This Thesis Committee for Huihuang Zheng

Certifies that this is the approved version of the following thesis:

Acoustic Motion Tracking and Its Application

APPROVED BY

SUPERVISING COMMITTEE:

Lili Qiu, Supervisor

Mohamed Gouda

Acoustic Motion Tracking and Its Application

by

Huihuang Zheng, B.S.

THESIS

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

THE UNIVERSITY OF TEXAS AT AUSTIN

May 2017

Acknowledgments

Life is not always easy. During my hard time of graduate school, I wish to thank the people who helped me.

Sadly, I think I am an unlucky man. There were many things that happened which changed my life. One of these things is that I didn't want to go to U.S but I was forced to go there for graduate study because of some secret reasons. My family doesn't have enough money for my tuition so I have extreme pressure during the time of my Master program. I also came across problems of confusion in research. Not to mention, the hard time for an international student in language, culture, food, and habit difference.

I would like to thank Professor Lili Qiu, who is my supervisor. She helped me not only in research but also gave me a fellowship. This thesis could not be completed without her guide.

I would like to give special thanks to Wenguang Mao for collaborating with me on this project.

I would like to thank my lab-mates Yi-Chao Chen, Sangki Yun, Swadhin Pradhan, Jian He, Mubashir Quershi, Ghufran Baig, and Mei Wang for helping during my graduate school.

I would like to thank Professor Kristen Grauman. I have been doing

some research stuffs in her computer vision lab. She recognized my ability and referred me to the computer science department and Professor Krähenbühl to be a teaching assistant in Deep Learning class.

I would like to thank my family for supporting me a lot. They gave me everything they could and they always told me not to worry about uncertain future.

Went through a hard life time is easier said than done, it is a loud pain in my mind. Falling down and getting up is not easy. If I, an unfortunate, suffer a hard time again, I hope I can find the right direction. I hope everything that drowns me makes me want to fly.

Acoustic Motion Tracking and Its Application

Huihuang Zheng, M.S.COMP.SC.
The University of Texas at Austin, 2017

Supervisor: Lili Qiu

Video games, Virtual Reality (VR), Augmented Reality (AR), and Smart appliances (e.g., smart TVs) all call for a new way for users to interact and control them. This thesis explores high precision acoustic motion tracking system which aims to replace traditional control devices such as mouse and let the user play games, interact with VR/AR headsets, and control smart appliances. We develop a lightweight system which can achieve mm-level tracking accuracy using inaudible sounds.

At the heart of our system lies a *distributed* Frequency Modulated Continuous Waveform (FMCW) which is able to accurately estimate the absolute distance between a receiver and a transmitter that are separate and unsynchronized. We further develop an optimization framework to combine FMCW estimation with Doppler shifts and Inertial Measurement Unit (IMU) measurements to enhance the accuracy, and efficient algorithm to solve the optimization problem.

We develop several interesting applications on top of our motion tracking technology, including audio ruler, drawing in the air, and playing motion-controlled games.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Virtual Reality (VR), Augmented Reality (AR), Video games, and Smart appliances all call for a new way for users to interact and control them. For example, motion games (i.e., the games played by movement) are popular across the world. On the other hand, through interviewing 100+ game players, we have found many of them are unsatisfied with the existing tracking technologies in the motion games: (i) they often complain about the tracking accuracy, and (ii) the coarse-grained tracking only supports limited types of motion games, such as sports and dancing games. However, many players prefer motion games that require more fine-grained movement, such as first person shooter (FPS).

In addition, the current interfaces of VR/AR are rather constrained (e.g., relying on tapping, swiping, or voice recognition). This significantly limits its potential applications. Moreover, smart appliances are becoming increasingly popular. For example, smart TVs offer a rich set of controls and it is important

This thesis is based on the work [22, 23]

for users to easily control smart TVs. More and more appliances will become smart, and allow users to control them remotely.

Since mobile devices, such as smartphones and smart watches, are becoming powerful and ubiquitous, they can potentially serve as universal motion controllers. To turn a mobile device into an effective motion controller, its movement should be tracked accurately - within a centimeter.

There have been a number of very interesting works on motion tracking and localization (e.g., audio-based schemes [28, 32, 49, 51], RF-based schemes [41, 43, 47, 48], and vision-based schemes [2, 3]). Recent works reduce the tracking error significantly by using many antennas and new spectrum (e.g., 60 GHz). Despite significant work on localization and tracking, achieving mm-level tracking on commodity devices remains an open challenge. Therefore, we aim to achieve high tracking accuracy, minimize error accumulation over time, and remove the need of special hardware.

1.1 Our Approach

In this work, we develop a novel system, called high-pre**C**ision **A**coustic motion **T**racker (CAT), which turns a mobile phone into a motion controller. CAT can be potentially used to control game consoles, VR/AR, and smart appliances. A unique feature of our approach is that it uses existing hardware already available, while achieving high accuracy and ease of use.

We use audio signals for our tracking system because (i) it propagates s-

lowly, which makes it possible to achieve high accuracy, (ii) it can be supported by commodity devices thanks to widely available speakers and microphones, and (iii) its processing cost is low due to its low sampling rate.

In our system, the speakers serve as anchor points and play specially designed acoustic signals. Our system then estimates the distances and velocities with respect to the speakers based on the received signals using a *distributed* Frequency Modulated Continuous Waveform (FMCW) and Doppler shifts. It then fuses the distances and velocities in an optimization framework to accurately track the movement.

Doppler shift is a well known phenomenon where the signal frequency changes as a sender or a receiver moves. By tracking the amount of frequency shift, we can estimate the speed of the mobile with respect to the speakers. To determine its position, the speed needs to be integrated over time, which incurs error accumulation.

To minimize error accumulation, we develop a novel FMCW-based approach to directly estimate the distance between the mobile and the speakers. Our FMCW differs from existing approaches due to the distributed nature of our system: the speakers (i.e., transmitters) and the microphone on the mobile (i.e., receiver) are separate and unsynchronized. In this case, the transmission time, which is required by traditional FMCW approaches, is not known by the receiver. Our distributed FMCW addresses the issue using the following steps: i) find a reference point and determine its absolute position, ii) estimate the distance change with respect to the reference point when a mobile moves, 3)

derive the absolute distances between the current point and speakers. In this way, we no longer need the transmission time. Moreover, a separate transmitter and receiver have different sampling frequencies. To address the issue, we develop a simple procedure to calibrate and compensate for the frequency offset.

Furthermore, we develop an optimization framework to incorporate FMCW measurements and Doppler shifts over time for accurate tracking. These two types of measurements are complementary: the former gives distance estimation, which does not incur error accumulation, while the latter provides more accurate distance change in a short term, which helps smooth the estimated trajectory. The framework can further incorporate IMU measurements (e.g., accelerometers and gyroscopes) to improve the accuracy.

We implement our approach on two platforms: (i) one consisting of a desktop and a smartphone, where the desktop processes the audio signals fed back from the smartphone to track the smartphone, and (ii) another consisting of just speakers and a smartphone, where the smartphone tracks its location in real time based on the received audio signals. Our evaluation also uses different settings, which reflect scenarios of console games and VR/AR. For 2D tracking, we show that CAT can achieve median tracking error of 7 mm in a PC-like setup and 12 mm in a VR-like setup with two speakers. The corresponding errors under three speakers reduce to 5 mm and 7 mm in 2D, respectively. For 3D, the error is 8 mm - 9 mm. In comparison, our distributed FMCW alone has 2 cm error in 2D, and the Doppler alone [49] has 2 cm error in

the first few seconds but its error increases over time due to error accumulation (e.g., 8 cm after 30 seconds in our experiments).

Our major contributions include:

- a *distributed* FMCW approach that achieves highly accurate distance estimation without requiring synchronized and co-located sender and receiver,
- an optimization framework to combine distance and velocity estimation over multiple time intervals to accurately track motion, and an efficient algorithm to solve it on a mobile device,
- Applications
  - First, we enable audio ruler. We let a speaker continuously transmit specific acoustic signals. We invite a user to place a smartphone at an arbitrary distance away from the speaker. The smartphone will analyze the acoustic signal to compute and display the distance from the speaker. The user can easily evaluate the accuracy of our tracking system by comparing the readings displayed by the smartphone with the readings measured by the ruler.
  - Second, we will apply our tracking to drawing in the air. We let a user hold a smartphone and draw in the air. The smartphone will display the trajectory of its movement by analyzing the received audio signals using CAT.

– Third, we enable motion based computer gaming. We make the smartphone running CAT serve as a motion controller for video games by mapping the phone' s movement into a cursor movement in games using Windows API *mouse_event*. We invite a user to play a motion-based game using the smartphone. The smartphone acts as a game controller and analyzes the audio signal transmitted by several speakers to determine its location in real-time and feeds back to the computer, which will update the cursor position in the game. Users can assess the accuracy of CAT based on their gaming experience (e.g., whether the cursor in the game moves as they intend).

In the evaluation, our CAT system can achieve 5-7mm error in 2D and 8-9mm error in 3D.

1.2 Outline:

The rest of this thesis is organized as follows. We describe our approach in Chapter 2, and present implementation details and applications in Chapter 3. We evaluate its performance in Chapter 4. We review related work in Chapter 6, and conclude in Chapter 7.

Chapter 2

Our Approach

In our system, multiple static speakers (e.g., those on TV, computer, VR headset) transmit audio signals to a mobile. The mobile continuously estimates its velocity (Section 2.1) and distance (Section 2.2) to the speakers, and uses an optimization framework to combine these estimates to track its location (Section 2.2.2). We can further incorporate the IMU measurements in our optimization framework to improve the tracking accuracy (Section 2.2.3).

2.1 Estimating Velocity

The Doppler effect is a well known phenomenon where the frequency of a signal changes as a sender or a receiver moves [25]. Without loss of generality, we consider only the receiver moves while the sender remains static. The frequency changes with the velocity as follows:

$$v = \frac{F^s}{F}c, \tag{2.1}$$

where F is the original frequency of the signal, F^s is the amount of frequency shift, v is the receiver's speed towards the sender, and c is the propagation speed of sound waves. Therefore, by measuring the amount of frequency shift

F^s, we can estimate the receiver's velocity, which can further be used to get distance and location.

We use the approach described in [49] to estimate the mobile's velocity as follows:

- Each speaker continuously emits sine waves at inaudible frequencies. Different speakers use different frequencies to distinguish from each other.
- The mobile samples the received audio signals at 44.1 KHz (the standard sampling rate), applies Hanning window to avoid frequency leakage, and then uses Short-term Fourier Transform (STFT) to extract frequencies. We use 1764 samples to compute STFT, which corresponds to the audio samples in 40 ms. Then, we find a peak frequency and compare it with the frequency of the original sine wave. The difference between the two is the frequency shift F^s
- In order to enhance the accuracy, we let each speaker emit multiple sine waves at different frequencies and let the mobile estimate the frequency shift at each of the frequencies and combine these estimates by removing outliers and averaging the remaining estimates. We translate the final estimated frequency shift to the velocity based on Formula 2.1.

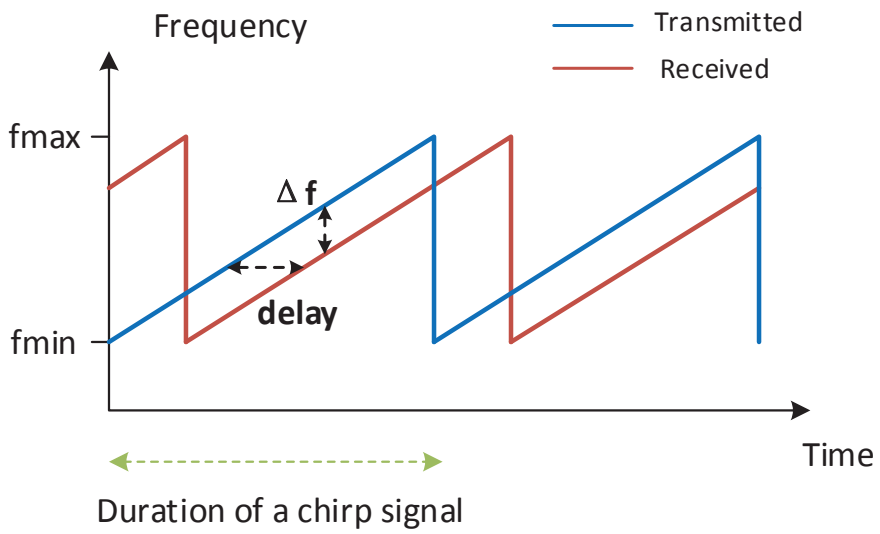


Figure 2.1: Chirp signals.

2.2 Estimating Propagation Delay

2.2.1 Motivation

As shown in [49], the Doppler shift alone can be used to provide reasonable tracking for a short time. However, since the Doppler shift gives a velocity estimate, it has to be integrated over time to get a distance estimate. Therefore, the error grows over time. For tracking over a long time interval, the accuracy of the Doppler shift based tracking degrades. In fact, error accumulation is common in many localization schemes (e.g., dead reckoning [35, 40]).

This motivates us to develop a method to overcome the error accumulation problem. One possibility is to directly estimate distance between the

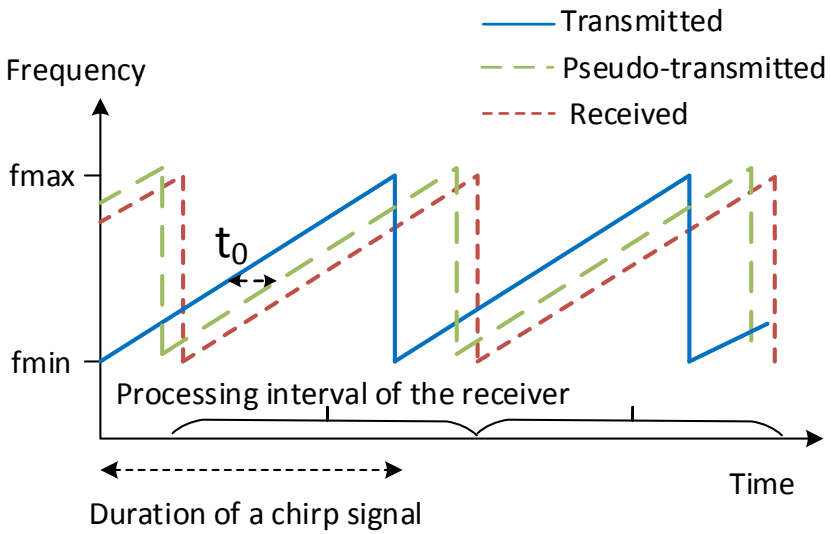


Figure 2.2: Pseudo-transmitted signals.

speakers and the mobile based on the propagation delay. Unlike velocity, which needs to be integrated over time and suffers from error accumulation, distance measurements can be used to directly determine the mobile's location and have no error accumulation. One way to estimate the propagation delay is to send a pulse signal and compute the difference between transmission time and arrival time. There are two practical challenges with this simple approach: (i) Due to the time-frequency uncertainty principle, we need large bandwidth in order to send a sharp pulse signal with good time resolution. Otherwise, the arrival time estimate will be inaccurate. (ii) Even if we can perfectly detect the start of the received signal, it is still challenging to estimate the propagation delay because (1) the sender and receiver's clocks are not synchronized and (2) there is non-negligible and variable processing delay at both ends; in commercial devices like smartphones, it is difficult to separate these delays from the propagation delay. In this section, we develop a new distributed FMCW

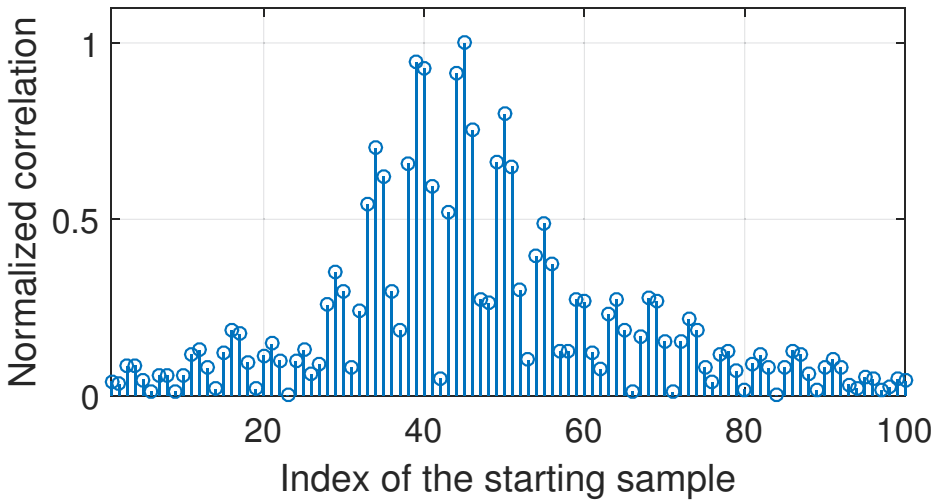


Figure 2.3: Correlation.

based approach to address these challenges.

2.2.1.1 Traditional FMCW

To accurately estimate the propagation delay, instead of sending a sharp pulse or pseudo-random sequence using large bandwidth [4, 51], we leverage a FMCW-based approach, which can achieve high estimation accuracy with moderate bandwidth usage [4].

FMCW approach lets each speaker transmit a chirp signal every period. Figure 2.1 shows periodic chirp signals, whose frequency sweeps linearly from f_{min} to f_{max} in each period. The frequency within each sweep is $f = f_{min} + \frac{Bt}{T}$, where B is the signal bandwidth, T is the sweep time. We integrate the frequency over time to get the corresponding phase: $u(t) = 2\pi(f_{min}t + B\frac{t^2}{2T})$. Therefore the transmitted signal during the n-th sweep is $v_t(t') =$

$\cos(2\pi f_{min}t' + \frac{\pi Bt'^2}{T})$, where $t' = t - nT$.

Consider a chirp signal propagates over the medium and arrives at the receiver after a delay t_d. The received signal is attenuated and delayed in time, and becomes:

$$v_r = \alpha \cos(2\pi f_{min}(t' - t_d) + \frac{\pi B(t' - t_d)^2}{T}),$$

where α is the attenuation factor.

The receiver mixes (i.e., multiplies) the received signal with the transmitted signal. That is, $v_m(t) = v_r(t)v_t(t)$. Thus, $v_m(t)$ is a product of two cosines. By using $\cos A \cos B = (\cos(A-B) + \cos(A+B))/2$ and filtering out the high frequency component $\cos(A+B)$, $v_m(t)$ becomes:

$$v_m(t) = \alpha \cos(2\pi f_{min}t_d + \frac{\pi B(2t't_d - t_d^2)}{T}). \tag{2.2}$$

Suppose the mobile is at distance R from the speaker initially and moves at a speed of v. Then t_d is given by $(R + vt')/c$. Plugging t_d into Formula 2.2 gives us

$$\alpha \cos(2\pi f_{min}\frac{R + vt'}{c} + (\frac{2\pi Bt'(R + vt')}{cT} - \frac{\pi B(R + vt')^2}{c^2T})).$$

If we analyze the frequency components of the above signal by taking the derivative of the phase, the constant term can be ignored and the terms quadratic with respect to $(1/c)^2$ are too small and can also be ignored. The remaining frequency component, denoted as f_p, becomes:

$$f_p = \frac{1}{2\pi}\frac{\delta Phase}{\delta t'} = \frac{BR}{cT} + \frac{f_{min}v}{c} + \frac{Bv}{c},$$

since mean$(t') = T/2$. When v is close to 0, there is a peak at $\frac{BR}{cT}$ in the frequency spectrum. If there are multiple propagation paths between the transmitter and the receiver, multiple peaks are observed in the spectrum of the mixed signal. In this case, f_p is determined by the first peak, which should correspond to the direct path. Based on measured f_p, the distance R can be derived as:

$$R = \frac{f_p cT}{B}. \tag{2.3}$$

2.2.1.2 Our FMCW

Traditional FMCW assumes that the transmitter and receiver are co-located and share the same clock. However, in our system, the speakers and microphone are separate and unsynchronized. Thus, we develop a distributed FMCW approach to support this situation. In our approach, we apply FMCW technique to derive the change of distance to the speaker when the mobile moves from one position to another. Moreover, we propose a scheme to find a reference point and leverage this point to convert the distance change to the absolute distance to the speaker. Also, we explicitly take into account the impact of movement on FMCW to improve its accuracy. We further account for the impact of sampling frequency offset between the speaker and microphone to get more accurate distance estimation. Below we elaborate each of these techniques.

Supporting a separate sender and receiver: In traditional FMCW technique, the transmitter and receiver have a shared clock. However, in our

setup the transmitter (i.e., speaker) and receiver (i.e., microphone) are separate and have unsynchronized clocks. Precise synchronization between the speaker and microphone in our scenario is challenging. Even a small synchronization error of 0.1 ms will lead to $0.1\,\text{ms} \times c \approx 3.46\,\text{cm}$ error, where c is the propagation speed of sound and around 346 m/s.

We estimate the propagation delay between a separate sender and receiver as follows. First, we perform approximate synchronization on the received signals to ensure that every *processing interval* (when we fetch and process audio samples) is aligned with a single chirp signal, as shown in Figure 2.2, since FMCW requires an almost complete received chirp. Approximate synchronization can be achieved by correlating the received signals with the original chirp signal. The maximum correlation indicates the best alignment. We select the time when the highest correlation peak is detected as the start time of the first processing interval. This synchronization only needs to be performed once at the beginning. Afterwards, we fetch received signals every 40ms (our processing interval) for FMCW processing. Note that the synchronization is approximate since the cross correlation usually shows multiple peaks with similar magnitude as shown in Figure 2.3.

After synchronization, we need to mix a received signal in each interval with a transmitted signal. However, the exact start time of a transmitted signal is unknown to the receiver. So we introduce a notion of *pseudo-transmission time*, i.e., the time when the receiver assumes the transmission begins. Let t_0 denote the difference between the pseudo transmission time and actual trans-

mission time of the first chirp signal. t_0 is an unknown constant to the receiver. At each interval, our estimated distance has a constant offset from the actual distance due to t_0. Since the offset is constant, we can estimate the distance change over time. To get an absolute distance at any time, we need to know the absolute distance at some point (called a reference point) and use the distance change to get the absolute distance at a new location.

Based on the pseudo transmission time, the receiver can construct a *pseudo transmitted signal*, which starts at that time, as shown in Figure 2.2. By mixing (i.e., multiplying) the received signals with the pseudo transmitted signals and applying a similar procedure as Section 2.2.1.1, we have:

$$R_n = \frac{cTf_n^p}{B} + ct_0, \tag{2.4}$$

where R_n is the distance between the transmitter and receiver during the n-th interval, f_n^p is the peak frequency of the mixed signals, c is the propagation speed of the audio signal, T is the chirp duration, and B is the bandwidth of the chirp signal, which is equal to $f_{max} - f_{min}$. Considering the above equations for two intervals, we can derive

$$R_n - R_1 = (f_n^p - f_1^p)\frac{cT}{B}.$$

If the distance between the transmitter and the receiver in the first interval (denoted as R_1) is known, R_n can be determined based on:

$$R_n = (f_n^p - f_1^p)\frac{cT}{B} + R_1. \tag{2.5}$$

Reference position estimation: To obtain the distance R_n, we need to know the absolute distance between the speaker and mobile at some point (i.e., R_1), also called a reference point. One way is to measure using a ruler. This could be cumbersome and error-prone. We develop the following simple yet effective calibration scheme to quickly obtain a reference position.

Consider there are two speakers. Without loss of generality, we assume the two speakers are at $(0, 0)$ and $(A, 0)$, respectively. We let a user move the mobile device back and forth parallel to the x-axis (i.e., the line connecting two speakers). As the mobile is moving towards the speaker 2, it experiences a positive Doppler shift with respect to the speaker before reaching $x = A$ and experiences a negative Doppler shift after departing from it. Therefore, we can detect the time when the Doppler shift changes its sign, and at that time the mobile moves to a point on $x = A$, which is our reference point.

Now we need to determine the distance between the reference point and speakers: D_1 and D_2 in Figure 2.4. We can apply FMCW to estimate the difference between D_1 and D_2, denoted as ΔD, by having the two speakers transmit at the same time and adopting the same pseudo-transmission time. In this case, t_0, the difference between the pseudo-transmission time and the actual transmission time, is the same for both speakers. Thus, by subtracting Formula 2.4 for the two speakers, we have

$$\Delta D = D_1 - D_2 = \frac{cT(f_{p,1} - f_{p,2})}{B},$$

where $f_{p,1}$ and $f_{p,2}$ denote the peak frequencies detected by FMCW technique

for the two speakers, respectively.

Besides $D_1 - D_2 = \Delta D$, we also know that $D_1^2 - D_2^2 = A^2$ due to the property of a rectangular triangle. Then we can determine D_1 and D_2 based on these equations. To improve the accuracy, we can sweep the mobile cross the reference position multiple times, and use the mean as the estimation for D_1 and D_2.

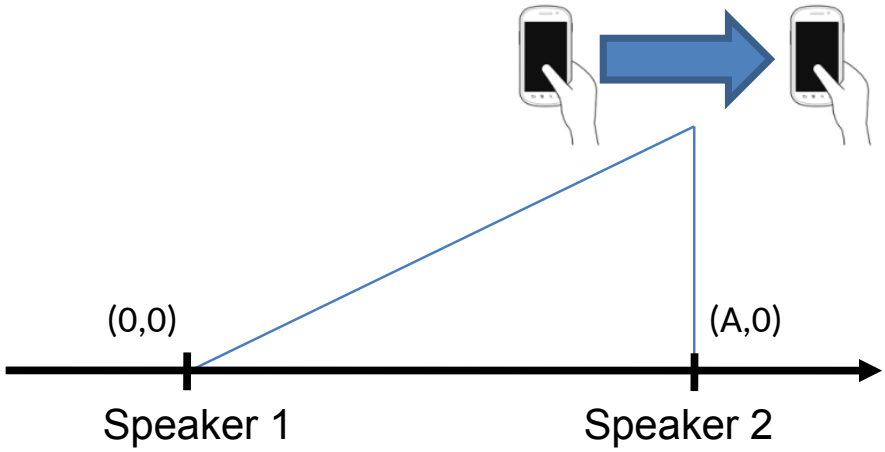


Figure 2.4: Estimating reference position.

When there are three or more speakers, the mobile can choose any position as the reference point. In this case, we can use the method discussed above to determine ΔD_{ij} between speakers i and j. Based on these ΔD's, we can solve the coordinate of the reference position, denoted as (x, y), by minimizing the following objective function:

$$(\sqrt{(x-x_1)^2+(y-y_1)^2} - \sqrt{(x-x_2)^2+(y-y_2)^2} - \Delta D_{12})^2$$
$$+(\sqrt{(x-x_1)^2+(y-y_1)^2} - \sqrt{(x-x_3)^2+(y-y_3)^2} - \Delta D_{13})^2$$

$$+(\sqrt{(x-x_2)^2+(y-y_2)^2}-\sqrt{(x-x_3)^2+(y-y_3)^2}-\Delta D_{23})^2,$$

where (x_i, y_i) is the position of i-th speaker.

Impact of movement: For clarity, we omit the impact of the receiver's movement when deriving Formula 2.5. But a non-negligible velocity will lead to an additional shift to the peak frequency of the mixed signals. In this case, the peak frequency is at:

$$f_n^p = \frac{B(R_n - ct_0)}{cT} + \frac{f_{min}v_n}{c} + \frac{Bv_n}{c}, \tag{2.6}$$

where v_n is the receiver's velocity with respect to the transmitter in the n-th interval. For ease of explanation, we assume the receiver is static in the first interval. Thus, R_n becomes:

$$R_n = (f_n^p - \frac{f_{min}v_n}{c} - \frac{Bv_n}{c} - f_1^p)\frac{cT}{B} + R_1, \tag{2.7}$$

According to the equation, the absolute distance R_n can be determined by measuring the FMCW peak frequencies in the first and n-th interval (f_n^p and f_1^p), the velocity during the n-th interval (v_n) based on the Doppler shift, and the reference distance (R_1).

Frequency offset: Due to imperfect clocks, the sampling frequencies at the transmitter and receiver are not exactly the same [13]. The frequency offset makes the sender and receiver experience different time when sending or receiving the same number of samples. This introduces an error in estimating the peak frequency of the mixed signals. Figure 2.5 shows an example to illustrate the issue, where a sender transmits a chirp consisting of 1764

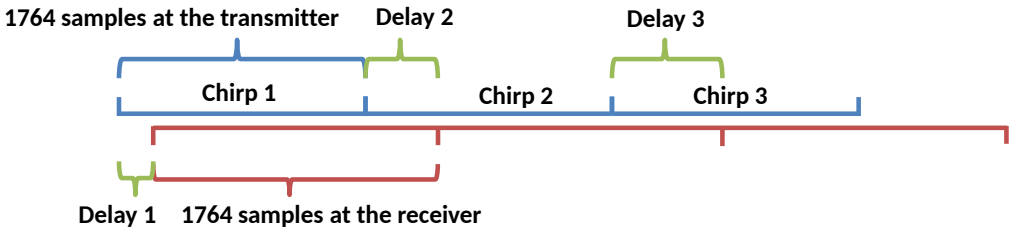


Figure 2.5: The frequency offset problem.

samples. After a propagation delay (e.g., Delay 1), the chirp arrives at the receiver. Since the receiver has a slightly different clock rate, it takes slightly longer for the receiver to accumulate these 1764 samples. Therefore, Delay 2 not only includes the propagation delay but also the difference between the transmission time and receive time for chirp 1 caused by different clock rates. Similarly, Delay 3 includes the propagation delay and the difference between the transmission and receive time for chirps 1 and 2. In general, if the sender and receiver are static and their sampling frequency offset is constant, the estimated delay will increase linearly over time.

To compensate for this effect, we introduce a short calibration phase at the beginning. We fix the receiver's location during the calibration. Without a sampling frequency offset, the peak frequency detected by FMCW should be fixed. The frequency offset will introduce a steady shift in the peak frequency over time. We can estimate the shift by plotting the peak frequency over time as shown in Figure 2.6. We then apply a least square fit to the measurement data. The slope of the line, denoted as k, captures how fast the peak changes over time due to the frequency offset. Given the estimated slope, we process

the raw measurements as follows:

$$f_p^{adjusted} = f_p^{raw} - kt, \tag{2.8}$$

where $f_p^{adjusted}$ and f_p^{raw} are the adjusted and raw peak frequencies, respectively, t is the time elapse from the start, and k is the slope of the fitted line. Figure 2.6(b) shows $f_p^{adjusted}$ is stable over time after the compensation.

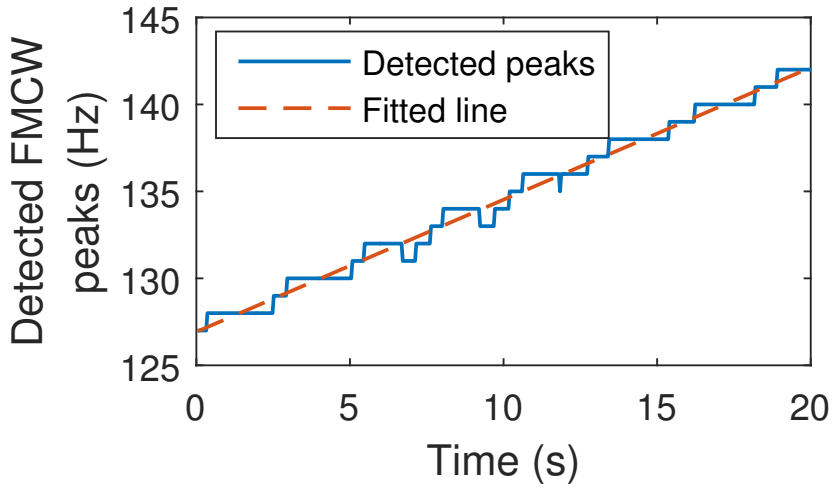
The sampling frequency offset may slowly change over time. Our experiments show that the initial estimation is valid for at least a few minutes. We can further improve the accuracy over a longer duration by re-calibrating the frequency offset whenever the receiver is stationary, which can be detected based on IMU.

2.2.2 Optimization Framework

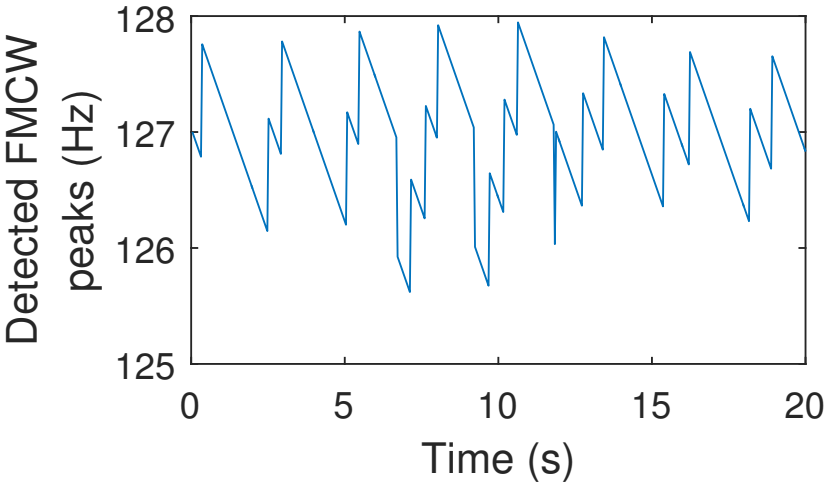
We propose the following optimization framework that combines the Doppler shift and FMCW measurements for accurate motion tracking. Specifically, we minimize the following function:

$$\begin{aligned} &\sum_{i\in[k-n+1..k]} \sum_{j} \alpha(|z_i - c_j| - |z_0 - c_j| - d_{FMCW}^{i,j})^2 + \\ &\sum_{i\in[k-n+2..k]} \sum_{j} \beta(|z_i - c_j| - |z_{i-1} - c_j| - v_{i-1,j}^{doppler} T)^2, \end{aligned} \tag{2.9}$$

where k is the current processing interval, n is the number of intervals used in the optimization, z_i denotes the mobile's position at the beginning of the i-th interval, z_0 denotes the reference position, c_j denotes the j-th speaker's position, $d_{FMCW}^{i,j}$ denotes the distance change from the reference location with



(a) Before compensation



(b) After compensation

Figure 2.6: Estimating FMCW peak shift.

respect to the j-th speaker at the i-th interval, $v_{i,j}^{doppler}$ denotes the velocity with respect to the j-th speaker during the i-th interval, T is the interval duration, and α and β are the relative weights of the measurement from FMCW and

Doppler shifts.

The only unknowns in the optimization are the mobile's position over time (i.e., z_i). The speakers' coordinates c_j can be determined using the method proposed by [49]. $d^{i,j}_{FMCW}$ and $v^{doppler}_{i,j}$ are derived from FMCW and Doppler shift measurement, respectively.

Essentially, the objective reflects the goal of finding a solution z_i that best fits the FMCW and Doppler measurements. The first term captures the distance calculated based on the coordinates should match the distance estimated from the FMCW, and the second term captures the distance traveled over an interval computed from the coordinates should match with the distance derived from the Doppler shift. Our objective consists of terms from multiple intervals to improve the accuracy. The above formulation is general, and can support both 2-D and 3-D coordinates. z_i and c_j are both vectors, whose sizes are determined by the number of dimensions.

This optimization problem is non-convex, which means that there is no guarantee on convergence and the computation cost can be high. To efficiently solve the problem, we develop an algorithm based on convex optimization. The unknowns in the original optimization problem are the node's coordinates over time. This yields a complicated objective involving $\sqrt{.}$. To simplify the objective, we use the node's distances to different speakers over time (denoted as $D_{i,j}$) as unknowns (i.e., replacing $|z_i - c_j|$ with $D_{i,j}$ in Formula 2.9), which is a convex function in terms of $D_{i,j}$. However, not all distances are feasible (i.e., there may not exist coordinates that satisfy the distance constraints). We

need to derive additional constraints to enforce feasibility. In the first step, we solve a convex relaxation of the original problem by using the distances as the unknowns and replacing feasibility constraints of the distances with triangular inequality constraints. Triangular inequality constraints are necessary in order for the distances to be realizable in a low-dimensional Euclidean space. They are also sufficient to guarantee feasibility in a 2D space, but not sufficient in a 3D space. Therefore, in the next step we project the solution obtained from the first step into a feasible solution space. This projection is related to network embedding, which embeds network hosts into a low-dimensional space while preserving their pairwise distances as much as possible. We develop a embedding method based on Alternating Direction Method of Multipliers (ADMM) [5] to efficiently solve the problem.

Our optimization has following nice properties. First, it combines FM-CW with Doppler measurement, where the former gives distance estimation without error accumulation while the latter provides more accurate distance change in a short term. Effectively combining the two allows us to achieve high tracking accuracy. Second, it uses measurements from multiple time intervals to improve the accuracy. Third, we develop an efficient algorithm to solve it. Fourth, other types of measurement can be easily incorporated in the framework to enhance the accuracy, as shown below.

2.2.3 Leveraging IMU Sensors

Next we leverage Inertial Measurement Unit (IMU) sensors along with audio signals to improve the tracking accuracy by synchronizing measurements from IMU with the audio and adding them into our optimization framework. Accelerometer is cheap in terms of hardware cost, processing overhead, and battery consumption. However, it is inaccurate for long-time tracking because double integration is needed. To limit error accumulation, we first estimate the initial velocity based on Doppler shifts and then integrate accelerations over only a short time (e.g., 360 ms in our implementation) to get the distance traveled during this period. Moreover, we use the gyroscope to measure the rotation and translate the accelerometer readings to the direction consistent to our tracking coordinate.

We add additional terms to the optimization objective and new optimization variables to incorporate the error with respect to the IMU sensors. Formula 2.10 shows the final objective. The first two terms are the same as above. Let k denote the current interval. The third term reflects the difference between the distance traveled during the past $n-1$ intervals calculated based on the mobile's coordinates versus the distance estimated from the initial velocity v^{init}_{k-n+1} and IMU sensor readings, where v^{init}_{k-n+1} is the new optimization variable and represents the initial velocity at the $(k-n+1)$-th interval. $d^{IMU}_{k-n+1,k}$ is the displacement from the start of $(k-n+1)$-th interval to the start of k-th interval, which is calculated based on IMU sensor readings assuming the initial velocity is zero. The fourth term reflects the difference between the

average velocity in the $(k-n+1)$-th interval estimated based on the IMU sensors versus based on the Doppler shift, where $v_i^{doppler}$ is the velocity in the i-th interval estimated based on the Doppler shift and is a vector, and Δv_{k-n+1}^{IMU} captures the velocity change in the $(k-n+1)$-th interval calculated based on IMU. σ and γ are the weights of the two new terms.

$$\begin{aligned}
&\sum_{i\in[k-n+1..k]}\sum_{j}\alpha(|z_i-c_j|-|z_0-c_j|-d_{FMCW}^{i,j})^2+\\
&\sum_{i\in[k-n+2..k]}\sum_{j}\beta(|z_i-c_j|-|z_{i-1}-c_j|-v_{i-1,j}^{doppler}T)^2+\\
&\sigma(z_k-z_{k-n+1}-v_{k-n+1}^{init}(n-1)T-d_{k-n+1,k}^{IMU})^2+\\
&\gamma(v_{k-n+1}^{init}+1/2\cdot\Delta v_{k-n+1}^{IMU}-v_{k-n+1}^{doppler})^2. \qquad (2.10)
\end{aligned}$$

2.2.4 Summary

Putting everything together, Procedure 1 shows the pseudo code of our final system.

Algorithm 1 CAT tracking system

1: Estimate speakers' positions as in [49];
2: Perform approximate synchronization to align the processing intervals with the received chirps;
3: Find a reference point;
4: Estimate the peak shift rate due to sampling frequency offset;
5: **while** TRUE **do**
6: Fetch next 40 ms audio signals (1764 samples);
7: Apply FFT to estimate Doppler shifts at various frequencies;
8: Combine multiple Doppler shift estimates to derive velocity;
9: Mix the received signals with the pseudo transmitted signals;
10: Apply FFT on the mixed signals;
11: Derive the distance based on peak frequency in the mixed signal, peak shift rate, reference position, and velocity;
12: Combine velocity and distance estimates (optionally with IMU) to derive the coordinates based on optimization.
13: **end while**

Chapter 3

Implementation and Applications

In this chapter, we present implementation details and the applications built on top of our motion tracking technology.

3.1 Implementation

We implement CAT on two platforms. Our first platform consists of a PC with speakers and an Android phone. The phone collects audio samples and inertial sensor data, and sends back to the PC through Android debug bridge. Our tracking program running on the PC analyzes the collected measurements to track the phone in real time. Our second platform consists of external speakers playing audio sound and a mobile phone analyzing the received audio signal to track its location in real time. We use the first platform to perform all our evaluations and use the second platform to demonstrate the feasibility of running CAT on a smartphone.

Our system separately estimates Doppler shift and propagation delay using sine waves and saw-shape chirp signals (as shown in Figure 2.1) on different frequency bands. Each Doppler measurement takes 1 KHz, which includes five

sine waves at five different frequencies separated by 200 Hz to avoid mutual interference. For each FMCW measurement, the chirp signal with 2.5 KHz bandwidth is used. These signals are generated by Matlab and saved in a standard wav audio file, which can be played directly from a general-purpose speaker.

We implement three different frequency allocations. The first one is a two-speaker system for 2D tracking. We allocate 1 KHz for each of the two speakers to continuously measure the Doppler shift. We allocate 2.5 KHz for FMCW estimation, and let two speakers alternatively send chirp sequences to share that frequency band. One speaker transmits chirps swiping from 17 KHz to 19.5 KHz, while the other transmits chirps sweeping from 19.5 KHz to 17 KHz. This helps to differentiate signals from different speakers. In addition. there is a 500 Hz guard band between the frequencies used for Doppler shift measurement and FMCW estimation. In this way, the audio signals in the two-speaker system occupy 14.5-19.5 KHz, which are virtually inaudible to most people [15]. Moreover, another benefit is that such frequency is not interfered by ambient sound. As reported in the measurements from [26], the ambient interference is almost close to noise levels beyond 6KHz and becomes negligible beyond 8KHz. Note that when the speakers alternate sending chirps, for each interval we remove the error term associated with the silent speaker during that interval from the optimization objectives (i.e., Formula 2.9 and 2.10).

The second is a three-speaker system for 2D tracking to further improve the accuracy. We let each speaker continuously send sine waves for the Doppler

shift estimation and chirp sequences for the FMCW measurement. In this case, each speaker occupies 4 KHz frequency band (i.e., 1 KHz for Doppler, 2.5 KHz for FMCW, and 0.5 KHz guard band). Including guard bands between speakers, the audio signals altogether occupy 6.5 - 19.5 KHz, which is audible. As shown in Section 4, the accuracy of our two-speaker system is comparable to that of the three-speaker system when the speakers are separated by 90 cm. When the separation reduces to 30 cm, the three-speaker system becomes substantially better.

The third is a four-speaker system for 3D tracking. In this case, the first two speakers share a 2.5 KHz frequency band for FMCW measurement by alternatively sending chirp signals. Similarly, the other two speakers use another 2.5 KHz band for FMCW estimation. In addition, each speaker is allocated 1 KHz for Doppler shift measurement. Thus, including the guard band, the audio signals occupy 9.5 KHz - 19.5KHz, which is audible.

There are several ways of fitting the audio signals from three or more speakers into inaudible band. One option is to let them alternate in sending chirp sequences and sine waves. Another option is to use ultrasound speakers and microphones, which can send and receive audio signals with frequencies higher than 20 KHz and have much wider available bandwidth. A third option is to swap the transmitters and receivers in our implementation, i.e., have the mobile transmit and use the received signals by the microphones connected to PC for tracking. The new challenge is to get separate audio streams from individual microphones since many systems output combined signals from all its

microphones. We verify it is feasible to use splitter and cable to get separate audio signals from each microphone. In this way, we can simply use 4 KHz for both the Doppler and FMCW estimation, which can easily fit into inaudible spectrum supported by the existing hardware. We can further apply the techniques in [18] to smooth transitions and make the sound even less audible.

3.2 Applications Built on Top of CAT

Based on the CAT system, we build several practical applications.

3.2.1 Audio Ruler

Audio Ruler is a distance-measuring-app which is installed on an Android smartphone. We let one speaker continuously transmit acoustic signals which we described in section 3.1. Then, users could place the smartphone at an arbitrary distance away from the speaker. The app will display the distance between the smartphone and the speaker by combining FMCW estimation with Doppler shifts and Inertial Measurement Unit. Figure 3.1 shows an example of comparing the distance displayed by Audio Ruler and the readings measured by a ruler.

3.2.2 Drawing in The Air

We also apply CAT to drawing in air. We let a user hold a smartphone and draw shapes in the air, the smartphone will draw the trajectory of its movement

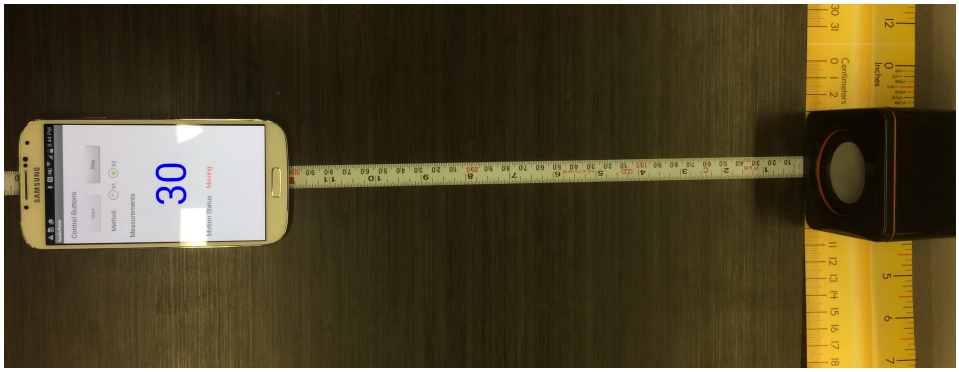


Figure 3.1: Audio Ruler

by analyzing acoustic signals received by our tracking system. Figure 3.2 shows an example of trajectory.

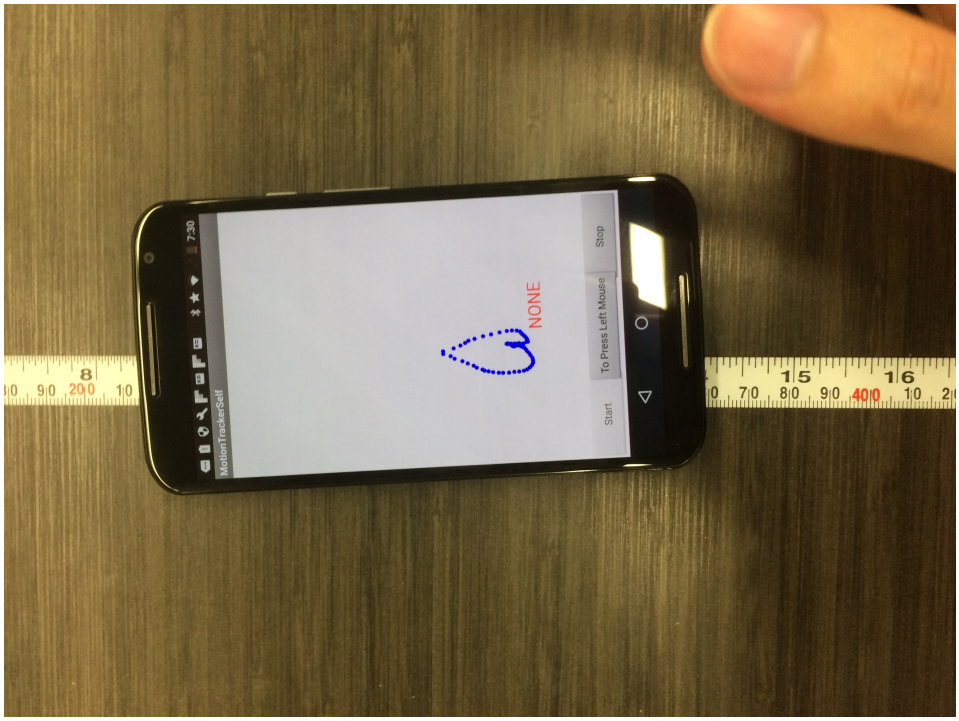


Figure 3.2: Drawing in The Air

3.2.3 Controlling Game

We further develop our tracking system as a motion controller for playing video games. In this system, we map the phone's movement into a cursor movement in games. The smartphone acts as a game controller and analyzes the audio signal transmitted by several speakers to determine its location in real-time and feeds back to the computer, which will update the cursor position in the game. There is a demo video, which shows we play a popular Android game Fruit Ninja using CAT, in [1]

Chapter 4

Evaluation

We evaluate CAT using Dell XPS X9100 desktop with Intel i7 CPU and 8GB memory as a main processor. This desktop supports at most 6 speakers. We use Logitech S120 2.0 speakers ($10 each). The speakers' volume is set to 30 out of 100 to ensure it works in a normal range. We use Nexus 4 as our mobile device, which moves within 1m/s in our experiments.

For 2D tracking, we randomly move our hand with various speeds in a horizontal plane. We compare the tracking accuracy of CAT when using 2 speakers versus 3 speakers. In addition, we also compare with the Doppler shift based approach using 2 speakers [49] and camera-based approach, the latter of which serves as the ground truth. To facilitate the camera to get the ground truth, we put a blue marker on the phone and let the camera track the blue marker. Camera is not a general tracking solution because it requires good lighting condition, a visually distinct target, and line-of-sight. We ensure all these requirements are satisfied for the vision based tracking to work well.

In our two-speaker system, the default separation between the speakers is 0.9 m. In the three-speaker system, the third speaker $S3$ is added as shown in Figure 4.1. The mobile device is 1.1 m away from the line defined by the two

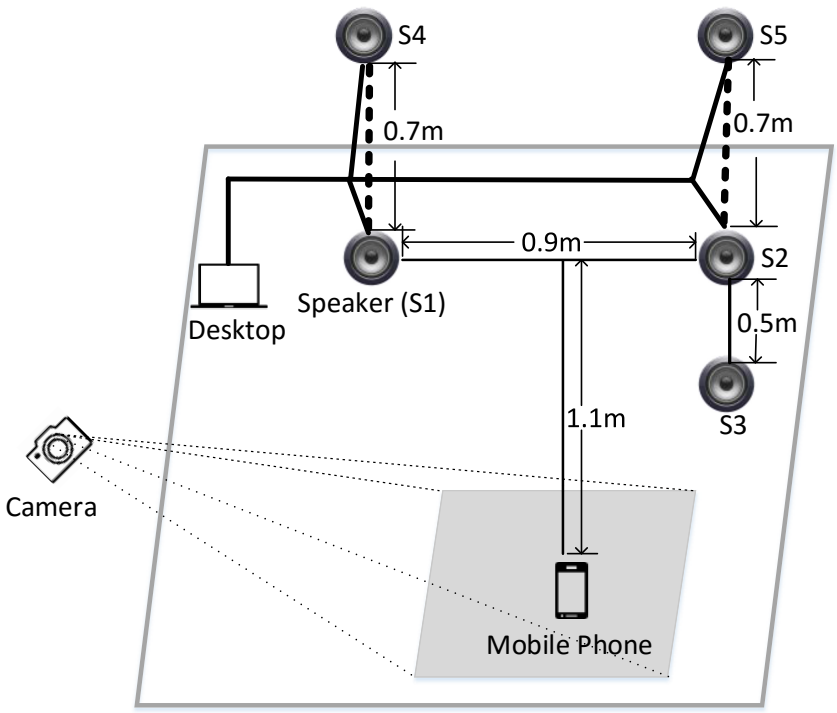
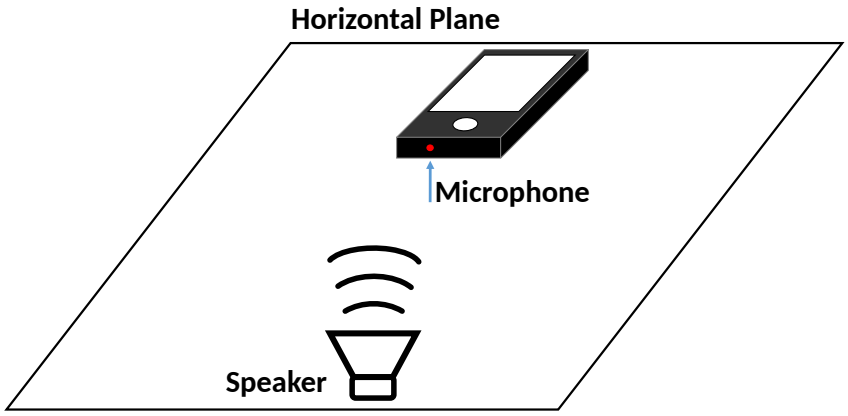


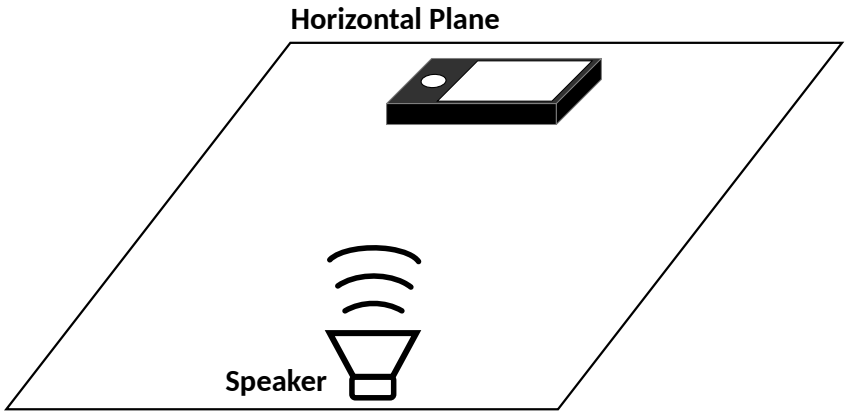
Figure 4.1: Experiment setup.

speakers. Moreover, the microphone located at the bottom of the smartphone faces the speakers, as shown in Figure 4.2(a).

For 3D tracking, we use four speakers. The placement of these speakers are indicated by S1, S2, S4, and S5, as shown in Figure 4.1, where S4 (S5) is fixed at 0.7m above S1 (S2). Since it is difficult to get the precise ground truth location in 3D space with a camera, we conduct experiments for 3D tracking in the following way. We print the trajectory of a given shape (e.g., triangle or circle) on a paper, and attach the paper to a tilted surface. As the slope of the surface is known, the position of the trajectory in 3D space can be determined. In the experiments, we move the smartphone following



(a) LOS



(b) NLOS case 3

Figure 4.2: Microphone orientation.

the trajectory on the paper and use our scheme to track the movement in the 3D space. The tracking results are compared with the printed trajectory to compute the tracking errors.

4.1 Micro Benchmark

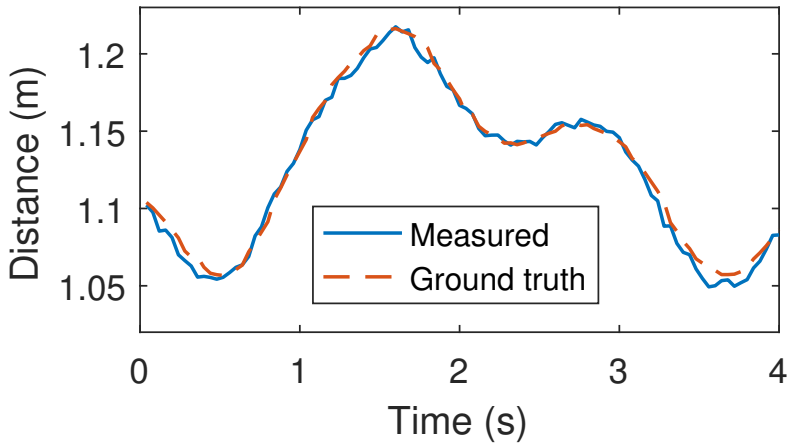
First, we present micro benchmarks.

4.1.1 Estimating Distance

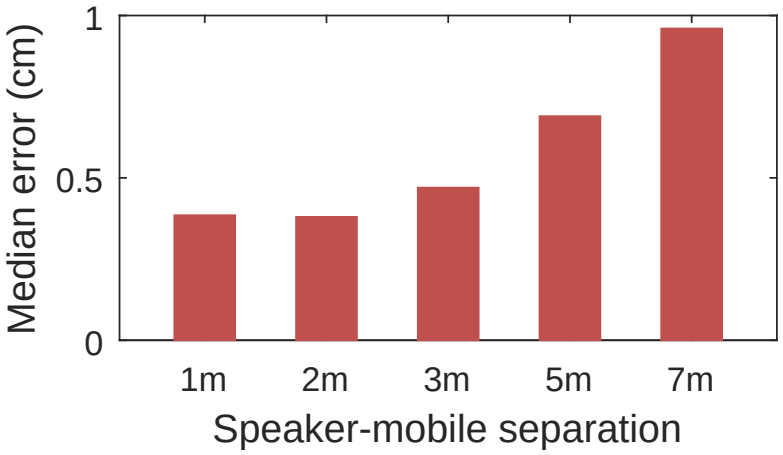
In this experiment, the distance between the speaker and the mobile is estimated based on Formula 2.7, assuming the reference point (i.e., R_1) is known in advance. Figure 4.3(a) plots the estimated distance for a portion of our trace. As we can see, the estimated results closely follow the ground truth. The median error is less than 4 mm, and 90-th percentile error is 9 mm.

Figure 4.3(b) further plots the error in the distance estimation as we vary the separation between the speaker and mobile, while the speaker volume remains unchanged (30 out of 100). The estimation error increases when the separation is larger than 3 meters, and reaches 1 cm when the separation is 7 meters. Further increasing the separation may lead to the failure of detecting FMCW peaks (i.e., f_n^p in Formula 2.7). In this case, we need to increase the speaker volume to increase the operating range.

We further evaluate the proposed scheme under no light-of-sight (LOS) between the speaker and microphone. First, we use a piece of cloth to cover the microphone to emulate the mobile is in the pocket (case 1), and find the tracking error is 4.4 mm, similar to that under LOS. In this case, the signals from all paths are attenuated by a similar amount and we can detect the correct FMCW peak and Doppler shift to accurately estimate the distance



(a) Measured trace



(b) Estimation errors

Figure 4.3: The error of estimating distance.

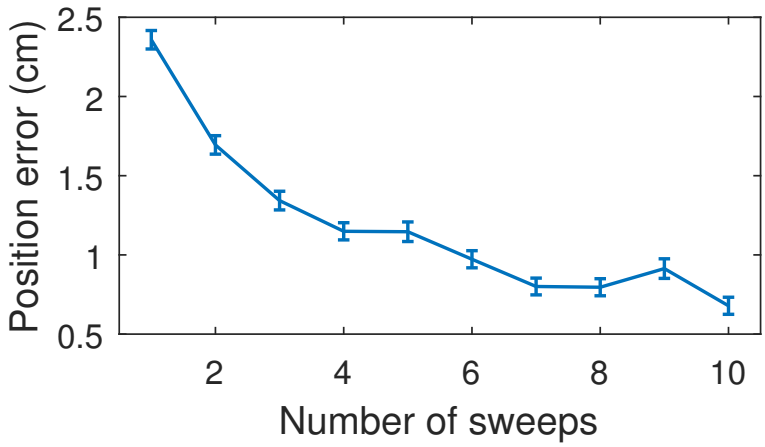
and velocity. Next, we put a small cardboard box (8 cm×5 cm×2 cm) at the middle point of the line connecting the speaker and the microphone, while the active region of our speaker is 10 cm tall and 2 cm wide (case 2). The tracking error increases to 5.9 mm, slightly higher than that under LOS. In

this case, the direct path between the speaker and microphone is blocked but the signal can arrive at the microphone through a path close to the direct path. So our system can estimate the distance and velocity with a reasonable accuracy. Then we turn the mobile 90^o away from the speaker (case 3), as shown in Figure 4.2(b). In this case, we can barely see the microphone from the speaker. The tracking error becomes 5.1 mm due to a reason similar to case 2. Further increasing the cardboard size or turning the mobile away from the speaker will degrade the tracking accuracy since both the direct path and nearby paths are blocked, which may cause an incorrect detection of FMCW peaks. As part of our future work, we plan to enhance the accuracy for these most challenging cases.

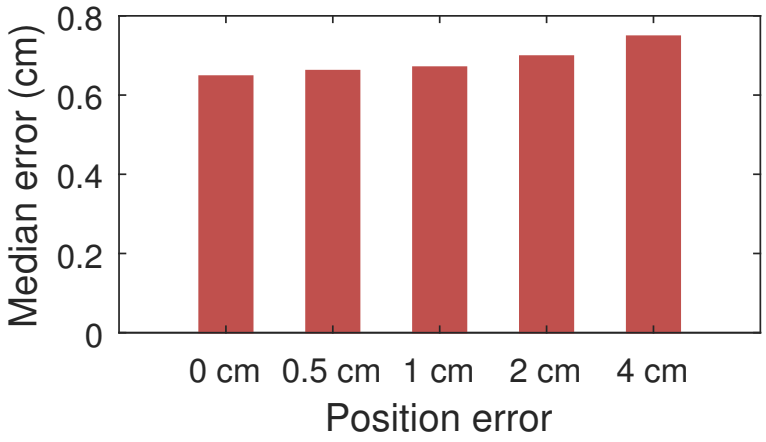
4.1.2 Estimating the Reference Position

Next we evaluate the error in estimating the reference position. Each time when we swipe across the speaker, we get one estimation of the reference position. When we swipe multiple times, the reference point is estimated as the average across all sweeps. The more times we sweep, the more accurate the estimation is. We collect a trace from 969 swipes. Then we compute the average estimation error as we vary the number of sweeps S and report the average across S sweeps. As shown in Figure 4.4(a), the error bar is centered at the mean with the length set to its standard deviation of sampled mean. The error reduces considerably as we increase the number of sweeps from 1 to 2. It continues to decrease until 4 sweeps. Afterwards, additional sweeps do

not significantly reduce the error.



(a) Error in reference position



(b) Impact of error

Figure 4.4: Results for estimating the reference position.

Figure 4.4(b) compares the median tracking error as we inject a varying amount of error to the reference position. We compute the trajectory error by shifting the entire trajectory by the error in the reference position and computing the difference between the estimated and ground-truth trajectories. To

compute the trajectory difference, for each point on the estimated trajectory, we identify the point on the ground-truth trajectory from the camera that has the closest timestamp, compute the Euclidean distance between the two points, and average over all points on the trajectory. Even with 4 cm error in the reference position, we can still achieve around 7 mm trajectory error, which demonstrates that the trajectory tracking is robust to the error in the reference position.

4.1.3 Estimating FMCW Peak Shift

Figure 4.5 shows the estimated peak shift rate (due to the sampling frequency offset) as we increase the number of chirps used for estimation. As we can see, the estimation converges when we use 50 chirps, which take 2 s in our implementation, since the chirp duration is 40 ms.

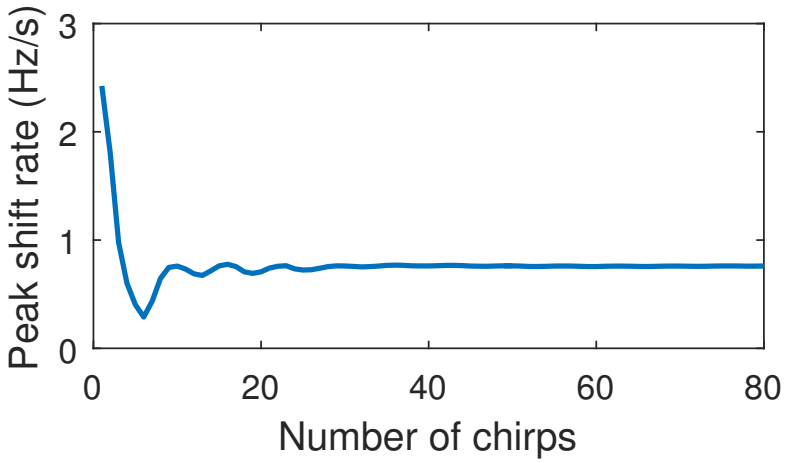


Figure 4.5: Estimating peak shift rate.

4.2 2D Tracking Accuracy

In this section, we quantify the tracking accuracy by varying a few parameters to understand their impacts. We compute the error by comparing with the ground truth obtained from the camera.

4.2.1 Impact of The Speaker Separation

We first examine the impact of the speaker separation. Figure 4.6 plots the median error as we vary the separation between the two speakers. Different separations represent various application cases: the larger separations correspond to the home theater/smart TV/ game console scenarios, whereas the small separation corresponds to VR/AR settings. For example, in the VR/AR setting, the ratio of the distance between the hand and head vs. the distance between the two speakers on the headset is around 3-4, which corresponds to the setting in the left most bar. As the figure shows, the error reduces as we increase the separation between the speakers. For example, the median error using 2 speakers is 12 mm under 30 cm separation, and reduces to 7 mm under 90 cm separation.

Figure 4.7 further plots the tracking accuracy under 3 speakers. We consider two settings. The first setting is shown in Figure 4.1. The other setting is similar but the distance between S1 and S2 and that between S2 and S3 both reduce to 30 cm. The median error reduces to 5 mm in the first setting, and to 7 mm under the second setting. Compared with the results from the two-speaker setup, the three speakers bring larger improvement under a small-

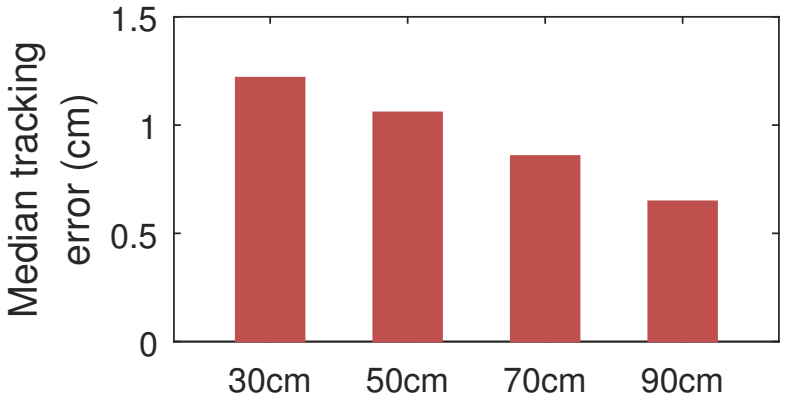
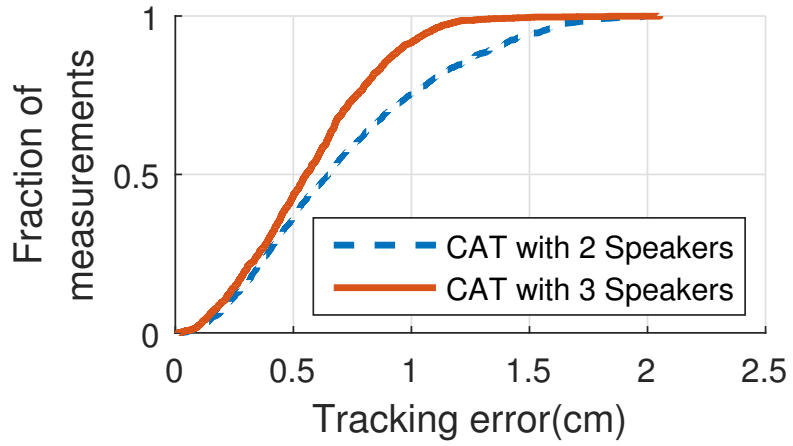


Figure 4.6: Errors with various speaker separations.

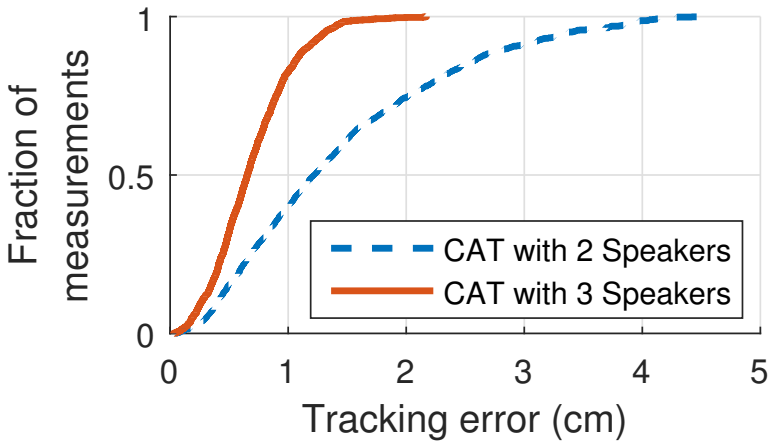
er separation, because the additional speaker helps to reduce ambiguity in the two closely located speakers.

4.2.2 Number of Intervals

Figure 4.8(a) and (b) plot the median error and running time as we vary the number of processing intervals used in our optimization, respectively. The error bars in Figure 4.8(b) are centered at the mean and the bar length reflects its standard deviation of sampled mean. For the running time, we compare two optimization solvers: 1) general non-linear solver NLopt [29] to optimize Formula 2.9 in Section 2.2.2; 2) embedding-based solver on the converted problem as proposed in Section 2.2.2. The two solvers yield the same error, but the embedding is more efficient especially under a larger number of intervals. With 10 intervals (the default value in our evaluation), the optimization time is 2 ms. In additional, it takes 2 ms to measure Doppler shift, and 3 ms to



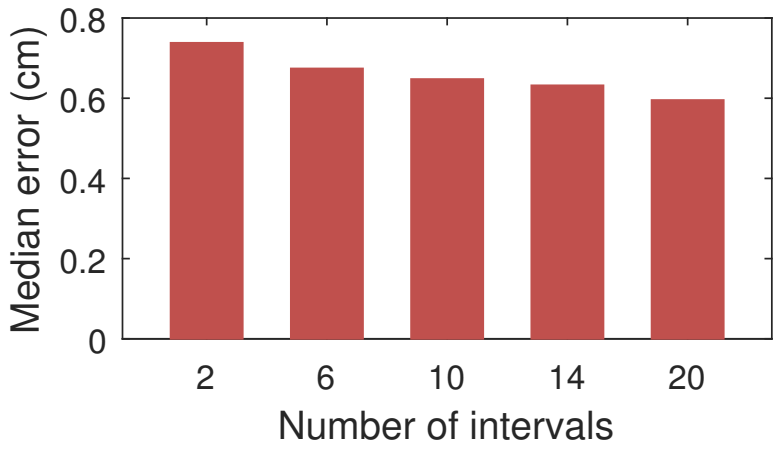
(a) 0.9 m speaker separation



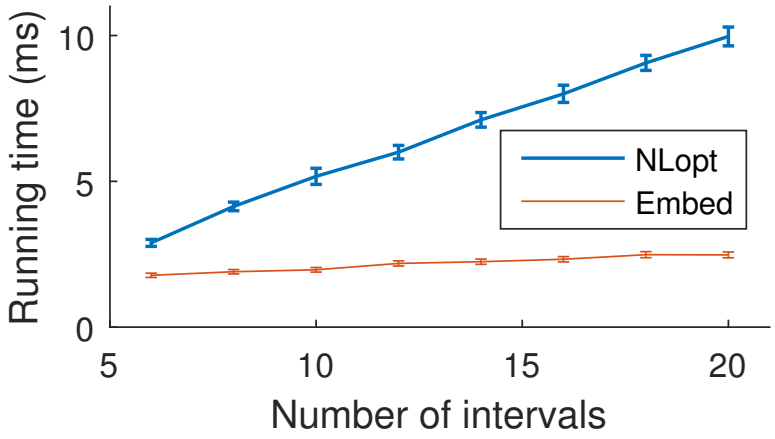
(b) 0.3 m speaker separation

Figure 4.7: 2 speakers vs 3 speakers.

measure FMCW. The total time to compute the position is 8 ms, well below our processing interval 40 ms.



(a) Tracking Error



(b) Running time

Figure 4.8: Number of intervals used in the optimization.

4.2.3 Impact of Weights

We examine the impact of the weights (α, β) in our objective function (Formula 2.9 in Section 2.2.2). Figure 4.9 plots the CDF of errors under different weights. With weight (0,1), only Doppler shift is used in our scheme. This

is essentially AAMouse [49]. With (1,0), only FMCW measurement is used and the optimization becomes finding the intersection of circles whose sizes are the distance estimates from FMCW. The other weights all use both Doppler and FMCW. As we would expect, the tracking error of using both information is significantly lower than using one of them. The weight (1,40) performs the best because it makes the two terms (after multiplying the weights) in our optimization objective have similar magnitude. However, the tracking error is not sensitive to the exact weights: a wide range of weights offer similar performance, as shown in the figure.

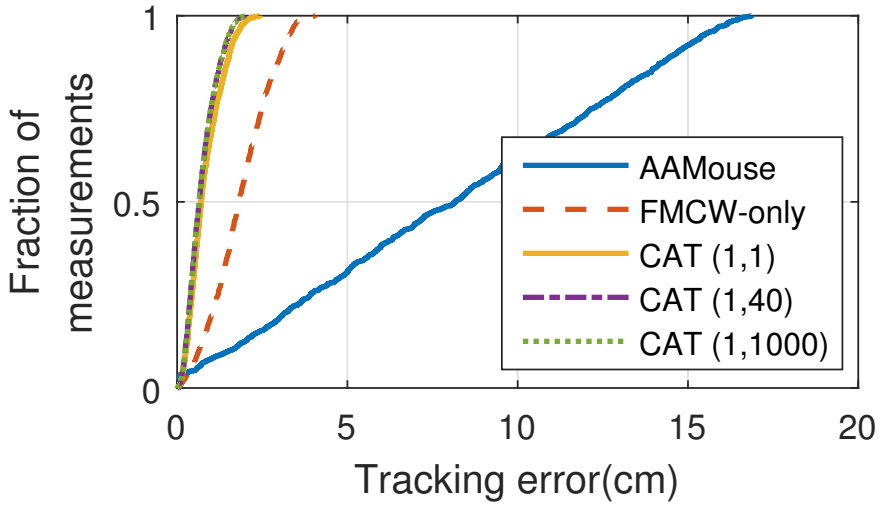


Figure 4.9: Varying the weights.

4.2.4 Leveraging the Sensors

Figure 4.10 plots the median errors with or without using IMU sensors in our two-speaker tracking system. As it shows, using IMU sensors is beneficial

to improve the tracking performance. When two speakers are separated by 90 cm, the tracking error is reduced by 3%. When the separation between the speakers decreases to 30 cm, using IMU reduces the tracking error by 6%. We expect the benefit of incorporating IMU increases as the accuracy of IMU improves. Moreover, our optimization framework is flexible to leverage other types of measurements to further improve the performance.

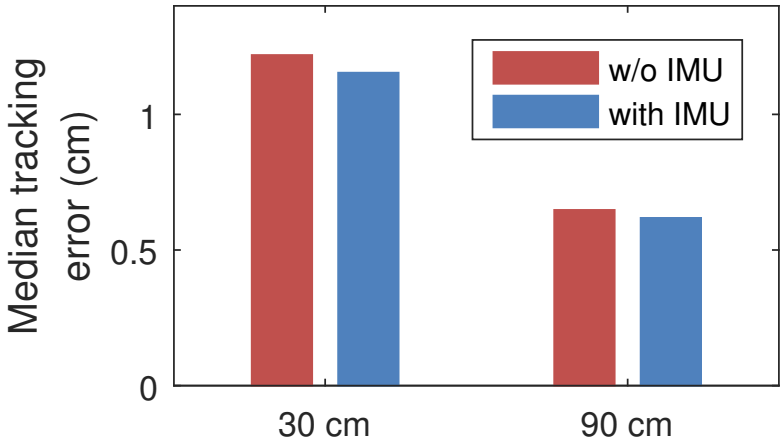


Figure 4.10: Comparison between with and without IMU.

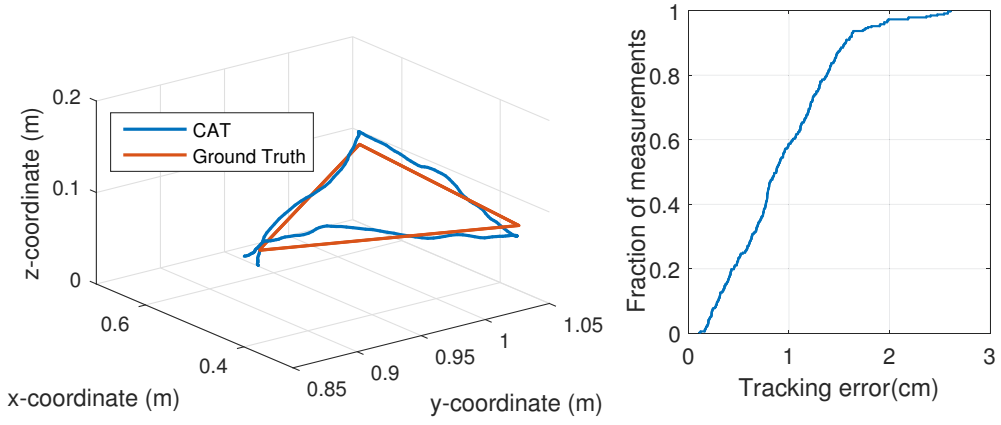
4.3 3D Tracking Accuracy

In this section, we evaluate CAT with four speakers in 3D space.

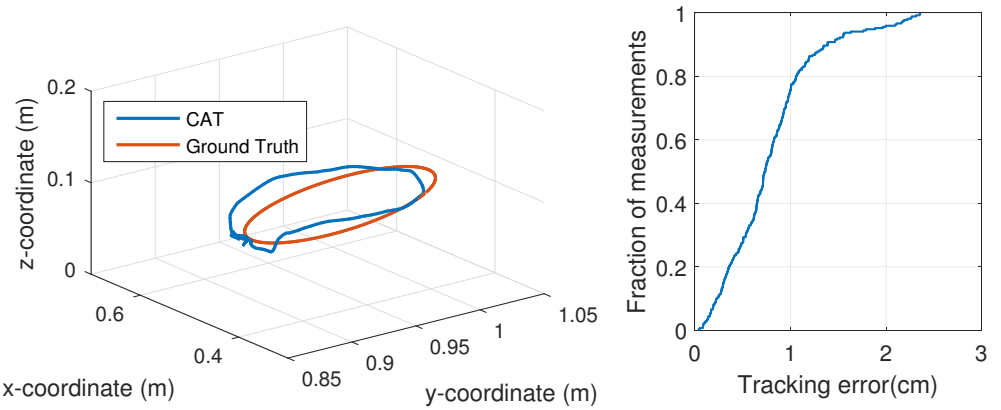
4.3.1 3D Tracking Performance

In this experiment, we evaluate the tracking performance for drawing a triangle and a circle in the 3D space. In Figure 4.11, the graphs on the left

show the trajectories tracked by CAT versus the ground truth, and the graphs on the right show the corresponding tracking errors. As we can observe, the median error for 3D tracking is about 8 mm - 9 mm, which is slightly larger than those in 2D experiments because of larger degree of freedom.



(a) Trajectory (b) Error



(c) Trajectory (d) Error

Figure 4.11: 3D tracking accuracy.

4.3.2 Error Accumulation

To check if our scheme has error accumulation, we compare the tracking errors for drawing triangles in 3D space with AAMouse [49] (using four speakers). The experiment lasts for 180 s. During this period, we keep moving the smartphone following the printed triangle trajectory. As shown in Figure 4.12, the 3D tracking error of CAT is stable over time, and significantly outperforms AAMouse. This demonstrates our method effectively addresses error accumulation problem.

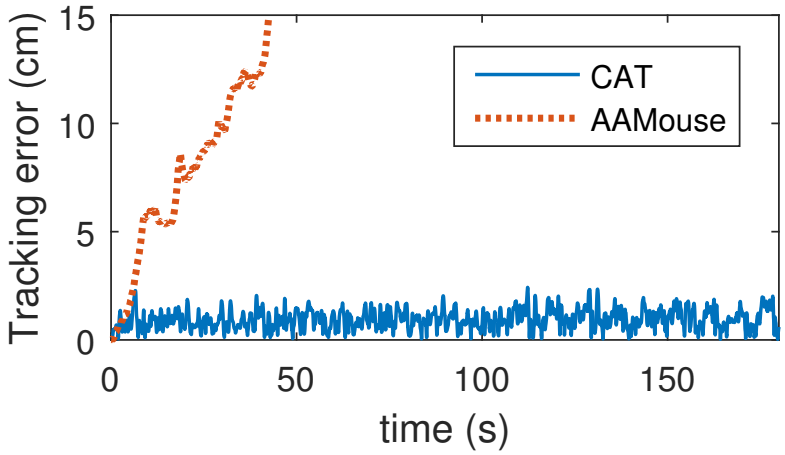


Figure 4.12: Error over time.

4.3.3 Robustness to Ambient Sound

To evaluate the robustness of our scheme to ambient sound, we continuously play music when using CAT to track the smartphone. We play several different genres of music (e.g., Jazz, Pop, and Classic) together to emulate different ambient sound. The speaker for playing music is placed near S1 as

shown in Figure 4.1. The music volume is the same as the acoustic signals for tracking. Figure 4.13 compares the tracking error with and without playing music. We observe no significant difference between two cases, which indicates that our scheme is robust to ambient sound.

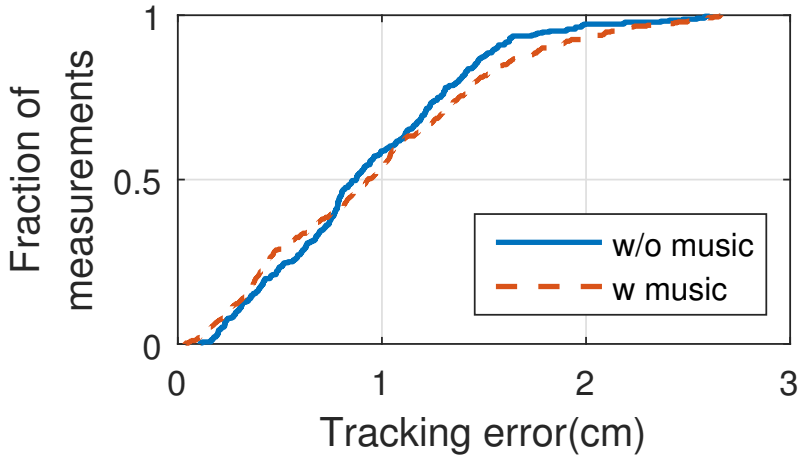


Figure 4.13: Impact of ambient sound

Chapter 5

User Study

We recruit 10 users and report evaluation results from our user study. We conduct two types of experiments. In the first experiment, we let users touch targets shown on a screen and measure the distance traveled. A shorter distance indicates more accurate tracking and easier to use. In the second experiment, users are asked to draw some shapes on the screen and we compare the measured trajectories versus the original trajectories to quantify the accuracy. We use our 2D tracking systems for user study.

In the evaluation, we run each experiment 10 times involving different users. Each user has 10-minute training for each scheme (i.e., CAT and AA-Mouse). When using CAT, the users are asked to hold the device in such a way to avoid their hands blocking the microphone. The overall experiment lasts about 1 hour for each user. The distances of our ground truth trajectories used for pointing and drawing experiments are 41.37 cm for pointing, 73.26 cm for drawing a triangle, 60.67 cm for drawing a double-circle, and 153.1 cm for drawing a loop back, as shown in Figure 5.4. The tracking and visualization are both done online in real-time. Meanwhile, we also store the complete traces to compute various performance metrics offline. For touching point, we

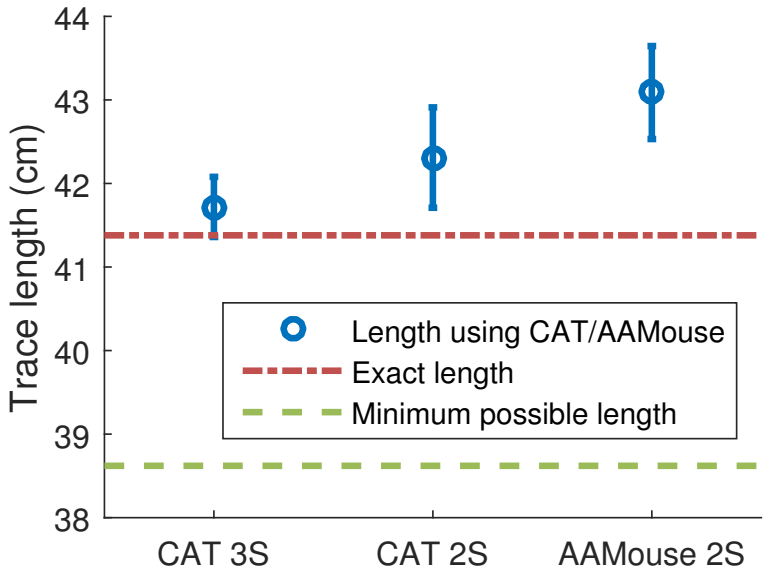


Figure 5.1: Target pointing evaluation.

compare the distance traveled in order to touch a target, which reflects the amount of user effort. For drawing shapes, we measure the error by comparing the actual pointer trajectory with the ground-truth trajectory.

5.1 Target Pointing Evaluation

We evaluate the usability of CAT as a pointing device based on the distance traveled in order to touch several targets. We let a user start from a middle point in a line, move to the right end, and then come back to the left end.

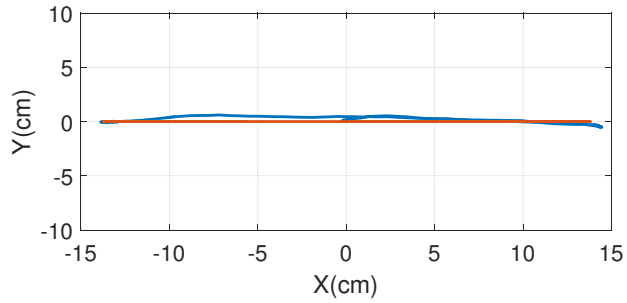
Figure 5.1 shows the average distance traveled by our scheme and AAMouse, which is a Doppler shift-based method. As a reference, we plot the exact path length to touch these points. We consider a pointer touches the

target if it is within 10 pixels (around 1.38 cm) of the target. As another reference, we also plot the minimum possible trace length, which corresponds to the exact distance - 2×1.38 = 41.37 - 2.76 = 38.62 cm. CAT with two or three speakers (labeled by "CAT 2S" and "CAT 3S", respectively in the figure) both have smaller distance than the Doppler shift based approach. We expect the gap between CAT and Doppler based scheme will increase as the number of points to touch increases due to error accumulation in the latter.

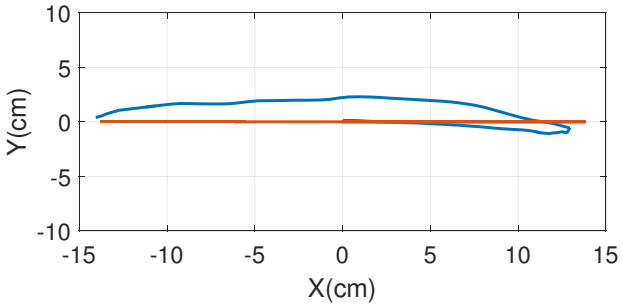
Figure 5.2 plots the trajectories that correspond to the median performance for each scheme. Even though the distances traveled by both schemes do not differ significantly, there is a clear difference between their trajectories. CAT closely follows the ideal trajectory. The trajectory of CAT under 3 speakers (not shown) is even better. In comparison, even though the Doppler based scheme is already close to the ideal trajectory, there is still clear deviation.

5.2 Drawing Evaluation

Next we ask a user to draw simple shapes: a double-circle, triangle, and loop back, shown on the screen using the pointer controlled by CAT or AA-Mouse [49]. We measure the quality of the drawings by calculating the distance between the drawn figure and the original shape. For each point in the original figure, we calculate its distance to the closest point in the drawing, and average across all points. While this does not perfectly capture the quality of the drawing, it provides reasonable distinction between well-drawn and poorly-drawn figures.



(a) CAT (2 speakers)

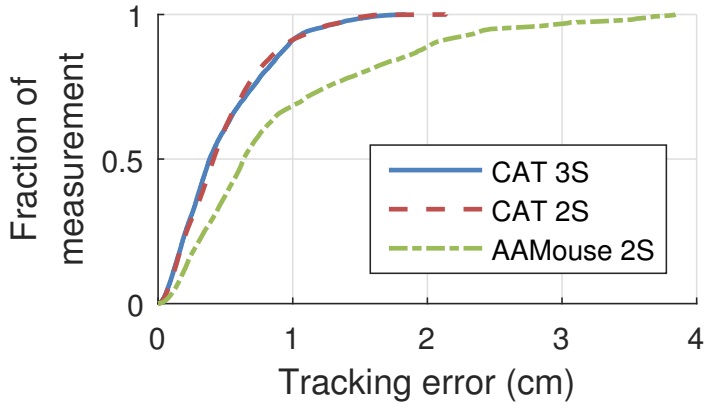


(b) AAMouse (2 speakers)

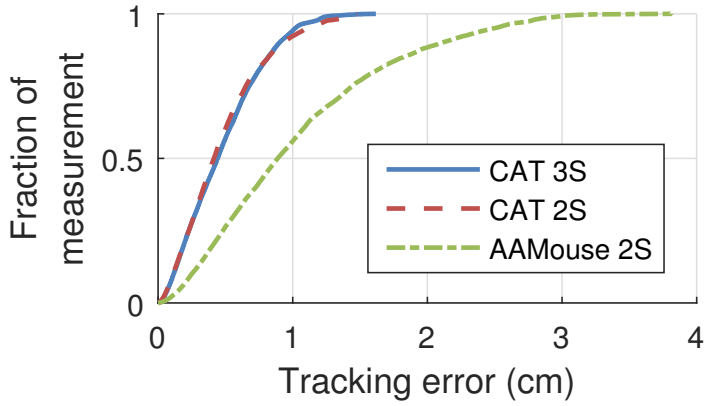
Figure 5.2: Trajectories for pointing target.

Figure 5.3 shows the CDF of the drawing error. CAT yields significantly lower error than AAMouse. For example, the median error for drawing loop back with CAT are 3.8 mm (3 speakers), 4.3 mm (2 speakers), while that for Doppler-based tracking is 11.2 mm. It is interesting to see that the tracking errors in user study are usually smaller than those presented in Section 4.2. This is because that in the user study users can adjust their movement when they see the trajectory deviate from the ground-truth.

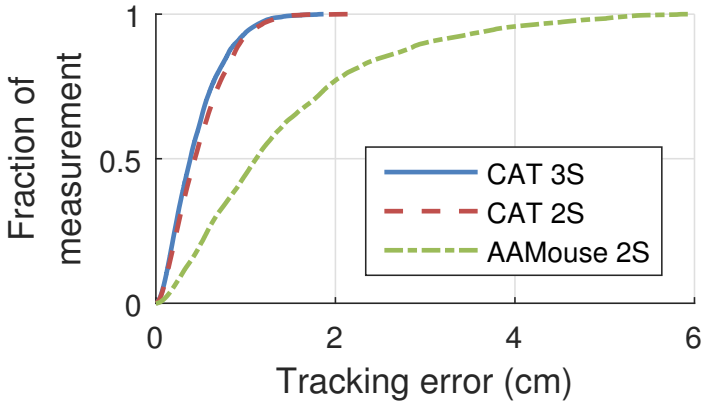
Figure 5.4 further plots trajectories drawn by the two schemes corresponding to the median error. As before, it is evident that CAT consistently follows



(a) Double circle



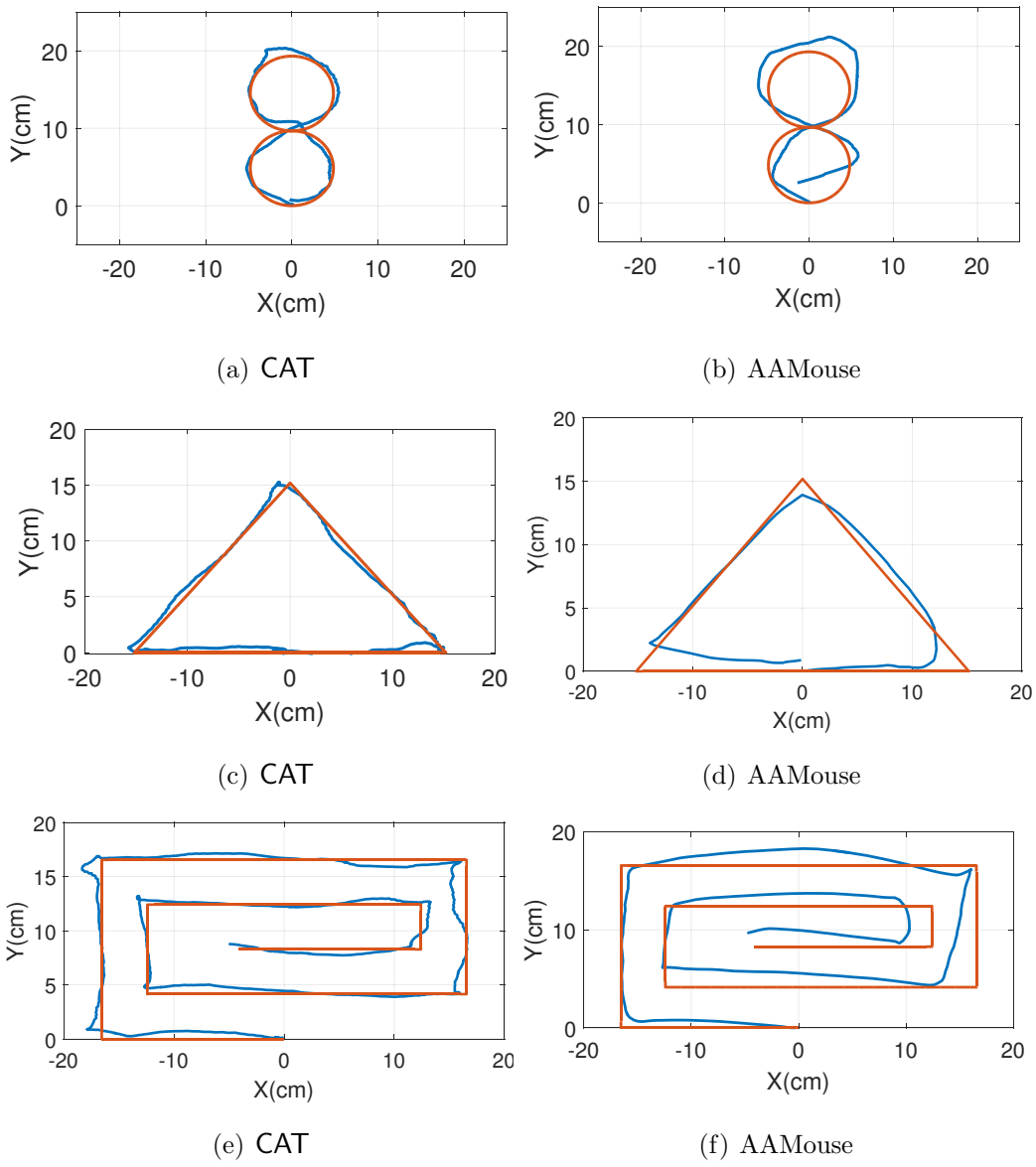
(b) Triangle



(c) Loop back

Figure 5.3: CDF of drawing error.

the original shapes much closer than the Doppler based scheme.



(a) CAT (b) AAMouse

(c) CAT (d) AAMouse

(e) CAT (f) AAMouse

Figure 5.4: Patterns drawn by CAT (2 speakers) and AAMouse (2 speakers) corresponding to the median error.

5.3 Mobile Phone Implementation

We also implement CAT on a mobile phone (Nexus 4). The phone uses CAT to efficiently track its own location using fast signal processing in [24] and the optimization solving algorithm mentioned in Section 2.2.2. The total time required to process audio samples and determine the position is 31 ms, lower than our processing interval (40ms). The CPU usage is around 35%. The tracking accuracy of the mobile phone is the same as that of the desktop version, because the only difference between the two versions is where the signal processing and computation are performed.

We make the phone running CAT to serve as a motion controller for video games, including Crossy Road [45] and Fruit Ninja [37] by mapping the phone's movement into a cursor movement in games using Windows API *mouse_event*. We ask 5 users to use our motion controller to play these games, and they find the performance is comparable to a traditional mouse.

Chapter 6

Related Work

We classify the related work based on the types of the signals and underlying techniques used for tracking and localization.

6.1 Audio Based Schemes

Audio signals are attractive for localization and tracking due to its slow propagation speed, which improves accuracy. Cricket [36] uses a combination of RF and ultrasound, and achieves a median error of 12 cm with 6 beacon nodes. Compared with Cricket, CAT improves the accuracy, and removes the need of dense deployment and special hardware. [32] develops a novel scheme that can estimate the propagation delay by having both ends send and receive audio signals to cancel out the processing time and clock difference. Based on [32], [51] develops a series of system approaches to make accurate distance ranging for mobile gaming. Similar to [32], [51] relies on cross-correlation to determine the propagation delay. To achieve high accuracy, 10-16 KHz bandwidth is used in [51]. In comparison, FMCW can achieve more accurate estimation of propagation delay using more narrow bandwidth (e.g., 2.5 KHz). FingerIO [28] develops a novel device-free tracking scheme to track a moving

finger near a smartphone or a smartwatch. CAT differs from FingerIO in that it is a device based tracking and works for a larger distance (e.g., a few meters), but faces synchronization problem that does not exist in device-free tracking. AAMouse [49] is closest to this paper. Different from [49], we estimate the distance using a new FMCW-based approach in addition to velocity measurement and fuse the two using an effective optimization framework to enhance the accuracy and minimize error accumulation.

6.2 RF-Based Schemes

RF has been widely used for localization and tracking. ArrayTrack [47] is a pioneering fine-grained tracking system based on WiFi by using an array of antennas. It achieves a median error of 23 cm using 16 antennas. RF-IDraw [41] achieves high resolution and low ambiguity by placing 8 RFID antennas with different spacing. Its median error is 3.7 cm. WiDraw [38] enables hand-free drawing in the air by estimating angle of arrival (AoA) based on CSI. Its median error is within 5 cm when using 25 WiFi transmitters. mTrack [43] achieves high tracking accuracy by leveraging the phase of 60 GHz RF signals as well as sophisticated hardware (e.g., highly directional and steerable 60 GHz antennas). Tagoram [48] uses commercial off-the-shelf RFID for localization and tracking. When the target moves along an unknown track (as in our context), the median error is 12 cm. In comparison, our system can run on commodity hardware and achieve higher accuracy.

6.3 Other Sensor Based Schemes

IMU sensors can also be used for motion tracking. However, its tracking error accumulates rapidly over time due to noisy measurements and the need of double integration [49]. Kinect [2] uses depth sensors and Wii [3] uses infrared cameras to track movement. They both require line-of-sight and have limited accuracy. LeapMotion [19] uses sophisticated vision techniques to recognize a wide range of gestures. Compared with the vision based techniques, audio-based approaches are generally more efficient and flexible: its signal processing cost is low and it works under different lighting conditions and often without line-of-sight (e.g., under small obstacles since there exists a detour path close to the direct one or obstacles that do not significantly attenuate the audio signal, such as cloth and paper).

6.4 FMCW-Based Schemes

FMCW based technique has been used for localization and motion tracking. Most existing schemes use co-located transmitter and receiver sharing the same clock. For example, [4] applies RF FMCW signals to 3D device-free tracking and achieves the tracking error of 10-20 cm. [27] leverages acoustic FMCW to detect chest and abdomen movements, which is used to identify sleep apnea.

FMCW has also been applied to the systems with separate transmitters and receivers. Such systems tend to have larger range and stronger SNR.

However, most of these systems assume strict time synchronization between the transmitter and the receiver without studying how to achieve synchronization (e.g., [11, 17, 21]). [42, 44] study synchronization for distributed FMCW systems. GPS signals have been considered for synchronization, which is very expensive. In general, synchronization significantly complicates the design and implementation of a distributed system. Therefore, we use relative distance change and reference point localization to avoid the need of synchronization.

There are a couple of schemes that do not need synchronization. In [8, 46], a separate transmitter and receiver use FMCW to track another moving target. In this case, chirp signals arrive at the receiver via two paths: the direct path from the transmitter to the receiver, and the reflected path from the transmitter to the target and finally reaching the receiver. FMCW is used to estimate the difference between the propagation delay of the two paths. The length of the reflected path via the target is determined based on the length of the direct path, which is assumed to be known. When multiple receivers exist, the target's position can be estimated based on the length of reflected paths to different receivers. In our scenario, the target to be tracked is the receiver itself and the above scheme is not applicable. [7, ?] develop FMCW to measure the time difference of arrival (TDOA) from different transmitters to the receiver. Since such methods give the difference between the distances to different anchors, they require more anchor nodes than the number of dimensions for localization, whereas our approach derives absolute distance from each anchor and the number of anchor nodes required is equal to the number

of dimensions. [12] measures the round-trip time by letting one node send and another node respond upon receiving the signals. This approach may lead to significant errors in many systems, including smartphones, since they cannot precisely control the transmission time or determine the exact receiving time. Moreover, since the scheme in [12] requires both nodes to send, receive, and process signals whereas our approach only requires a node to either transmit or receive and process the signals, our approach is more widely applicable.

Some radars use a triangular chirp based FMCW for joint distance and velocity estimation [33], where the chirp frequency increases over time in the first half and then decreases in the second half. The first and second halves of the received chirp signal experience different frequency shifts. When we observe two peak frequencies in the spectrum of the mixed signal, we can derive distance based on the average of the two frequencies and derive velocity based on their difference. However, when the chirp duration is short, the frequency domain resolution is limited and two peaks may merge together, which makes the joint estimation impossible. Based on our experiments, we find that two peaks will merge together when the chirp is within 200 ms, and become fully separated when the chirp is longer than 1 s. Such a long chirp duration is not acceptable for tracking, since it leads to significant processing delay. Therefore, joint estimation is not applicable to our context. Moreover, our distributed FMCW and optimization framework are general, and can support different waveforms, including the triangular waveform.

6.5 Gesture Recognition

The goal of gesture recognition is to determine which gesture best matches the current measurement. Hence it requires training data collected from pre-defined gestures. IMU sensors are widely used for gesture recognition [6, 16, 20, 31]. They can also be combined with other measurements, such as EMG sensors [50] and ultrasound [30], to further improve the performance. WiSee [34] proposes a novel device-free gesture recognition based on WiFi signals. [9, 10, 14, 39] use the Doppler shift of the audio signal for gesture recognition. In general, continuous tracking is more challenging due to the lack of training data or patterns to match against.

Chapter 7

Conclusion

This paper presents a novel tracking system. At its core is a new distributed FMCW-based distance estimation that (i) supports a separate sender and receiver, (ii) localizes a reference point to translate the relative distance to the absolute distance at new locations, (iii) explicitly takes into account the additional frequency shift caused by the movement, and (iv) takes into account the sampling frequency offset between the sender and receiver. We further combine FMCW with the Doppler shift and IMU sensors over multiple time intervals to continuously track the mobile. We further develop three applications based on CAT: (i) an app can measure the distance between the speaker and the phone, (ii) an app for drawing in air, (iii) and an app serving as a controller for video games. Our evaluation shows that CAT achieves mm-level accuracy and ease of use using existing hardware.

Our future work includes further enhancing the robustness of our tracking system for a wide range of usage scenarios including varying degrees of multipath, and developing applications that leverage such capabilities.

Index

Bibliography

[1] Cat demo. https://www.youtube.com/watch?v=oed0pzzdLxo.

[2] Microsoft X-box Kinect. http://xbox.com.

[3] Nintendo Wii. http://www.nintendo.com/wii.

[4] Fadel Adib, Zach Kabelac, Dina Katabi, and Rob Miller. WiTrack: motion tracking via radio reflections off the body. In *Proc. of NSDI*, 2014.

[5] Augmented lagrangian method. http://en.wikipedia.org/wiki/Augmented_Lagrangian_method.

[6] Sandip Agrawal, Ionut Constandache, Shravan Gaonkar, Romit Roy Choudhury, Kevin Caves, and Frank DeRuyter. Using mobile phones to write in air. In *Proc. of ACM MobiSys*, pages 15–28, 2011.

[7] Belal Al-Qudsi, Mohammed El-Shennawy, Yan Wu, Niko Joram, and Frank Ellinger. A hybrid TDoA/RSSI model for mitigating NLOS errors in FMCW based indoor positioning systems. In *Proc. of IEEE 11th Ph. D. Research in Microelectronics and Electronics (PRIME)*, pages 93–96, 2015.

[8] Matthew Ash, Matthew Ritchie, Kevin Chetty, and Paul V Brennan. A new multistatic FMCW radar architecture by over-the-air deramping. *IEEE Sensors Journal*, 15(12):7045–7053, 2015.

[9] Md Tanvir Islam Aumi, Sidhant Gupta, Mayank Goel, Eric Larson, and Shwetak Patel. Doplink: Using the Doppler effect for multi-device interaction. In *Proc. of ACM UbiComp*, 2013.

[10] Ke-Yu Chen, Daniel Ashbrook, Mayank Goel, Sung-Hyuck Lee, and Shwetak Patel. Airlink: Sharing files between multiple devices using in-air gestures. In *Proc. of ACM Ubicomp*, 2014.

[11] TE Derham, S Doughty, K Woodbridge, and CJ Baker. Design and evaluation of a low-cost multistatic netted radar system. *IET Radar, Sonar & Navigation*, 1(5):362–368, 2007.

[12] Roland Gierlich, Jörg Hüttner, Alexander Dabek, and Mario Huemer. Performance analysis of FMCW synchronization techniques for indoor radiolocation. In *Proc. of European Conference on Wireless Technologies*, pages 24–27, 2007.

[13] Shyamnath Gollakota and Dina Katabi. Zigzag decoding: Combating hidden terminals in wireless networks. In *Proc. of ACM SIGCOMM*, 2008.

[14] Sidhant Gupta, Daniel Morris, Shwetak Patel, and Desney Tan. Soundwave: using the Doppler effect to sense gestures. In *Proc. of the ACM*

CHI, pages 1911–1914, 2012.

[15] Pravein Govindan Kannan, Seshadri Padmanabha Venkatagiri, Mun Choon Chan, Akhihebbal L Ananda, and Li-Shiuan Peh. Low cost crowd counting using audio tones. In *Proc. of the 10th ACM Conference on Embedded Network Sensor Systems*, pages 155–168, 2012.

[16] Ji-Hwan Kim, Nguyen Duc Thang, and Tae-Seong Kim. 3-d hand motion tracking and gesture recognition using a data glove. In *2009 IEEE International Symposium on Industrial Electronics*, pages 1013–1018, 2009.

[17] Krzysztof Kulpa. Continuous wave radars–monostatic, multistatic and network. In *Advances in Sensing with security applications*, pages 215–242. Springer, 2006.

[18] Patrick Lazik and Anthony Rowe. Indoor pseudo-ranging of mobile devices using ultrasonic chirps. In *Proc. of Sensys*, 2012.

[19] Leap motion. `https://www.leapmotion.com/`.

[20] Jiayang Liu, Lin Zhong, Jehan Wickramasuriya, and Venu Vasudevan. uWave: Accelerometer-based personalized gesture recognition and its applications. *Pervasive and Mobile Computing*, 5(6):657–675, 2009.

[21] Y Liu, Y Kai, R Wang, O Loffeld, X Wang, et al. Model and signal processing of bistatic frequency modulated continuous wave synthetic aperture radar. *IET Radar, Sonar & Navigation*, 6(6):472–482, 2012.

[22] Wenguang Mao, Jian He, and Lili Qiu. Cat: high-precision acoustic motion tracking. In *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking*, pages 69–81. ACM, 2016.

[23] Wenguang Mao, Jian He, Huihuang Zheng, Zaiwei Zhang, and Lili Qiu. High-precision acoustic motion tracking: demo. In *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking*, pages 491–492. ACM, 2016.

[24] Wenguang Mao and Lili Qiu. Efficient implementation of synchronization and FFT for mobile. `http://www.cs.utexas.edu/~wmao/ReferenceLink/CATEff.pdf`.

[25] Robert A Meyers. Encyclopedia of physical science and technology. *Facts on File*, 1987.

[26] Rajalakshmi Nandakumar, Krishna Kant Chintalapudi, Venkat Padmanabhan, and Ramarathnam Venkatesan. Dhwani: Secure peer-to-peer acoustic NFC. In *Proc. of ACM SIGCOMM*, 2013.

[27] Rajalakshmi Nandakumar, Shyam Gollakota, and Nathaniel Watson. Contactless sleep apnea detection on smartphones. In *Proc. of ACM MobiSys*, 2015.

[28] Rajalakshmi Nandakumar, Vikram Iyer, Desney Tan, and Shyamnath Gollakota. FingerIO: Using active sonar for fine-grained finger tracking. In *Proc. of ACM CHI*, pages 1515–1525, 2016.

[29] NLopt. http://ab-initio.mit.edu/wiki/index.php/Main_Page.

[30] Georg Ogris, Thomas Stiefmeier, Holger Junker, Paul Lukowicz, and Gerhard Troster. Using ultrasonic hand tracking to augment motion analysis based recognition of manipulative gestures. In *Proc. of 9th IEEE International Symposium on Wearable Computers (ISWC)*, pages 152–159, 2005.

[31] Taiwoo Park, Jinwon Lee, Inseok Hwang, Chungkuk Yoo, Lama Nachman, and Junehwa Song. E-gesture: a collaborative architecture for energy-efficient gesture recognition with hand-worn sensor and mobile devices. In *Proc. of ACM SenSys*, pages 260–273, 2011.

[32] Chunyi Peng, Guobin Shen, Yongguang Zhang, Yanlin Li, and Kun Tan. BeepBeep: a high accuracy acoustic ranging system using COTS mobile devices. In *Proc. of ACM SenSys*, 2007.

[33] Diego Pierrottet, Farzin Amzajerdian, Larry Petway, Bruce Barnes, George Lockard, and Manuel Rubio. Linear FMCW laser radar for precision range and vector velocity measurements. In *MRS Proc.*, volume 1076, pages 1076–K04. Cambridge Univ. Press, 2008.

[34] Qifan Pu, Sidhant Gupta, Shyam Gollakota, and Shwetak Patel. Whole-home gesture recognition using wireless signals. In *Proc. of ACM MobiCom*, 2013.

[35] Haroon Rashid and Ashok Kumar Turuk. Dead reckoning localisation technique for mobile wireless sensor networks. *IET Wireless Sensor Systems*, 2015.

[36] Adam Smith, Hari Balakrishnan, Michel Goraczko, and Nissanka Priyantha. Tracking moving devices with the Cricket location system. In *Proc. of ACM MobiSys*, 2005.

[37] Halfbrick Studios. Fruit ninja. `http://fruitninja.com/`.

[38] L. Sun, S. Sen, D. Koutsonikolas, and K. Kim. WiDraw: enabling hands-free drawing in the air on commodity WiFi devices. In *Proc. of ACM MobiCom*, 2015.

[39] Zheng Sun, Aveek Purohit, Raja Bose, and Pei Zhang. Spartacus: spatially-aware interaction for mobile devices through energy-efficient audio sensing. In *Proc. of ACM Mobisys*, pages 263–276, 2013.

[40] Hans-Jaochim von der Hardt, Didier Wolf, and Rene Husson. The dead reckoning localization system of the wheeled mobile robot romane. In *Proc. of IEEE/SICE/RSJ International Conference on Multisensor Fusion and Integration for Intelligent System*, 1996.

[41] Jue Wang, Deepak Vasisht, and Dina Katabi. RF-IDraw: virtual touch screen in the air using RF signals. In *Proc. of ACM SIGCOMM*, 2014.

[42] Wen-Qin Wang and Huaizong Shao. Performance prediction of a synchronization link for distributed aerospace wireless systems. *The Scientific World Journal*, 2013, 2013.

[43] Teng Wei and Xinyu Zhang. mTrack: high precision passive tracking using millimeter wave radios. In *Proc. of ACM MobiCom*, 2015.

[44] Matthias Weib. Synchronisation of bistatic radar systems. In *Proc. of IEEE Geoscience and Remote Sensing*, volume 3, pages 1750–1753, 2004.

[45] Hipster Whale. Crossy road. `http://www.crossyroad.com/`.

[46] Nicholas J Willis and Hugh D Griffiths. *Advances in bistatic radar*, volume 2. SciTech Publishing, 2007.

[47] Jie Xiong and Kyle Jamieson. Arraytrack: A fine-grained indoor location system. In *Proc. of NSDI*, pages 71–84, 2013.

[48] Lei Yang, Yekui Chen, Xiang-Yang Li, Chaowei Xiao, Mo Li, and Yunhao Liu. Tagoram: Real-time tracking of mobile RFID tags to high precision using COTS devices. In *Proc. of ACM MobiCom*, 2014.

[49] Sangki Yun, Yi chao Chen, and Lili Qiu. Turning a mobile device into a mouse in the air. In *Proc. of ACM MobiSys*, May 2015.

[50] Xu Zhang, Xiang Chen, Yun Li, Vuokko Lantz, Kongqiao Wang, and Jihai Yang. A framework for hand gesture recognition based on accelerometer and EMG sensors. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 41(6):1064–1076, 2011.

[51] Zengbin Zhang, David Chu, Xiaomeng Chen, and Thomas Moscibroda. Swordfight: Enabling a new class of phone-to-phone action games on commodity phones. In *Proc. of ACM MobiSys*, 2012.

Vita

Huihuang Zheng (郑辉煌) was born in Xianyou County, Putian Xity, Fujian Province, China on April 22nd, 1993, the son of Liangrong Zheng (郑良荣) and Xiufeng Guo (郭秀凤). He received his Bachelor of Science degree in Computer Science from Shanghai Jiao Tong University. Then, he applied to the University of Texas at Austin for enrollment in their Master of Science in Computer Science program. He was accepted and started graduate studies in August, 2015. After graduation, he will work as a software engineer at Google Inc in Mountain View, California on July 2017. For updated informaction, check `https://zhhsplendid.github.io/`

Address: huihuang@utexas.edu

This thesis was typeset with LaTeX[†] by the author.

[†]LaTeX is a document preparation system developed by Leslie Lamport as a special version of Donald Knuth's TeX Program.