triangle (5 beeps)	427008	9523	50.40	1.16	3.68	0.05

previous works [1] and [4] claim that noise is not a problem for their designs. To solve the problem, we have also implemented a chirp in the inaudible frequency range of [18k, 22k]. The new chirp is more noise resistant but suffers from shorter distance and certain inaccuracy.

Phone orientation: We measured the 3D polar pattern of Galaxy Nexus's built-in speaker and microphone through PCB 378B02 condenser microphone. As we expected, the rear acoustic field is stronger than its front counterpart, because the curvature near the bottom edge of Nexus causes a small angle between the speaker front and the x-y plane. For different phone orientation, the localization distance instead of the accuracy is relatively affected.

3D localization: In our evaluation, we moved the phone on the surface of the floor or a table for 2D localization. We have tested our system by freely moving the phone in the air. Unfortunately, the performance is not satisfactory due to the noisy readings from IMUs. Therefore, we have not included the data for free movements. In our future work, we will work on extending our system to 3D space.

Multiple users: The current implementation assumes that there is only one player at a time. However, it can be easily extended to support multiple users by selecting different frequency bands for beeps and employing a similar multiple access protocol used in [7].

User Effort: 1) There is not much effort involved in drawing those gestures. The user only needs to move the phone in her hand to draw the gesture quickly. 2) AMIL can be implemented as a system service in the Android framework to avoid the installation effort. 3) It is possible to avoid the network setup effort by relying on a service in Cloud providing public APIs to applications.

VI. Related Work

While GPS-based localization has been improved in recent years [8], it can not be directly applied to indoor scenario. How to locate without GPS has been studied over two decades, which can be categorized into: Radio Frequency (RF) based techniques and Acoustic techniques.

RF based techniques. In these techniques, location is determined by measuring the radio signals from Wi-Fi APs, RFID or cellular towers. Some of those techniques are proximity-based, which can only provide low accuracy [9]–[11]. By profiling received signal strength (RSS) fingerprints for each location, finer localization is performed by finding a location with the matched fingerprint [12]–[15]. Besides that, FM radio [16] and channel responses from multiple OFDM subcarriers [17] are recently proposed as signatures. Different from signature-based approaches, several techniques exist for deriving range, angle and proximity information from radio signals, and then positions can be inferred by applying geometric algorithms. Time-of-arrival (TOA) systems such as [18] determine the distance between devices by measuring RF propagation delays. Time-difference-of-arrival (TDOA) systems such as [19] rely on the signal difference in arrival time and phase on time-synchronized devices to determine range. Angle-of-arrival (AOA) systems [20] utilize the directions from which a signal is received to derive positions. Through measuring the RSS of RF signals, the location of devices can be also determined by employing a radio propagation model [21], [22]. These approaches do not provide provisions to accurately locate nearby mobile users in any circumstance, since they typically need profiling in advance, special hardware design, or only provide coarse-grained precision (e.g., room-level).

Acoustic techniques. Acoustic techniques can measure the range more precisely, owing to its relatively slow speed compared with RF signal. Hence, most acoustic localization schemes leverage range-based approaches. Many systems such

as [23]–[26] adopt custom hardware to measures the time-of-flight of modulated ultrasonic signals to estimate the range between devices. The ENSBox system [27] leverages microphone array to obtain orientation information for localization. These approaches cannot be applied to mobile phones without additional hardware. The BeepBeep system [1] designed to work with ordinary mobile devices with speaker/microphone introduces a novel way to measure the range based on the elapsed time between two time-of-arrival (ETOA) of two audio tones. Based on BeepBeep, the work [2] uses multiple speakers and microphones to perform phone-to-phone localization in 3D space. SwordFight [4] improves BeepBeep by supporting fast and continuous phone-to-phone ranging. Different from those approaches, only a single device emits audio tones in our work, thus eliminating the requirement of time synchronization and significantly improve the scalability. Recent work [5] proposes another acoustic TDOA-based ranging technique for mobile phone self-localization with infrastructure support. In addition, acoustic fingerprint is also used for indoor localization such as the work [28].

Miscellaneous. There also has been research focused on hybrid techniques of both RF based and acoustic localization. WALRUS [29] can achieve room-level localization in office environment by broadcasting the identity of the room through sound and Wi-Fi channels. Centaur [6] improves the resolution of localization by acoustic ranging plus Bayesian inference. Acoustic ranging techniques are also leveraged to detect driver phone use [30], and pair intended devices by a pointing gesture [31], [32]. Other related works in the context expect for localization is to leverage IMU sensors for the movement recognition. The techniques proposed in the work [33] can recognize human handwriting using phones. Those algorithms relies on IMU sensors to extract the features of the movement, while our targeted problem is more challenging that demands measuring the precise displacement of the movement.

VII. Conclusion

In this paper, we consider the problem of efficiently and securely grouping and locating mobile phone users in proximity. A system called AMIL is proposed to leverage a simple gesture to perform localization during network setup. By using internal motion sensors and speakers/microphones, our scheme combines gesture detection and acoustic techniques for a user to locate other users in an efficient, low-cost and scalable manner. We have designed, implemented and evaluated our system on commercial smartphones. Extensive experiments have shown that AMIL can achieve less than three degree error in orientation and 50cm error in distance.

Acknowledgment

The authors would like to thank all the reviewers for their helpful comments. This project was supported in part by US National Science Foundation grant CNS-1320453.

References

[1] C. Peng, G. Shen, Y. Zhang, and K. Li, Yanlin swand Tan, "Beepbeep: a high accuracy acoustic ranging system using cots mobile devices," in *SenSys '07*, pp. 1–14.

[2] J. Qiu, D. Chu, X. Meng, and T. Moscibroda, "On the feasibility of real-time phone-to-phone 3d localization," in *SenSys '11*, 2011, pp. 190–203.

[3] J. Luo, H. V. Shukla *et al.*, "Non-interactive location surveying for sensor networks with mobility-differentiated toa," in *INFOCOM '07*.

[4] Z. Zhang, D. Chu, X. Chen, and T. Moscibroda, "Swordfight: enabling a new class of phone-to-phone action games on commodity phones," in *MobiSys '12*, pp. 1–14.

[5] P. Lazik and A. Rowe, "Indoor pseudo-ranging of mobile devices using ultrasonic chirps," in *SenSys '12*, pp. 99–112.

[6] R. Nandakumar, K. K. Chintalapudi, and V. N. Padmanabhan, "Centaur: locating devices in an office environment," in *Mobicom '12*.

[7] G. E. Santagati and T. Melodia, "U-Wear: Software-defined ultrasonic networking for wearable devices," in *Mobisys '15*.

[8] X. Zhu, Q. Li, and G. Chen, "Apt: Accurate outdoor pedestrian tracking with smartphones," in *INFOCOM '13*, pp. 2508–2516.

[9] C. C. Tan, Q. Li, and L. Xie, "Privacy protection for rfid-based tracking systems," in *RFID '10*. IEEE, pp. 53–60.

[10] H. Han, F. Xu, C. C. Tan, Y. Zhang, and Q. Li, "Vr-defender: Self-defense against vehicular rogue aps for drive-thru internet," *Vehicular Technology, IEEE Transactions on*, vol. 63, no. 8, 2014.

[11] Y. Zhang, C. C. Tan, F. Xu, H. Han, and Q. Li, "Vproof: Lightweight privacy-preserving vehicle location proofs," *Vehicular Technology, IEEE Transactions on*, vol. 64, no. 1, pp. 378–385, 2015.

[12] P. Bahl and V. N. Padmanabhan, "Radar: An in-building rf-based user location and tracking system," in *INFOCOM '00*, pp. 775–784.

[13] M. Youssef and A. Agrawala, "The horus wlan location determination system," in *MobiSys '05*, pp. 205–218.

[14] L. M. Ni, Y. Liu, Y. C. Lau, and A. P. Patil, "Landmarc: Indoor location sensing using active rfid," *Wireless Networks*, vol. 10, 2004.

[15] Z. Yang, C. Wu, and Y. Liu, "Locating in fingerprint space: wireless indoor localization with little human intervention," in *Mobicom '12*.

[16] Y. Chen, D. Lymberopoulos, J. Liu, and B. Priyantha, "Fm-based indoor localization," in *MobiSys '12*, pp. 169–182.

[17] S. Sen, B. Radunovic *et al.*, "You are facing the mona lisa: spot localization using phy layer information," in *MobiSys '12*, pp. 183–196.

[18] M. Youssef and U. Shankar, "Pinpoint: An asynchronous time-based location determination system," in *MobiSys '06*, pp. 165–176.

[19] B. Kusy, J. Sallai, G. Balogh, A. Ledeczi, V. Protopopescu, J. Tolliver, F. DeNap, and M. Parang, "Radio interferometric tracking of mobile wireless nodes," in *MobiSys '07*, pp. 139–151.

[20] J. Friedman, Z. Charbiwala, T. Schmid, Y. Cho, and M. Srivastava, "Angle-of-arrival assisted radio interferometry (ari) target localization," in *MILCOM '08*, pp. 1–7.

[21] K. Chintalapudi, A. Padmanabha Iyer, and V. N. Padmanabhan, "Indoor localization without the pain," in *MobiCom '10*, pp. 173–184.

[22] N. Banerjee, S. Agarwal, P. Bahl *et al.*, "Virtual compass: relative positioning to sense mobile social interactions," in *Pervasive '10*.

[23] A. Harter, A. Hopper, P. Steggles, A. Ward, and P. Webster, "The anatomy of a context-aware application," in *MobiCom '99*, pp. 59–68.

[24] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan, "The cricket location-support system," in *MobiCom '00*, pp. 32–43.

[25] M. Hazas and A. Ward, "A novel broadband ultrasonic location system," in *Ubicomp '02*, pp. 264–280.

[26] K. Liu, X. Liu, and X. Li, "Guoguo: Enabling fine-grained indoor localization via smartphone," in *MobiSys '13*, pp. 235–248.

[27] L. Girod, M. Lukac *et al.*, "The design and implementation of a self-calibrating distributed acoustic sensing platform," in *SenSys '06*.

[28] S. P. Tarzia, P. A. Dinda *et al.*, "Indoor localization without infrastructure using the acoustic background spectrum," in *MobiSys '11*, pp. 155–168.

[29] G. Borriello, A. L. Liu, T. Offer *et al.*, "Walrus: wireless acoustic location with room-level resolution using ultrasound," in *MobiSys '05*.

[30] J. Yang, S. Sidhom, G. Chandrasekaran *et al.*, "Detecting driver phone use leveraging car speakers," in *MobiCom '11*, pp. 97–108.

[31] C. Peng, G. Shen, Y. Zhang, and S. Lu, "Point&connect: intention-based device pairing for mobile phone users," in *MobiSys '09*, pp. 137–150.

[32] Z. Sun, A. Purohit *et al.*, "Spartacus: Spatially-aware interaction for mobile devices through energy-efficient audio sensing," in *Mobisys '13*.

[33] S. Agrawal, I. Constandache, S. Gaonkar, R. Roy Choudhury, K. Caves, and F. DeRuyter, "Using mobile phones to write in air," in *MobiSys '11*.