MoBat: Sound-Based Localization of Multiple Mobile Devices on Everyday Surfaces

Adrian Kreskowski
Jakob Wagner
Jannis Bossert
Florian Echtler

Bauhaus-Universität Weimar
Weimar, Germany
`firstname.lastname`
`@uni-weimar.de`

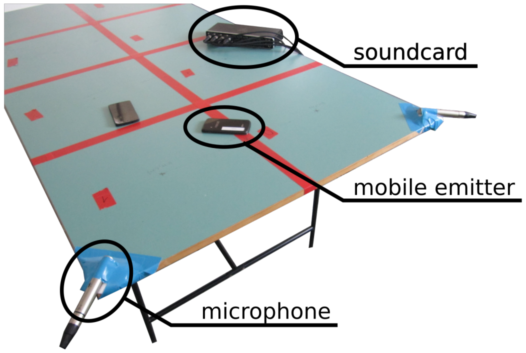

Figure 1: An overview of the used hardware. Shown is the USB sound device in the back, two mobile phones on the table and two microphones in front.

Abstract

We present MoBat, a combined hard- and software system designed to locate and track multiple unmodified mobile devices on any regular table using passive acoustic sensing. Barely audible sound pulses are emitted from mobile devices, picked up by four microphones located in the corners of the surface and processed in a low-latency pipeline to extract position data. We demonstrate an average positional accuracy and precision of about 3 cm on a table of 1 m x 2 m size, and discuss possible usage scenarios regarding proxemics and tangible interaction.

Author Keywords

Sound-based Localization; Tangible Interfaces; Multitouch Surfaces; Mobile Devices; Time Difference of Arrival

ACM Classification Keywords

H.5.2 [User Interfaces]: Input devices and strategies

Introduction and Related Work

Personal mobile devices are quickly turning into a central element of users' information behaviour. Various types of data are collected and shared using these devices, and novel interaction metaphors like proxemics need to take this aspect of personal information management into account. Multiple approaches exist to locate mobile devices with respect to each other and to the environment. However, such

ITS '15, November 15-18, 2015, Funchal, Portugal.
ACM 978-1-4503-3899-8/15/11.
http://dx.doi.org/10.1145/2817721.2823488

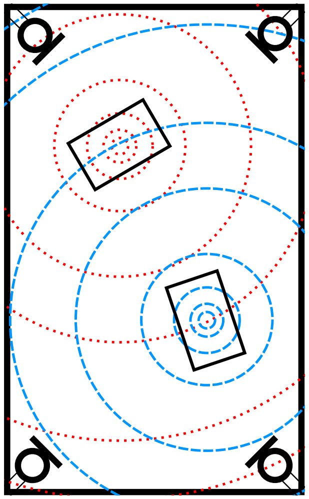

Figure 2: A sketch of the tabletop hardware setup: The rectangle indicates a tabletop area of arbitrary 2D dimensions. The black circles near the corner indicate the four microphones, which are used to record the audio sources. We furthermore sketched two audio sources emitting sound on different frequencies as illustrated by the color coding and different stroke style of the concentric circles. The center of each circle represents the position of the audio sources.

methods usually require prior setup in the form of attaching fiducial markers or other tracking aids.

In this paper, we present *MoBat*, a hard- and software system designed to locate mobile devices on a surface using passive acoustic sensing of barely audible sound pulses. We distinguish between two core parts, the emitter and the tracking side. On the emitter side, only the capability to play an audio stream is required. For the tracking, the surface merely needs to be augmented using four cardioid microphones in the corners.

The usage of passive acoustic sensing to turn a flat surface into an input device was first realized by *Ishii et al.* [2], detecting the impact points of a ball on a Ping Pong table. This system was extended by *Paradiso et al.* [3] to detect and discern both taps with knuckles and a metal object on a glass surface. *Urashima et al.* [4] developed another system to identify letters from pen movements. However, all of these systems focus on detecting discrete sound events like taps rather than continuous signals.

There are two approaches for locating sound sources. The first one is the "Location Pattern Matching", which utilizes a collection of recorded signals and was used in [1] [2]. By comparing the recorded signal with the signals in the database, a position is determined. This pattern matching approach allows application on arbitrary surfaces with few microphones but requires a learning phase for each new setup. The second approach, "Time Difference of Arrival" (hereafter TDOA) was used in [3] [4] [5]. It is based on measuring the time difference of the sound's arrival at the different microphones. Knowing the position of the microphones and the speed of sound, the sound source can be located. This approach requires no learning phase but a homogeneous transfer medium and at least three microphones. For reasons of portability, we decided to base our system on TDOA.

Hardware Architecture

Due to the TDOA technique we apply for our sound localization, we rely on synchronously transmitted audio buffers. Therefore, the core part of our hardware setup is an external sound card with four connected microphones at known positions. It streams an audio buffer containing four channels via USB to our localization pipeline. Initially, we experimented with piezoelectric elements to sense the vibration within a chipboard table. However, the signal-to-noise ratio was insufficient for our transformation pipeline. By utilizing low-budget directional microphones, it was possible to clearly identify the signal. In addition to the soundcard and the microphones a standard PC was used for the signal processing. Figure 2 sketches a possible setup with four mounted microphones, and two audio sources; Figure 1 shows a photo of the used hardware.

Software Architecture

The first part of our software architecture is a sound emitting application. Running on a mobile device, it emits periods of a sine wave alternating with a pause. The sine signals can have a frequency of up to 20 kHz, depending on the individual devices' audio properties. The sine signal and the pause each have a fixed length. This alternation allows for continuous tracking of the devices by identifying peaks in the frequency domain after a pause (see Figure 4). Our system is able to parallel track multiple emitters with each of them producing different sine frequencies.

The pipeline is built to record an audio signal and locate the individually configured sound sources. The three main stages of the pipeline can be classified as the **Recorder**, **Analyzer** and the **Locator Stage** and are shown in Figure 3. In the following paragraphs, we will describe these in

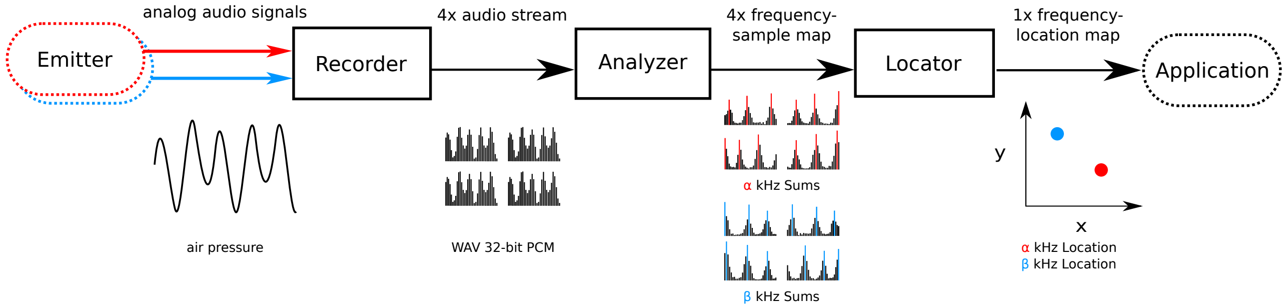

Figure 3: A simplified version of the MoBat Pipeline: Several emitters produce signals of different trackable frequencies, which are fed into the pipeline. They are then processed in three stages indicated by the boxes. These perform the recording of the audio signal, determination of frequency peaks per channel and the localization based on determined TDOAs per frequency. The output of the pipeline is a position for each located emitter.

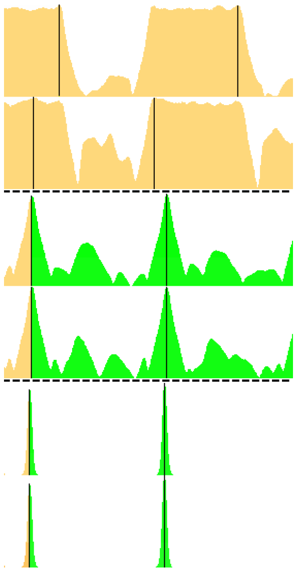

Figure 4: The image above visualizes three different types of accumulated amplitudes of a 19 kHz signal, for 2 channels each. The top third shows the results for 2 audio channels when the emitter is configured to use a longer sine than pause signal. A lot of consecutively executed STFTs will contain a similar frequency sum amplitude and therefore finding a peak is error prone. The center third shows the characteristic for a pause that is five times longer than the sine. The peaks, indicated by the black vertical bars, are detected within a valid range. The last third shows a post-processed signal.

detail.

RECORDER STAGE In this stage the analogue signal of the ambient sound is recorded and converted to digital 32-bit PCM by the sound card. The communication with the audio device is done using the *ALSA* sound library. For each of the four microphones, the audio signal is recorded within a time interval of 130 ms (contingent on the sound card) to natively ensure a high update rate for the entire localization.

ANALYZER STAGE The analyzer is responsible for determining a time-delta of the recorded channels for each of the frequencies registered in the pipeline. Similarly to the implementation in the Toffee pipeline of Xiao et al. [5], this is done by considering amplitudes per sample of each audio channel. However, since we aim to detect not only a single audio signal but multiple frequencies over time, it is not sufficient to sample the amplitude of the buffer of the recorded signal. In order to retrieve information about the different frequencies over time, we apply a series of short-time Fourier transformations (STFT) for each of the chunks per channel. For our pipeline, we consider a Fourier window size of 256 samples. Furthermore, a sliding window offset of one sample is used to work efficiently and with a sufficient temporal resolution for our purpose. Since the signal-to-noise ratio of the Fourier transformation of just one window per time-step is too low for a reasonable peak detection, we sum up the results of several consecutive Fourier windows. The latter can be applied with negligible costs by caching all but one result for the next frequency-amplitude sum. In order to reduce the memory footprint of our application, only the amplitudes of the registered frequencies are stored and further analyzed.

The frequency sums tend to take the shape of banks and therefore contain several local maxima as shown in Figure 4. The first step to distinguish the peaks from noise is a

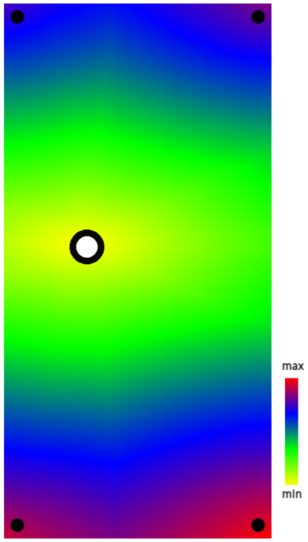

Figure 5: The error distribution of the TDOA method: The color coding of this heatmap visualizes the TDOA error for each of the uniform grid cells. The emitter (white circle) is assumed to be on the cell with the lowest error.

normalization of the frequency sums per chunk. The lowest and highest sums act as normalization limits. Hereby, the peaks below a certain threshold are filtered out. Yet, it is still not possible to accurately distinguish frequency peaks from intermediately high values by simple thresholding. This is due to different amounts of noise in recorded chunks, which can result in peaks of different height or different sharpness. In order to obtain a small number of distinct peaks, we exponentiate the sums at each sample. This operation results in a set of easily distinguishable peak candidates per chunk and channel.

When all channels contain a peak and the time distance between them is not larger than an adjustable threshold, closest peaks are matched. This is done under the assumption that the corresponding frequency peaks in different channels have a relatively small sample distance in a real environment. Based on this method, we achieve satisfactory results for our application as seen in the following section. Naturally, further filtering steps can be applied to the raw position data depending on the application context.

As a final remark on the analyzer stage, it should be noted that the STFT will create a power spectrum containing all frequencies. This transformation is by far more time consuming than the peak detection for a transformed signal. Therefore, the peak detection for additional frequencies does not severely affect the overall runtime.

LOCATOR STAGE The final stage of the MoBat pipeline receives the four determined signal starting times for each registered frequency. The area between the microphones is then uniformly divided into a 2D grid and the error is computed for each of the vertex positions in a TDOA manner using the same error formula as *Xiao et al.* Figure 5 shows the error distribution of the grid cells. Note that we concentrated on the analyzer stage for recognizing multiple sound sources; therefore, the locator stage does not employ a gradient-based optimization step. The processing overhead introduced by this naïve error-sampling approach can be considered as negligible for our system since the most expensive part of the pipeline remain the Fourier transformations.

Performance and Accuracy

SYSTEM SPECIFICATION The test set-up consisted of 4 *t.bone EM700* microphones, one *Behringer U-PHORIA UMC404* sound-card and a notebook with an *Intel Core i7-2630QM 2Ghz* CPU. As sound emitters, several *Nexus 4* devices were used. In each test we measured the position error and calculated the distance to the actual position, standard deviation and the 90-percentile.

The following list shows the most important parameters used in the tests described below:

- sine length: 5 ms
- pause length: 25 ms
- recording duration per smartphone position: 1 min
- table size: 2m x 1m
- sampling grid cell size: 1cm
- Fourier window size: 256 samples

The parameters above were established empirically and led to the best test results.

TEST SERIES

Test 1 One smartphone playing a 19 kHz frequency was placed at 12 different test positions on the table. Figure 6 visualizes the results. The test results show a standard deviation of 1.55 cm, while the mean distance to the actual

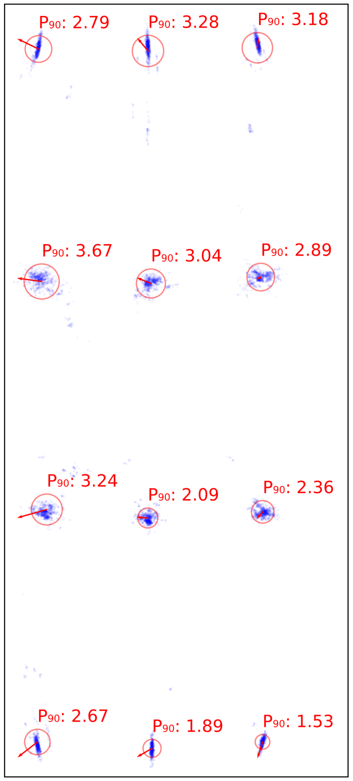

Figure 6: Position distribution for Test 1: One 19 kHz smartphone was placed sequentially on different positions on the table. The located points are shown in blue. The darker the blue, the higher the amount of positions that fell on this very location. The red arrow indicates the shift of the mean position to the original position of the smartphone. The red circle indicates the 90 percentile.

smartphone position was 3.16 cm. The average 90 percentile was 2.72 cm.

Test 2 In this test, the influence between two emitters with frequencies of 18 and 19 kHz was evaluated. We ascertained a similar standard deviation of 1.44 cm and a 90 percentile of 2.94 cm. The average distance to the actual position was 1.77 cm.

Test 3 In this test, the influence between three emitters with frequencies of 16, 18 and 20 kHz was evaluated. The results can be seen in Figure 7. The average standard deviation of the three emitters amounted to 2.18 cm. The average 90 percentile was 3.7 cm. The distance to the actual position was 4.56 cm.

As indicated by the test results, the usage of multiple smartphones leads to a decreased accuracy and precision. This behaviour is likely caused by the limited frequency resolution for the chosen pipeline parameters. Throughout all tests, the system had an average position update-rate of 20.26 positions per second. Even after using the system for several months, we could not observe an impairment of the speaker caused by the permanently emitted signals. Furthermore, the devices' batteries lasted for more than 12 hours of continuous use.

It is worth mentioning that the performance of the system is dependent on the speaker quality and placement of the smartphone. On some devices (e.g. Nexus 4) the speakers are located on the back, which causes noticeable damping of the signal when the device is lying flat on the surface. We also observed a slight directional component for phones with side-mounted speakers (e.g. Nexus 5).

Discussion and Outlook

In this paper, we showed that continuous sound-based tracking of multiple devices is possible using a normal desk, four microphones and a semi-professional sound card. In the following paragraphs, we discuss the performance and applicability of our system.

In contrast to the reference systems, which focus on impulse detection of one sound source, MoBat is designed to continuously locate multiple sources simultaneously. Despite these differences, the applied localisation techniques are similar enough to allow a performance comparison.

Compared to existing systems such as Toffee [5], our hardware setup is similarly simple. Furthermore, our system does not require specialized devices except a four-channel sound card.

The accuracy of our system is on par with other sound based tracking systems. PingPongPlus [2] has an accuracy of a few inches, just as Toffee [5]. Our system determines the position of the emitters several times per second and has a maximum standard deviation of 2.2 cm. With the continuous tracking approach the position variance becomes noticeable. The variance decreases as the table size increases. This is due to the audio signal's increased time of flight, which allows for greater TDOA differences. Consequently, the peak detection becomes more resistant to inaccuracies on larger tables.

The latency of our system is not only caused by the processing time but also the inclusion of recently determined positions to stabilize the results. As with all post-processing methods, there is a trade-off between latency and precision.

MoBat is especially well suited for applications in which the emitters are not moved frequently. In more dynamic use

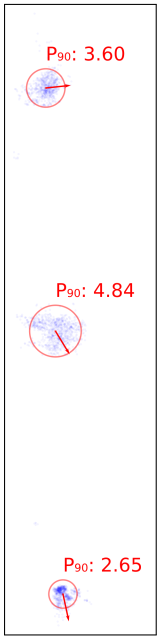

Figure 7: Position distribution for Test 3: Each of the circles indicates one smartphone with one of the frequencies 16, 18 or 20 kHz. The smartphones were simultaneously positioned on one line on the table with a distance of 50 cm from one to the next. The interpretation of the color and shape coding is the same as in Figure 6

cases where movement is a crucial part, our system may not be preferable over conventional input methods because the stabilization of the position causes a higher latency. As a test application, we developed an air-hockey game, using mobile phones as paddles. To get first impressions of the system, we let several persons play the game and observed them. We gained the impression that after a small amount of time, people could adapt to the paddles movements. A side effect of rapidly alternating between sine and silence is a clicking noise produced by the emitters. This may be distracting if a user focuses on it. However, the clicking noise can be easily concealed by other sounds such as music or talking. The accuracy will not be affected as long as the background sounds do not contain frequencies that are produced by the emitters.

In contrast with visual tracking approaches, we do not need to modify the appearance of the tracked devices using markers. As another notable advantage, sound-based tracking is not affected by visual occlusions. As a result the tracking is not interrupted when the emitters are held and moved. However, a disadvantage is the influence of acoustic occlusion caused by obstacles located between emitter and microphones.

For further research, we would like to remove the need for specialized software on the smartphones. For instance, it should be possible to establish a data-link between the emitters and a server that assigns a unique frequency to each smartphone. The server could then stream the corresponding signal to each mobile device via a website. It is also possible to implement proxemics techniques based on the relative distance of the emitters. This implies again an interaction between the mobile devices and a server. For integrating further degrees of freedom into our system, it would be possible to use information provided by the smartphones' sensors (gyroscope, accelerometer, magnetometer) to also incorporate the orientation in the result of the pipeline. This allows for more complex input metaphors. Another interesting approach would be to integrate speakers and a simple circuit into small objects to create low-cost sound-based tangibles. These could provide similar interaction possibilities for users not willing to use their smartphones as input device for this system. At the moment, the tracking capabilities of our system are restricted to two dimensions. To allow for three-dimensional tracking, the system could be extended to work with more microphones. Our system is open-source and is available for download from https://github.com/mmbuw/mobat.

REFERENCES

1. C. Harrison and S. E. Hudson. 2008. Scratch Input: Creating Large, Inexpensive, Unpowered and Mobile Finger Input Surfaces, In Proc. UIST '08. 205–208.

2. H. Ishii, C. Wisneski, J. Orbanes, B. Chun, and J. A. Paradiso. PingPongPlus: Design of an Athletic-Tangible Interface for Computer-Supported Cooperative Play. Marian G. Williams and Mark W. Altom (Eds.). ACM, 394–401.

3. J. A. Paradiso, C. K. Leo, N. Checka, and K. Hsiao. 2002. Passive Acoustic Sensing for Tracking Knocks Atop Large Interactive Displays. 11–14.

4. A. Urashima and T. Toriyama. 2010. Ubiquitous Character Input Device Using Multiple Acoustic Sensors on a Flat Surface.

5. R. Xiao, G. Lew, J. Marsanico, D. Hariharan, S. E. Hudson, and C. Harrison. 2014. Toffee: enabling ad hoc, around-device interaction with acoustic time-of-arrival correlation, In Proc. MobileHCI '14. 67–76.