PUSH THE LIMIT OF IOT SYSTEM DESIGN ON MOBILE DEVICES

By

Manni Liu

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Computer Science—Doctor of Philosophy

2023

ABSTRACT

Internet of Things (IoT) utilizes sensors as the information source of machine intelligence. Its applications widely exist from Smart Home, Smart City to Wearable Healthcare and Smart Farming. An IoT architecture usually covers four stages: sensor data connection, data transmission, data processing and application model. On top of prediction precision, the interest of IoT research includes efficiency, economic saving and system scalability.

In pursuit of these goals, we push the limit of IoT system design from the following three perspectives. (1) We exploit the potential of sensors of smart devices, including sensor fusion and possibility of new IoT applications. (2) We design Machine Learning models for IoT applications, including feature engineering and model selection. (3) We implement lightweight IoT systems for smart devices like laptops, smartphones and voice assistants, considering the constraint of computation resources.

In this dissertation, we especially introduce our effort to IoT applications related to localization and security. EyeLoc is a smartphone vision enabled localization we designed for large shopping malls. The results show that the 90-percentile errors of localization and heading direction are 5.97 m and 20° in a 70,000 m^2 mall. Patronus protects acoustic privacy from malicious secret audio recordings using the nonlinear effect of microphones. Our experiments show that only 19.7% of the words protected by Patronus can be recognized by unauthorized recorders. SoundFlower is a sound source localization system for voice assistants. It can locate a user in 3D space through the wake-up command with a median error of 0.45 m.

In general, we explore the potential of diverse sensors to IoT services and build machine learning models to exploit the most information from sensor data. The applications we study are specifically about localization and security.

ACKNOWLEDGEMENTS

First and foremost, I would like to express my deepest gratitude to my PhD advisor, Dr. Zhichao Cao, for his advice, inspirations and encouragement. During the years at Edge Intelligence and Networking Group, Dr. Cao has been providing endless support to my research and career. Not only has he guided me into the giant picture of IoT research, but also he provides detailed instructions on experiment design, paper writing and research presentations. Dr. Cao is always patient and positive in spite of the ups and downs throughout my PhD life. I would also like to thank Dr. Li Xiao, Dr. Guan-Hua Tu and Dr. Mi Zhang for being on my thesis committee. The guidance from my thesis committee is invaluable to my academic career.

It is my pleasure to have spent two years under Dr. Yunhao Liu's supervision before he went to Tsinghua University. Dr. Liu pointed a direction for me when I first studied IoT. In the lab which is founded by Dr. Yunhao Liu and Dr. Zhichao Cao, I have met great labmates who have become my lifetime friends. Li Liu and I started research on IoT together and have been supporting each other all these years. Our friendship extends beyond the realms of research and career, reaching into the fabric of our lives. Maolin Gan's accompany is a source of joy and light to everyone in the lab. Gen Li and Yidong Ren helped me when I started teaching and always encouraged me whenever I doubted myself. Friendship from Yimeng Liu is also great support to me during my PhD life.

I also want to thank my mentor Dr. Xin Zhou. Despite all my inexperience during my first industrial internship, Dr. Zhou was supportive and inspired me to apply my PhD research to autonomous driving. After the internship is over, Dr. Zhou continuously shares her experience as a woman in tech and encourages me to pursue my career.

At last, I would like to thank my parents for their unconditional love and support.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

Internet of Things (IoT) refers to the network of computing devices which are embedded with sensors and interconnected through the Internet [1]. Distributed sensors gather data from physical objects, networking transfers diverse information from various locations to the computational system, which enables a comprehensive understanding of the environment as well as reactions of actuators. With cloud service deploying computation resources and machine learning analyzing sensor data, IoT could lead to complete automation of large infrastructures. Lots of novel concepts have been proposed and relevant systems are under construction, such as Smart Home [2], Smart Office [3] and Smart City [4]. In this dissertation, we are particularly interested in smart applications related to mobile devices. As mobile devices are the most common and widespread computational systems, exploring their existing sensors and building effective computational models is important to many IoT applications. With proper utilization, a mobile device can play an essential role to IoT products like Smart Home and Smart Office.

IoT architecture can be divided into four layers:

• **Sensing layer:** To initiate exchange of information between a physical object and a computational system, sensors play an essential role. Sensors monitor the physical conditions of the environment and collect data. Our IoT systems in this dissertation studies sensors like microphones, cameras and inertial sensors (IMU). As for physical signals, we can either use existing environmental signals, like sound source localization, or use modulated signals that interact with the surroundings, like gesture recognition through WiFi signals. For the latter one, the modulated signal is expected to be non-intrusive to the environment. For example, when acoustic signal is adopted to monitor infants, BreathJunior [5] especially selects chirps from 6 kHz to 21 kHz and modulate the chirps into pseudo white noise because long-term exposure to high-frequency chirps does harm to infants.

• **Network layer:** Network layer manages communication between devices. It can be wired or wireless communication. Wired communication transfers data through a wired medium like Eth-

ernet or USB. Our research focuses more on wireless signals. Nowadays wireless communications such as WiFi, LoRa, Bluetooth Low Energy (BLE), cellular networks (3G, 4G, 5G) and Radio-Frequency Identification (RFID) are very popular in IoT. In this dissertation, we especially studied acoustic signals to form a sensor network.

• **Data preprocessing layer:** After sensor data is collected and transferred to the computational system, the next step is to remove noises and extract relevant features. Sometimes we also need to remove redundant information to reduce the computational overload. In our application scenarios, hardware imperfection and environmental disturbances are major sources of noises. Signal processing and machine learning techniques are common methods to preprocess data in IoT.

• **Application layer:** After we have clean data, we can build computational models to achieve smart applications. Geometric models [6–9] and machine learning models [10–13] are two popular choices.

In spite of attractive capabilities, IoT system design faces manifold challenges. First, no sensor could provide perfect transfer function between the physical signal and the sensor data. For example, acoustic sensing suffers from information loss due to the discretization between analog signal and digital signal. WiFi sensing struggles with phase offset across radio chains, sampling frequency offset, symbol timing offset and carrier frequency offset. Image sensing is limited by resolution. Second, environmental noise is pervasive. The environmental noise might be more than random white noise. The sensor data contains traces of environmental information and interference, which is challenging to be completely removed. For example, one common environmental noise source for acoustic sensing is multipath effect. Due to this reason, the performance of an IoT system might degrade significantly after the system is deployed to a new environment. Third, the computational power of mobile devices is limited while most IoT applications require real-time responses. The training of larger neural networks like ChatGPT requires a cluster of GPUs[14], which is not affordable for mobile devices like laptops, smartphones and voice assistants. Last but not the least, the existence of sensors incurs privacy concern from users. On the one hand, we should choose proper sensors in different application scenarios. On the other hand, we need to

protect use privacy from malicious attackers by exploring the potential of sensors.

Over the years, many IoT systems have been designed to overcome the challenges and push the limit of IoT system design. In this dissertation, we introduce one system for indoor localization, one system for sound source localization and one system for acoustic security. For indoor localization, we propose EyeLoc [15] to enable self-localization in large shopping malls. For malls like Outlets, their area can reach as high as 70,000 m^2. Usually people depend on nearby kiosks to figure out directions, which is time-consuming and tiresome. With EyeLoc, people hold smartphones and turn a circle in place, then their location and heading direction will be shown on floor-plan images, which are widely offered by map providers like Google Maps and Gaode Maps. EyeLoc combines cameras and IMUs to explore the geometric relationship between angles and the relative location of the user with respect to three or more Point-of-Interests (POI). The 90-percentile errors of localization and heading direction are 5.97 m and 20° in 70,000 m^2 malls, which is sufficient to find a shop or an exit in a mall.

For sound source localization, we implemented SoundFlower for voice assistants like Amazon Echo. If voice assistants like Amazon Echo, Google Home or Apple HomePod can locate a person based on the speech he/she utters, they can better understand the context of commands and deliver more considerate tasks. Sound source localization is challenging because voice assistants are blind to the original speech. SoundFlower extracts Time Difference of Arrival (TDoA) information from phase data of cross spectrum. To cope with internal noise and environmental noise, we design a self-adjusting speech detection method to recognize speech-involved phase data. Robust regression is applied to extract TDoA from phase data against multipath effect. Our experiments prove that SoundFlower is robust and efficient.

For acoustic security, we designed a system to protect acoustic privacy from unauthorized recordings. Smart devices such as smartphones, smartwatches and digital wristbands are all designed with recording feature. In spite of conveniences, such function from portable and widespread devices expose us to the risk of malicious secret recording. We propose Patronus [16] to scramble unauthorized recordings with ultrasounds, while authorized devices can still recover informa-

tion from scrambled recordings with keys received through WiFi or bluetooth. The rationale of scrambling speech with ultrasounds is drawn from an observation discovered by BackDoor [17]. Although commercial mobile devices cannot sense ultrasounds due to their sampling rate limit, two ultrasounds with different frequency can incur a low-frequency signal within the microphone. When it comes to recovering information from the scrambled recording with the received key, we apply Normalized Least-Mean-Square (NLMS) adaptive filter. The process of NLMS filter is to simulate the received ultrasound and remove it from the recording.

For the structure of this dissertation, Chapter 2 introduces EyeLoc which is designed for indoor localization in large shopping malls. EyeLoc especially shows our efforts on achieving the trade-off among cost, computational power and real-time responses. Chapter 3 introduces SoundFlower, which is a sound source localization system for voice assistants. SoundFlower places emphasis on how to overcome pervasive environmental noise. Chapter 4 presents Patronus which is used to protect acoustic privacy. Patronus shows how to protect user privacy by making use of sensors and wireless signals. Finally, we conclude our work in Chapter 5.

CHAPTER 2

SMARTPHONE VISION ENABLED PLUG-N-PLAY INDOOR LOCALIZATION IN LARGE SHOPPING MALLS

Nowadays, the physical layout of many large shopping malls is becoming more and more complex [18]. As there are many location-based activities (e.g., shopping, eating, watching movies) in large shopping malls, indoor localization is becoming an important service for people. Although outdoor localization (i.e., GPS) has been put into practice for many years, there is still no practical deployed indoor localization systems.

Many indoor localization systems rely on pre-collected information (e.g., Wi-Fi signals [19] [20] [21] [22] [23], lamp positions [24] [25] [26], scene images [27] [28] [29] and magnetic fingerprints [30] [31]), called *site survey*, to construct a localizable map. In large shopping malls, the site survey usually incurs extensive bootstrap overhead which hinders corresponding approaches from widespread adoption. Even when the site survey is accomplished, the information needs to be timely updated and calibrated to ensure the accuracy. Moreover, some indoor localization systems [22] require custom hardware, which is not supported in commodity smartphones.

Our core question is *can we set up a plug-and-play indoor localization system in large shopping malls with commodity smartphones*? We notice a possible way by leveraging the widely available floor-plan images, which can be obtained from indoor map providers (e.g., Google Maps, Gaode Maps, Baidu Maps, etc.). Those floor-plan images contain positions of many shops, called Point of Interests (POI). POIs are used as visual hints when users try to manually localize themselves. This kind of nonautomatic self-localization usually requires users to have good geometric sense and the ability of space transformation. To reduce users' mental work and provide real-time localization service, we refine the question as *can users automatically obtain their positions on floor-plan images from their smartphones* just like traditional outdoor localization systems such as Google Maps and Baidu Maps? In this sense, it is possible to achieve a plug-and-play indoor localization system by bridging this gap.

In this chapter, we propose EyeLoc, a step towards plug-and-play indoor localization in large

shopping malls. The key idea of EyeLoc is to imitate human self-localization with smartphone vision. After obtaining a floor-plan image, EyeLoc uses scene text detection/recognition techniques to extract a set of POIs from the image. The recognized texts are used to identify different POIs and their corresponding text bounding boxes provide the approximate POI positions in the floor-plan coordinate system (called *floor-plan space*). Correspondingly in real space (called *vision space*), a user holds his/her smartphone and turns a 360° circle. The smartphone automatically shoots a series of images (called *view image*), which contain the surrounding POI signs. For those observed POIs, EyeLoc extracts their texts and geometric constraints in vision space, which are further used to match the user's position in floor-plan space.

Technically, EyeLoc develops several novel methods to address three challenges. First, there is a big difference between human vision system and smartphone vision system. Human vision system is a binocular system that supports estimating the direction and distance of an object. Most of the smartphones, however, only have one camera which is hard to achieve direction and distance measurements in a light-weight way with existing vision methods. We develop an accurate and ubiquitous monocular vision system which is available on most smartphones. We construct the constant geometric constraints of 3 observed POIs to enable position matching between floor-plan space and visual space. Second, to extract the directions of different POIs, text detection and recognition are necessary, but usually time-consuming. To reduce the processing time of POI extraction, an outlier image filtering method and a sparse image processing method are designed. Third, the measurement errors from motion sensors and floor-plan images may incur inaccurate position matching, for which we design an error-resilient method.

We implement EyeLoc on Android smartphones and evaluate its performance in an office environment, two large shopping malls (7,500m^2 and 10,000m^2) and a semi-outdoor large Outlets (70,000m^2). The evaluation results show that the 90-percentile errors of localization and heading direction can achieve 5.97 m and 20°. The contributions of this chapter are as follows.

• We propose EyeLoc, a smartphone vision enabled plug-and-play indoor localization in large shopping malls. No site survey nor periodical calibration of floor map is required.

• We develop a ubiquitous smartphone vision system and corresponding geometric localization model. To guarantee the localization accuracy and processing efficiency, we propose countermeasures to address several practical challenges.

• We implement EyeLoc on Android smartphones and evaluate its performance in an office environment and two large shopping malls. The evaluation results show that EyeLoc is effective in both localization accuracy and processing efficiency.

The rest of this chapter is organized as follows. Section 2.1 introduces the overview of EyeLoc. Section 2.2 illustrates the detailed design of EyeLoc. Section 2.3 and Section 2.4 show the details of EyeLoc implementation and evaluation respectively. Section 2.5 introduces the related work. Finally, we conclude our work in Section 2.6.

2.1 Overview

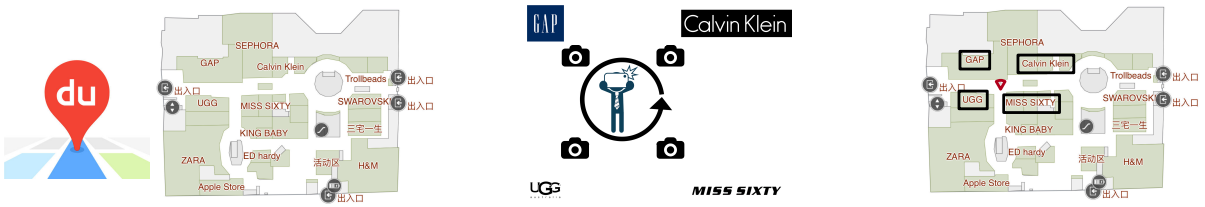
Plug-and-play outdoor localization has been successfully achieved on smartphones with the help of GPS. Referring to the criteria of GPS-based outdoor localization, EyeLoc has two goals:

• **Plug-and-play**. EyeLoc should not assume any extra bootstrap cost (e.g., site survey, system calibration) in large shopping malls. Moreover, EyeLoc should not require users to own any prior knowledge or follow complex smartphone operations.

• **Efficient and robust**. Facing computation-intensive image processing and various measurement errors from motion sensors and floor-plan images, EyeLoc should be able to accurately localize a user with short processing time.

To meet the first goal, EyeLoc is inspired by two observations. First, the indoor floor-plan images of shopping malls (e.g., shown in Figure 2.1(a)) can be easily fetched from indoor map providers through Android and iOS APIs. The other observation is that people are used to turning around to observe surrounding POIs and localize themselves. After fetching the floor-plan images, EyeLoc enables the self-localization of the smartphone through absorbing data from the on-board motion sensors and camera. No bootstrap cost or user training is involved.

Figure 2.1 is an example showing how EyeLoc works in a plug-and-play manner. Alice is lost in a large shopping mall and she wants to go to H&M. As Alice opens EyeLoc on her smartphone,



(a) Indoor Floor-plan Image (b) Circle Shoot (c) Location & Heading

Figure 2.1 Illustration of an example of the EyeLoc innovation.

the corresponding floor-plan image is automatically fetched. She holds the smartphone and turns a 360° circle, during which the camera and motion sensors keep working. This operation is called *circle shoot*. With input data from the camera and motion sensors, EyeLoc extracts geometric information of surrounding POIs (e.g., GAP, UGG, MISS SIXTY, Calvin Klein). Meanwhile, EyeLoc uses text detection and recognition techniques to find the POI positions on the floor-plan image. Finally, EyeLoc projects Alice's position and heading direction onto the floor-plan image as shown in Figure 2.1(c).

The involvement of image processing significantly increases the difficulty to meet the second goal. As Figure 2.1 illustrates, EyeLoc depends on text detection and recognition techniques to extract POI signs from view images. Text detection and recognition for color images have been widely studied in the past decade, especially with deep learning models like convolutional neural networks. A few open-source models (e.g., OpenCV [32], Tesseract [33]) are also available on smartphones. Smartphones can also utilize cloud service from companies like Google, Baidu, etc. However, none of the two approaches can achieve real-time execution due to computation overhead on images or extra network delay. This contradiction demands us to design an efficient method to extract enough geometric information without incurring long processing latency. We have two intuitions for the method design. First, since the extracted POIs with error geometric information is useless even harmful for localization, we should not deal with those view images of low quality. Second, we observe that the view images are usually redundant for extracting the geometric information of an observed POI. Hopefully, we can only select a subset of those view images which contain equivalent geometric information of the observed shops as the whole set

does for further processing.

On the other hand, various measurement errors are invertible and may lead to inaccurate localization. For example, as shown in Figure 2.1(c), the text bounding boxes of the four observed shops may be not exactly aligned to that of the corresponding shop signs that appeared in vision space. To mitigate potential errors and achieve robust localization, our observation is that the spatial POI distribution is usually dense in large shopping malls, which means multiple POIs are available. Hopefully, we can use the redundant information to refine the estimated user position.

In comparison with human binocular vision system, EyeLoc develops a monocular vision system, which is accurate and ubiquitous for smartphones. EyeLoc enables user position matching between vision space and floor-plan space with constant geometric constraints of observed POIs. The system architecture of EyeLoc is shown in Figure 2.2, including three parts as follows.

Raw Data Collection. The first part is to fetch floor-plan images from indoor map providers and collect raw information of view images from circle shoot. According to the coarse GPS localization, EyeLoc queries indoor map providers to obtain floor-plane images. During the circle shoot, Eyeloc uses the camera and motion sensors (e.g., compass, gyroscope, accelerometer) to continuously capture view images and corresponding motion attributes (e.g., camera facing direction, angle velocity). Section 2.2.3 shows the design details.

POI Extraction. Taking the information of view images and floor-plan images as input, the second part extracts geometric information of observed POIs in both vision space and floor-plan space. Because text detection and recognition are time-consuming, we need to extract enough geometric information while keeping the number of processed images as small as possible. EyeLoc filters out some view images which are blurred or have error motion attributes. Then EyeLoc develops a sparse image processing method to extract geometric information of all observed POIs from the rest of the images and keeps the number of processed images small. On the other hand, after extracting all POIs on the floor-plan image, EyeLoc obtains the positions of the observed POIs in floor-plan space by matching their names. The detailed design is illustrated in Section 2.2.4.

Position Matching. With the geometric information of the observed POIs, EyeLoc now

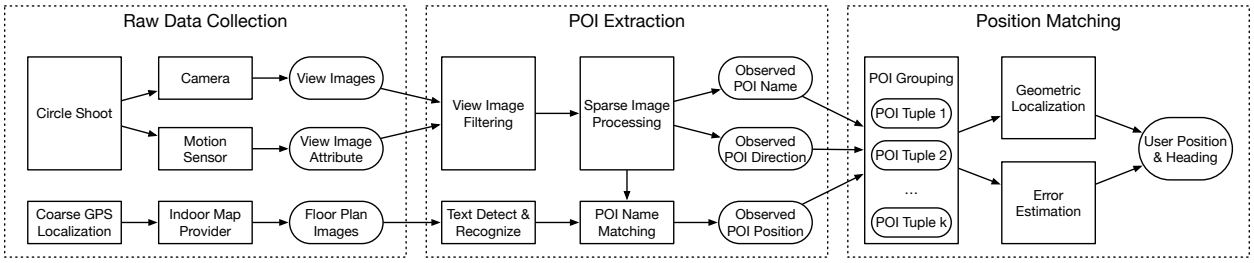


Figure 2.2 Illustration of the system architecture of EyeLoc.

projects the user's position and heading direction onto floor-plan space. The redundancy of the observed POIs is explored to mitigate unavoidable errors of geometric information in vision space and positions in floor-plan space. The observed POIs are grouped into tuples. Each POI tuple can be used to calculate the user's position and heading direction with geometric constraints. The localization errors of different tuples are diverse with the same measurement errors. EyeLoc combines several inferred positions and the corresponding errors to vote the final user's position and heading direction. The detailed design is shown in Section 2.2.5.

2.2 Design

To relieve humans of self-localization, EyeLoc achieves plug-and-play localization in large shopping malls. Due to the fundamental difference between the vision principle of humans and smartphones, we first establish the smartphone vision system and illustrate the geometric localization model. To further put EyeLoc into practice, we show the detailed design of three function components (shown in Figure 2.2) step by step.

2.2.1 Smartphone Vision System

We intend to define a ubiquitous smartphone vision system to fetch the geometric relationship between a smartphone and an observed POI. Human eyes form a binocular vision system, which enables us to estimate our distance and direction to an observed POI. Smartphone vision differs significantly from human vision. First, not all smartphones have been equipped with dual or triple cameras. It is hard to extract the distance and direction of an observed object from a monocular image except there is a preconfigured Structure from Motion (SfM) based model or learning based model. However, both of these two approaches require a large set of images for model training,

which incurs the heavy burden of site survey. Second, humans have practiced a lot since childhood, so camera calibration [34] is a must to achieve accurate estimation. Since the parameters of camera calibration are not explicitly known for those smartphones, the complicated operation induces unacceptable difficulty to bootstrap this ability for common users. In EyeLoc, the question is *can we estimate distance and direction as the geometric descriptor of an observed object through the monocular view images of circle shoot?*

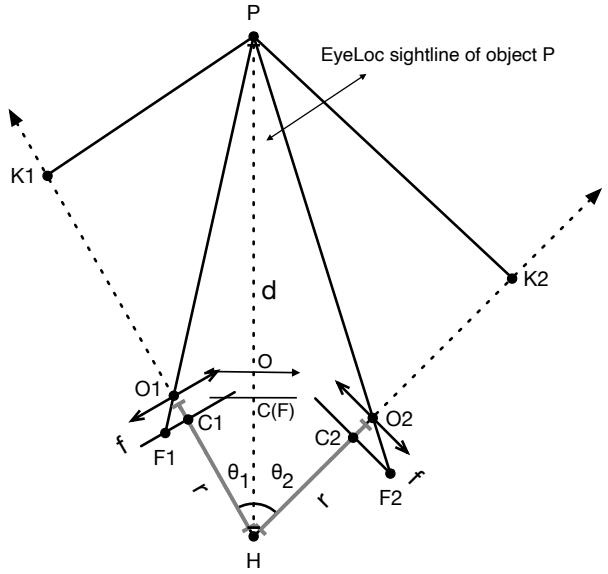


Figure 2.3 Direction and distance measurement with circle shoot.

It is confirmative that the direction information of a POI can be constructed with the monocular view images of circle shoot. We define the *EyeLoc sightline* of an object as the virtual line between the object and the user. For example, as shown in Figure 2.3, given a POI P and a user H, the EyeLoc sightline is HP. O is the optical center of the camera lens and C is the center of the image plane. H, C and O are approximately kept on the same line all the time during circle shoot. When the user is facing P, HP coincides with CO and its direction can be measured by smartphone motion sensors [35]. During circle shoot, however, the user's facing direction is continuously changing. For example, C_1O_1 and C_2O_2 are not aligned with HP. The key observation is that when CO and HP are aligned, F that indicates the position of P on a view image will coincide with C. Otherwise, it will appear at the side of C as F_1 and F_2 show. As shown in Figure 2.4(b), (c) and

(d), when a user turns in clockwise during circle shoot (e.g., Figure 2.4(a)), for shop "MOUSSY", its text bounding box will appear from left to right in the view images. EyeLoc finds the view image (e.g., Figure 2.4(c)) of which the text bounding box is at the center, then the direction of EyeLoc sightline can be estimated by motion sensor.

Figure 2.4 An example of the sightline change during circle shoot in physical space.

To enable distance estimation with monocular vision, it is possible to exploit the camera motion of circle shoot to imitate a binocular vision system. As shown in Figure 2.3, the distance between the POI P and the user H is indicated as d. The distance between H and the optical center of smartphone camera lens O_1 is r. The focal length of the smartphone camera lens f is unknown for most smartphones. θ_1 indicates the intersection angle between line HO_1 and EyeLoc sightline HP. Since $\triangle F_1O_1C_1$ is similar to $\triangle PO_1K_1$, we have the following equation:

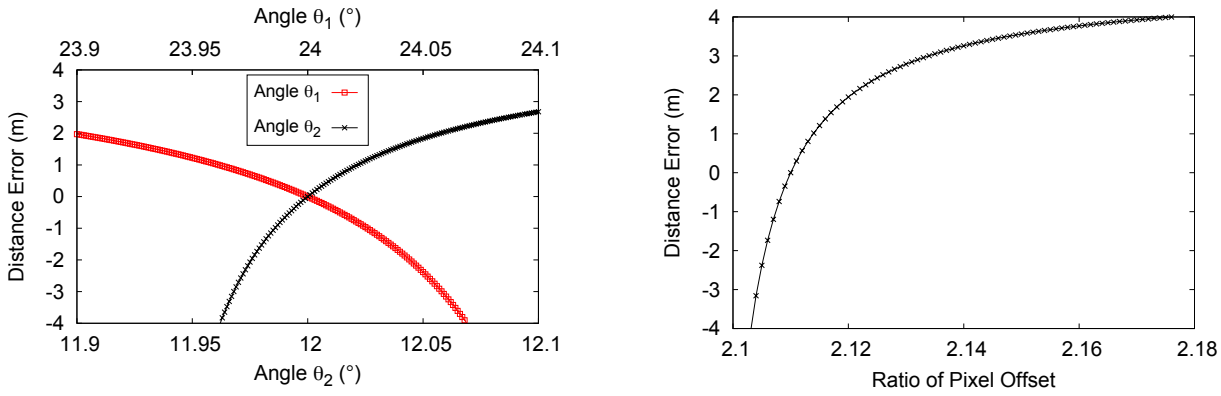
$$\frac{F_1C_1}{d\sin\theta_1} = \frac{f}{d\cos\theta_1 - r} \tag{2.1}$$

where F_1C_1 indicates the pixel offset between F_1 and $C1$. Combining the same equation under another angle θ_2 ($\theta_1 \neq \theta_2$), we can derive d as follow:

$$d = r\frac{\sin\theta_1 - k\sin\theta_2}{\cos\theta_2\sin\theta_1 - k\cos\theta_1\sin\theta_2} \tag{2.2}$$

where k equals the ratio between F_1C_1 and F_2C_2. If θ_1, θ_2, k and r are known, the distance d can be calculated. As the direction of HP, HO_1 and HO_2 can be obtained by motion sensors, θ_1 and

θ_2 can be calculated. For a shop, we recognize the center of its text bounding box as F_1 and F_2 on a view image so that F_1C_1, F_2C_2 and k can be calculated. r can be roughly estimated according to human arm length. In this way, EyeLoc can estimate the distance of a POI without any prior knowledge of camera parameters in large shopping malls.



(a) Influence of θ_1 and θ_2

(b) Influence of k

Figure 2.5 Illustration of distance error in terms of the error of θ_2 and k. (a) and (b) show the distance error under different θ_2 and k when other parameters are fixed.

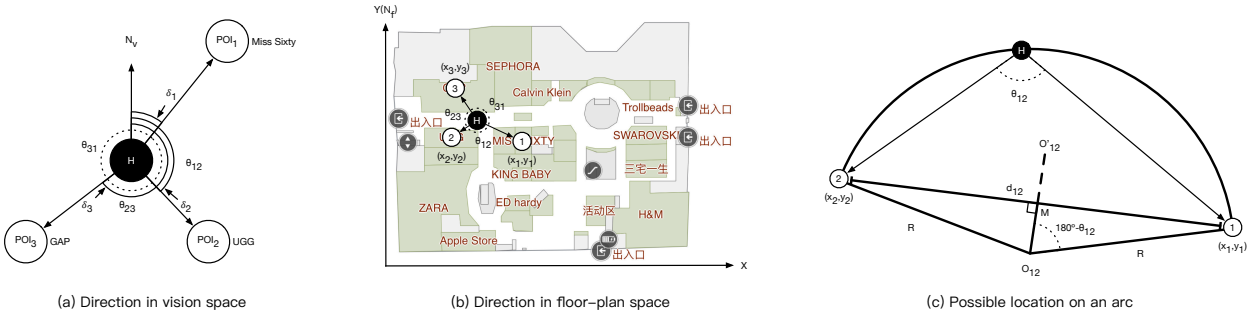
Since the error of θ and k estimation is inevitable, we further conduct the error analysis. We assume r is 0.5m. Given θ_1, θ_2 and k are 24°, 12° and 2.11, d will be 5.48m. We change one parameter (e.g., θ_1, θ_2, k) to calculate the distance error when other parameters are fixed. The results are shown in Figure 2.5a and Figure 2.5b. Surprisingly, given the distance as 5.48m, the distance error is huge when θ_1, θ_2 and k have a small bias. The distance error is getting to 1.97m, when θ_1 decreases from 24° to 23.9°. Similarly, when θ_2 increases from 12° to 12.1°, the distance error increases from 0m to 2.68m. 0.1° error is common for the facing direction measurement with motion sensors. The same trend happens on k. When k increases from 2.11 to 2.16, the distance error increases from 0m to 3.77m. Due to the limitation of image resolution, given that F_1C_1 is as large as 1000 pixels, 0.05 error of k means about no more than 50 pixels error of F_2C_2 which is hard to achieve due to the relatively large estimation error of text bounding box. When θ_1 increases or θ_2 and k decreases a little, the situation is getting even worse. Hence, due to the limitation of motion sensor precision and image resolution, the potential huge error makes the

distance estimation unpractical currently.

Overall, in our smartphone vision system, for a POI, we only use the direction of its EyeLoc sightline as the geometric descriptor to develop an error-controllable localization model (Section 2.2.2). To accurately and efficiently trace the sightline of a smartphone, we further develop several countermeasures in Section 2.2.4 and Section 2.2.5. We leave the accurate distance measurement using a monocular vision system as our future work.

2.2.2 Geometric Localization Model

After we obtain the direction of the EyeLoc sightline of several POIs, the next question is *how to construct constant geometric constraints, then figure out the user's position and heading direction on the floor-plan image.*



(a) Direction in vision space (b) Direction in floor–plan space (c) Possible location on an arc

Figure 2.6 Illustration of the model used to localize a user's position on floor-plan image with 3 observed POIs. (a) and (b) exhibit a constant geometric constraint in both vision space and floor-plan space. (c) shows the model to calculate the user's location with the extracted geometric information.

Let us show the constant geometric constraints through an example. As shown in Figure 2.6(a), H is a user's position. The user observes 3 POIs (e.g., POI_1 Miss Sixty, POI_2 UGG and POI_3 GAP) in their appearance order and N_v indicates the north direction in vision space. As shown in Figure 2.6(b), H also indicates the user's position. 1, 2 and 3 represent the corresponding center of text bounding boxes of 3 observed POIs. Given the coordinate system X-Y of the floor-plan image, (x_1, y_1), (x_2, y_2) and (x_3, y_3) are the corresponding coordinate of 1, 2 and 3. N_f is the north direction in floor-plan space which aligns with Y axis. In vision space, the directions of EyeLoc sightline δ_1, δ_2 and δ_3 can be estimated. However, since N_v and N_f may not be aligned with each

Algorithm 2.1 Geometric Constraints Extraction Algorithm

Input: 3 POIs sorted as their appearance order; the directions of the corresponding EyeLoc sightline δ_1, δ_2, δ_3 in vision space.

Output: d_{12}, d_{23}, d_{31}, θ_{12}, θ_{23} and θ_{31}.

1: vector of POI_1 EyeLoc sightline $v_1 = (\sin\delta_1, \cos\delta_1)$.
2: vector of POI_2 EyeLoc sightline $v_2 = (\sin\delta_2, \cos\delta_2)$.
3: vector of POI_3 EyeLoc sightline $v_3 = (\sin\delta_3, \cos\delta_3)$.
4: $d_{12} = v_1 \times v_2$, $d_{23} = v_2 \times v_3$ and $d_{31} = v_3 \times v_1$.
5: $\theta_{12} = (\delta_2 - \delta_1) \mod 360^\circ$; $\theta_{23} = (\delta_3 - \delta_2) \mod 360^\circ$; $\theta_{31} = (\delta_1 - \delta_3) \mod 360^\circ$

other, we cannot directly determine the coordinate of H with these directions in floor-plan space.

We have two constant geometric constraints in both vision and floor-plan spaces. The first is the rotation direction of circle shoot is constant. The 3 POIs will appear in the same order (e.g., $POI_1 \rightarrow POI_2 \rightarrow POI_3$ and $1 \rightarrow 2 \rightarrow 3$) along the rotation direction. We use d_{12}, d_{23} and d_{31} to indicate the rotation direction between each pair of adjacent POIs. The other is that, according to the similar triangles, $\angle POI_1HPOI_2$, $\angle POI_2HPOI_3$ and $\angle POI_3HPOI_1$ equal to $\angle 1H2$, $\angle 1H2$ and $\angle 1H2$. The 3 intersection angles are indicated as θ_{12}, θ_{23} and θ_{31}. The rotation direction and the intersection angles between any two POIs sever the constant geometric constraints for both vision space and floor-plan space. Algorithm 2.1 exhibits the details to determine the d_{12}, d_{23}, d_{931}, θ_{12}, θ_{23} and θ_{31} given 3 POIs and their corresponding directions of EyeLoc sightline.

Given POI coordinates ((x_1,y_1), (x_2,y_2), (x_3,y_3)), rotation direction (d_{12}, d_{23}, d_{31}) and intersection angle (θ_{12}, θ_{23}, θ_{31}), we need a method to calculate the coordinate (x_H, y_H) of the user's position H in floor-plan space. As shown in Figure 2.6(c), given the coordinates of two POIs (e.g., 1, 2), the rotation direction d_{12} and the intersection angle θ_{12}, if θ_{12} is 180°, H is on the segment between 1 and 2. Otherwise, the possible position of H is on an arc that takes the segment $\vec{12}$ as the chord and θ_{12} as the inscribed angle. The pixel distance between 1 and 2 is l_{12} which equals $\sqrt{(x_1-x_2)^2+(y_1-y_2)^2}$. M is the middle point of the chord $\vec{12}$ and it's coordinate (x_M, y_M) equals ($\frac{x_1+x_2}{2}$, $\frac{y_1+y_2}{2}$). O_{12} is the center of the circle and R is the length of its radius. We use (x_o, y_o) to indicate the coordinate of O_{12}. If θ_{12} is 90°, O_{12} and M have the same coordinate. Otherwise, since 1 and 2 are on the circle, the chord $\vec{12}$ is perpendicular to MO_{12} and we have the following

equation:

$$\frac{x_1 - x_2}{y_1 - y_2} = -\frac{y_o - y_M}{x_o - x_M} = k \tag{2.3}$$

where k is the slope of the chord $\vec{12}$. Moreover, the central angle $\angle 1O_{12}2$ is twice the corresponding inscribed angle which equals $180^\circ - \theta_{12}$ and $\angle 1O_{12}M$ is half the central angle $\angle 1O_{12}2$. Hence, $\angle 1O_{12}M = 180^\circ - \theta_{12}$ and $R = \frac{d_{12}}{2\sin\theta_{12}}$. For the length of $O_{12}M$, we have the following equation:

$$\sqrt{(x_o - x_M)^2 + (y_o - y_M)^2} = -\frac{l_{12}}{2\tan\theta_{12}} \tag{2.4}$$

Combining Equation 2.3 and Equation 2.4, we can obtain (x_o, y_o) as follows:

$$x_o = x_M \pm \frac{l_{12}}{2\tan\theta_{12}\sqrt{1+k^2}}; y_o = y_M \mp \frac{kl_{12}}{2\tan\theta_{12}\sqrt{1+k^2}} \tag{2.5}$$

Besides O_{12}, we obtain another false center of circle O'_{12} which is symmetric with O_{12} by taking $\vec{12}$ as a mirror. To filter out the outlier O'_{12}, we further exploit the information of the rotation direction d_{12} and acute/obtuse angle θ_{12}. If θ_{12} is an acute angle, O_{12} is on the same side with H regarding segment $\vec{12}$. Otherwise, O_{12} is on the opposite side with H. In this way, we can identify the unique coordinate of O_{12}. Algorithm 2.2 summarizes the detailed calculation of O_{12} and R. Now, we know H is on an arc determined by O_{12} and R. Similarly, we can calculate another arc where H is on with POI_2, POI_3 and θ_{23}. We further calculate the intersections of these two arcs. One intersection is POI_2, the other is the position of H.

In the angel-based geometric localization model, localization bias may be incurred by an unexpected situation: when H, POI_1, POI_2 and POI_3 are on the same circle, we cannot localize H through the geometric constraints of the 3 POIs. This situation rarely happens in practice as shown in Section 2.4. Moreover, we may be able to observe more than 3 POIs at large shopping malls. If the situation has happened, EyeLoc will popup a message to remind the user that he/she needs to walk several steps and relocalize himself/herself.

When the coordinate of H is known, we can calculate the directions of $H\text{POI}_1$, $H\text{POI}_2$ and $H\text{POI}_3$ in floor-plan space. Then, with δ_1, δ_2 and δ_3, we can calculate the angle offset Δ_N between the vision north N_v and floor-plan north N_f. Given any user's heading direction (e.g., camera facing

Algorithm 2.2 Arc Calculation Algorithm

Input: 2 POIs, POI_1 and POI_2, sorted as their appearance order; the rotation direction d_{12}; the angle θ_{12} between the directions of the corresponding EyeLoc sightline in vision space; the corresponding coordinates (x_1, y_1) and (x_2, y_2) in floor-plan space.

Output: The coordinate of circle center (x_o, y_o); the length of circle radius R.

pixel distance and slope of the chord $\vec{12}$ as $d_{12} = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$ and $k = \frac{x_1 - x_2}{y_1 - y_2}$.
if θ_{12} equals to 180° **then**
    H is on the segment between POI_1 and POI_2. (x_o, y_o) and R are set as NULL.
else if $\theta_{12} > 180^\circ$ **then**
    $\theta_{12} = 360^\circ - \theta_{12}$.
else if θ_{12} equals to 90°. **then**
    $(x_o, y_o) = (\frac{x_1 + x_2}{2}, \frac{y_1 + y_2}{2})$;$R = d_{12}/2$
else
    $R = \frac{d_{12}}{2 \sin \theta_{12}}$; two possible coordinates (x_{o1}, y_{o1}) and (x_{o2}, y_{o2}) of circle center are calculated by Equation 2.5.
    set (x_o, y_o) as (x_{o1}, y_{o1})
    calculate the rotation direction $d_{o12} = \text{vector}(x_o - x_1, y_o - y_1) \times \text{vector}(x_o - x_2, y_o - y_2)$
    if $d_{12} \cdot d_{o12} > 0 \oplus \theta_{12} < 90^\circ$. **then**
        set (x_o, y_o) as (x_{o2}, y_{o2}).
    end if
end if

direction) in vision space, we can infer his/her heading direction in floor-plan space. Overall, EyeLoc can calculate a user's position and heading direction by observing no less than 3 POIs. More POIs can further improve the accuracy as introduced in Section 2.2.5.

2.2.3 Raw Data Collection

EyeLoc takes three data sources as input: view images captured by the camera, view image attributes measured by motion sensors and the floor-plan image fetched from indoor map providers. The camera and motion sensors work during the circle shoot, while the user is moving and the smartphone may be shaking slightly. To control the consequent measurement errors, as well as the processing latency and overhead, we conduct the raw data collection as follows:

2.2.3.1 View Images

Two system parameters are crucial to image shooting. One is the image resolution I_r. The higher its value is, the text of more POIs can be accurately detected and recognized. However, the processing time also increases when I_r becomes high. Empirically, EyeLoc fixes I_r as 1536p

during the circle shoot. The other parameter is the shooting frequency f_s, which means the interval between two adjacent view images is $\frac{1}{f_s}$. A high f_s ensures all surrounding POIs can be recorded when the rotation speed of a user is fast. Redundant view images also have a negative influence on processing time. EyeLoc selects a relatively high f_s to guarantee the abundance of raw data. Later in Section 2.2.4 a smaller resolution I_p will be introduced for further image filtering and complement I_r and f_s.

2.2.3.2 Motions Sensor Readings

In most cases, when a user operates circle shoot, the facing direction of the user and his/her smartphone camera is the same as illustrated in Figure 2.7. We define the *EyeLoc sightline* of an object as the virtual line between the object and the user as shown in Figure 2.7. The direction of a sightline δ is the angle between earth north and the projected direction Z' of the smartphone Z axis, which is measured through the estimation of the camera facing direction.

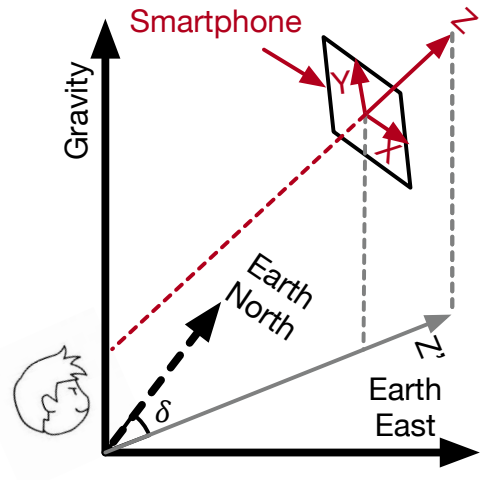


Figure 2.7 Illustration of the camera facing direction δ measurement.

We use the motion sensors (e.g., accelerometer, gyroscope and compass) to capture the camera facing direction. EyeLoc continuously samples the readings of the motion sensors. We collect the direction of gravity via the acceleration sensors along 3 smartphone axes (e.g., X , Y and Z), and determine the direction of north with compass sensors to calculate the direction of Z in the earth coordinate system. As a result, the direction of Z' and δ are calculated correspondingly. To remove potential magnetic interference and bursty noise, EyeLoc adopts several methods [35] [36]

to calibrate the camera facing the direction of each view image.

2.2.3.3 Floor-plan Image

Given the coarse GPS readings in a shopping mall, EyeLoc can fetch the floor-plan images of all floors in the shopping mall through APIs of indoor map providers. Each floor-plan image contains the skeleton and name of all POIs on that floor.

Overall, the raw data collection module outputs a series of view images, corresponding EyeLoc sightline directions and floor-plan images. However, the redundancy and measurement errors existing in raw data will incur computation inefficiency and localization error. Next, we introduce the methods to improve the efficiency and robustness.

2.2.4 POI Extraction

For the floor-plan image, we use text detection/recognition techniques to collect POI signs and record their coordinates on the floor-plan image as shown in Figure 2.1(c). As for view images, our goal is to detect all available POI signs and determine corresponding sightlines. The angle formed by two EyeLoc sightlines is critical for position matching in Section 2.2.5. The key issue is how to achieve real-time performance on smartphones.

2.2.4.1 View Image Outlier Filtering

During raw data collection, we obtain abundant view images and motion sensor readings. As the user is moving while the camera and motion sensors are working, some data contain errors that can significantly influence the localization result. Filtering out those data can also save the processing time.

If a view image is blurred, we cannot detect any text at all. We treat blurred view images as outliers that should be filtered out. EyeLoc adopts Laplacian-based operator [37], which is a widely used function for focus measure, to define the degree of image blur. We randomly selected 1692 view images from the whole data set shot in two large shopping malls (Section 2.4). We guarantee there is at least one POI sign in each of these view images. In Figure 2.8, the black curve shows the Laplacian variances of these images are distributed from 0 to 400. The larger the variance is, the

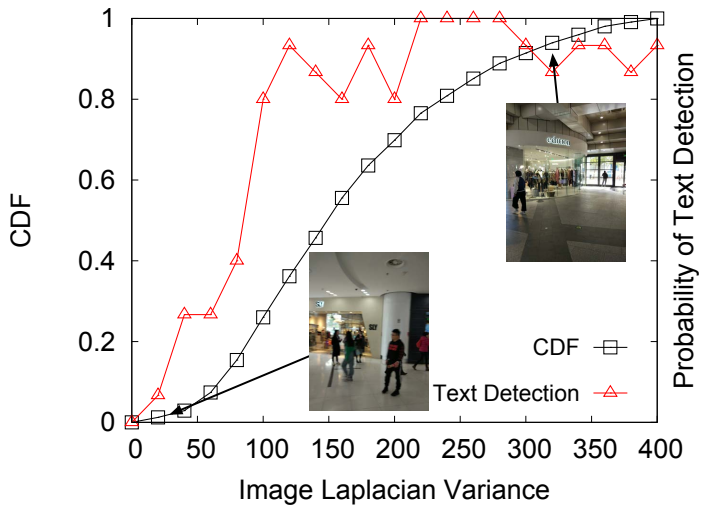


Figure 2.8 The influence of image blur on the accuracy of text detection.

less the image blur is, as shown in the comparison between the example view images with variance in the range [0,20] and [300,320]. In a view image, if the length of any recognized text string is more than 2, it is text-detectable. We further select 15 view images from each level of image blur to evaluate the probability of text detection under different levels of image blur. The red curve shows that the probability of text detection is higher than 80% when the Laplacian variance is larger than 80. Hopefully, we should filter out those view images whose Laplacian variance is less than 80 because it is hard to detect any text clues from it. Thus, we define a threshold Δ_{Lap} (e.g., approximate 80) to determine whether a view image is blurred or not.

Moreover, the smartphone vibration around Y axis and Z axis can influence the position of a POI on view images. As a result, the estimation error of EyeLoc sightline may increase. The gyroscope outputs the angular velocity around 3 axes as ω_x, ω_y and ω_z, then the total angular velocity is $\omega = \sqrt{\omega_x^2 + \omega_y^2 + \omega_z^2}$. On the other hand, we can also calculate the angular velocity ω' given the Z' direction of two adjacent view images and the corresponding time interval. We conduct a circle shoot by setting f_s as 2Hz. During circle shoot, we manually vibrate the smartphone as a common user does when the picture ID is from 15 to 18 and from 26 to 30. As shown in Figure 2.9, the angle velocity difference between ω and ω' is close to zero as usual. However, smartphone vibration will obviously increase the difference. This observation indicates different

motion sensors have different sensitivity for the vibration. Hence, EyeLoc sets a threshold Δ_ω and filters those view images when the angular velocity difference is larger than Δ_ω.

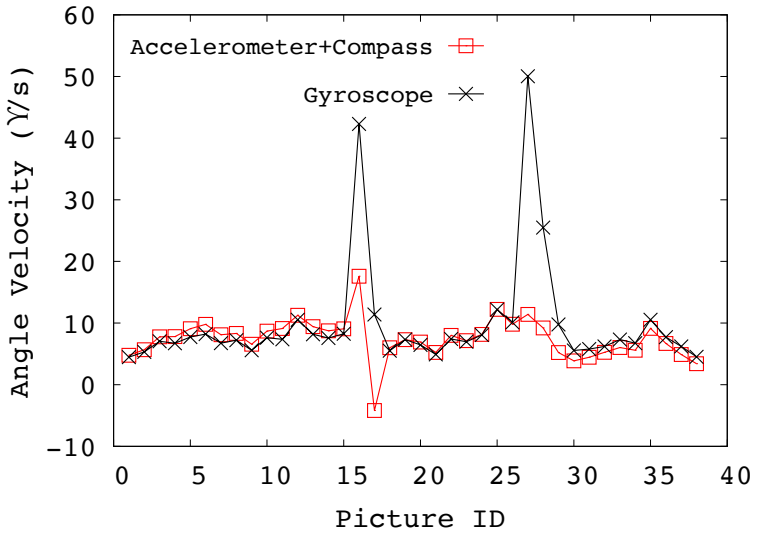


Figure 2.9 The angle velocity measured by different combination of motion sensors.

2.2.4.2 Text Filtering and Matching

In large shopping malls, text may appear or extract anywhere. It is possible to detect multiple text strings from a view image. Figure 2.10 shows the case in the office environment. The top figures are RGB scene pictures and the bottom figures exhibit the red text bounding boxes on corresponding binary images. We can see that besides the desired text bounding box of "STARBUCK", many redundant ones are also extracted from the textures of curtains, tables and switches. Moreover, in Figure 2.10(a), only part of "STARBUCK" are recognized. Situations in shopping malls can be more complex since the name of a POI may appear at multiple places.

EyeLoc filters out the irrelevant and duplicate text bounding boxes through the following steps. First, given the minimum and maximum length of POI names extracted from floor-plan images, EyeLoc filters out the illegal text strings. Second, we group the rest of the text strings. Two text strings belong to the same group when the difference between them is smaller than a threshold Δ_t. The difference between the two text strings is defined as the ratio between their Levenshtein Distance [38] and the maximum string length. Given the list of POI names extracted from floor-plan images, EyeLoc further removes those invalid groups whose text strings are not on the list

Figure 2.10 Landmark identification and text bounding box.

(i.e., the similarity is smaller than Δ_t in comparison with any POI name). Finally, in each valid group, EyeLoc combines the coordinates of all text bounding boxes to calculate the average value as the unique text bounding box position of the observed POI on the view image. In this way, EyeLoc identifies available POIs and corresponding positions of text bounding boxes on a view image.

2.2.4.3 Sparse Image Processing

After filtering the outliers of view images, for all observed POI, we need to exactly find the view images (e.g, Figure 2.4(c)) where the corresponding text bounding boxes appear in the middle. The intuitive approach is to process all view images, but this will incur heavy networking and computation burden as the sampling frequency f_s is set high. Even worse, the desired view image may not be captured or blurred. Instead of processing every view image to extract the geometric information of all potential POIs, EyeLoc develops a sparse image processing approach to achieve the same goal. The key idea is after the position of a text bounding box is known from a view image (e.g., Figure 2.4(b)), we can enable EyeLoc sightline estimation of a POI with one more view image which contains the same POI (e.g., Figure 2.4(d)) by feature point matching. As shown in Figure 2.11, given two view images I_1 and I_2, the text bounding box of I_1 is extracted and it is d_1 pixel from the middle line. Then, in I_2, we use ORB algorithm to extract the same feature points which fall into the text bounding box of I_1. Given a feature point, its coordinates on I_1 and

I_2 are (x_1, y_1) and (x_2, y_2). The pixel distance of the feature point is $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$. The average pixel distance of all feature points is indicated as l_f. Due to the approximate constant ratio between pixel distance and central angle, given their direction as δ_1 and δ_2, we can calculate the direction δ of the POI EyeLoc sightline as following:

$$\delta = \delta_1 + \frac{d_1}{l_f}(\delta_2 - \delta_1) \tag{2.6}$$

When we recognize the text bounding box of a POI from a view image, we use its adjacent images which probably contain the same POI to calculate the direction of POI EyeLoc sightline with Equation 2.6.

Figure 2.11 Feature point matching in two different view images of the same POI.

To extract the geometric information of all observed POIs, the problem becomes to quickly target a view image for each observed POI. Given n view images $\{I_1, I_2, ..., I_n\}$, we set a step length Δ_s and view images $\{I_{\Delta_s}, I_{2\Delta_s}, ..., I_{k\Delta_s}\}$ ($k = \lfloor \frac{n}{\Delta_s} \rfloor$) are selected for processing. Hopefully, if the minimum number of view images of a POI is larger than Δ_s, EyeLoc cannot miss the view image of any observed POI. However, due to the possible failure of text recognition, we may miss some POIs so that no enough POIs are extracted. In this case, EyeLoc will exponentially reduce Δ_s and reprocess the new view images until at least 3 POIs are extracted or all view images are processed. Overall, we can fetch enough available POIs and corresponding directions of EyeLoc

sightline as soon as possible for later location matching. Next, we remove the potential estimation errors to achieve accurate location matching.

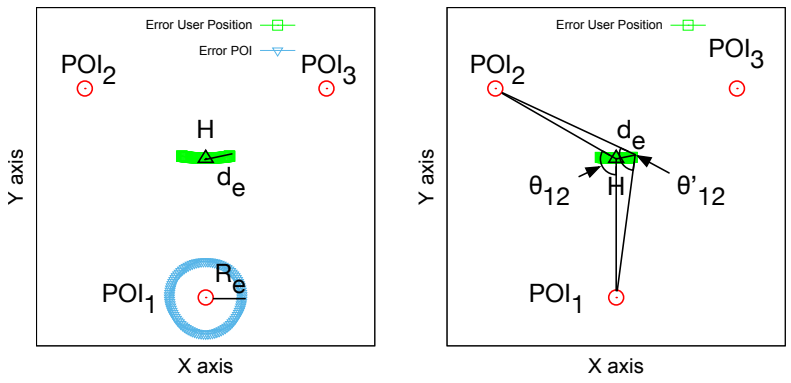
2.2.5 Position Matching

According to the localization model in Section 2.2.2, the user's position can be localized with three observed POIs, called *localization tuple*. Figure 2.12 shows the measured POI coordinates of a localization tuple (e.g., POI_1, POI_2, POI_3) and the calculated user's position H. The corresponding measured intersection angle θ_{12}, θ_{23} and θ_{31} are 120°. The POI text bounding boxes in the floor-plan image may not exactly align with that in physical space. Due to the POI coordinate errors, we assume the true POI positions may appear on a circle around it and the radius is R_e. As shown in Figure 2.12(a), when moving the POI_1 around a circle with a radius of 3 pixels and keeping the positions of other POIs fixed, the calculated positions are shown as the green marks in the figure. Moreover, due to the possible errors from motion sensor and image processing, θ_{12} may be inaccurately measured in comparison with the true intersection angle θ'_{12} as shown in Figure 2.12(b). The same situation may happen for θ_{23} and θ_{31}. We assume the error of θ_{12}, θ_{23} and θ_{31} is in the range of $[-\Delta_\theta, \Delta_\theta]$. For θ_{12}, Figure 2.12(b) shows the possible true positions shown as the green marks when Δ_θ is 10 °. Regarding the errors of POI_1 and θ_{12}, the maximum localization errors are indicated as d_e. The ratio between d_e and R_e or Δ_θ is further defined as the error sensitivity of a POI or an intersection angle. Given a localization tuple u, we define its *localization error sensitivity* $les(u)$ as the sum of the error sensitivity of all three POIs and three intersection angles.

When k ($k \geq 3$) POIs are extracted, we have total $m = k(k-1)(k-2)/6$ localization tuples indicated as $\{u_1, u_2, ..., u_m\}$. For the i^{th} tuple u_i, its localization result and error sensitivity are indicated as h_i and $les(u_i)$. The larger the $les(u_i)$ is, the more accurate the h_i is. Hence, EyeLoc sets the weight w_i of localization result h_i as $1/les(u_i)$. Then, the final match location h is calculated as follows:

$$h = \frac{\sum_{i=1}^{m} w_i h_i}{\sum_{i=1}^{m} w_i} \tag{2.7}$$

With h and k extracted POIs, EyeLoc can calculate the user's heading direction according to the



(a) POI Error (b) Direction Error

Figure 2.12 The localization sensitivity regarding (a) POI error and (b) direction error.

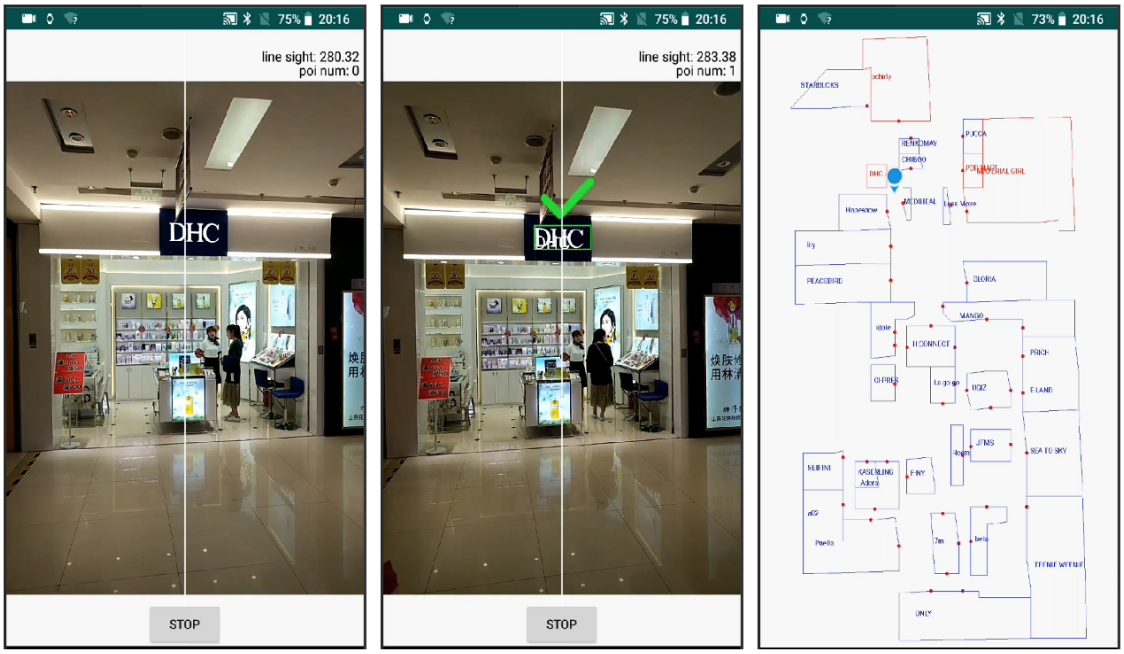
method in Section 2.2.2.

2.3 Implementation

We implement EyeLoc as a mobile application in Android 7.0. Figure 2.13 demonstrates the user interface (UI) of EyeLoc application when we conduct experiments in a shopping mall. As shown in Figure 2.13(a), after a user opens EyeLoc application, the view captured by the camera appears on the smartphone screen. The view keeps refreshing with the circle shoot. A vertical white line appears in the center of the screen as the reference. Once a POI appears during the circle shoot, EyeLoc extracts its text string from the view image. If the text bounding boxes is aligned with the sightline of the smartphone camera, a green checkmark appears on the screen like Figure 2.13(b). In this way, users can simply record POIs as many as possible. After the user finishes the circle shoot, EyeLoc exhibits the user's position and heading direction on the floor-plan image as shown in Figure 2.13(c). We discuss several system details and settings as follows.

2.3.1 Scene Text Detection and Recognition

Scene text detection and recognition techniques serve as a fundamental role in EyeLoc. We compare several existing techniques on Android smartphones in terms of recognition accuracy and processing time. Here, we adopt the same method in Section 2.2.4.2 to judge the similarity between two text strings. Δ_t is set as 50%. We randomly select 100 images shot by a smartphone in two large shopping malls. Some of them are shot at daytime and the others are shot at night.



(a) An view image during the circle shoot

(b) POI extraction

(c) Display the position and the heading direction on the floor-plan image

Figure 2.13 The UI of EyeLoc when running in a large shopping mall.

Each image contains one shop sign which is manually labeled as the ground truth.

2.3.1.1 Local processing v.s. Cloud processing

According to different processing platforms, we can either perform the text detection and recognition processes locally on the smartphone, or remotely on cloud. The approaches of local processing include OpenCV [32] and Tesseract [33]. We choose Baidu Cloud as the platform for the typical cloud processing. LTE network, which is available in most shopping malls nowadays, is adopted to connect the smartphone with the cloud server.

Given the dataset of 100 images, the recognition accuracy and processing time are shown in Figure 2.14. We can see that the text recognition accuracy of Baidu Cloud is 66%, which is much higher than 12% of Tesseract and 2% of OpenCV. The text recognition accuracy of Tesseract and OpenCV is surprisingly low since it is challenging for the text classifiers and extreme region extraction to adapt to the complex lighting condition and text format of POI signs. The average processing time of Baidu Cloud is 2s which is a little higher than that of OpenCV, but much smaller than Tesseract. Due to the superior recognition accuracy and low processing time, we choose cloud

processing instead of local processing.

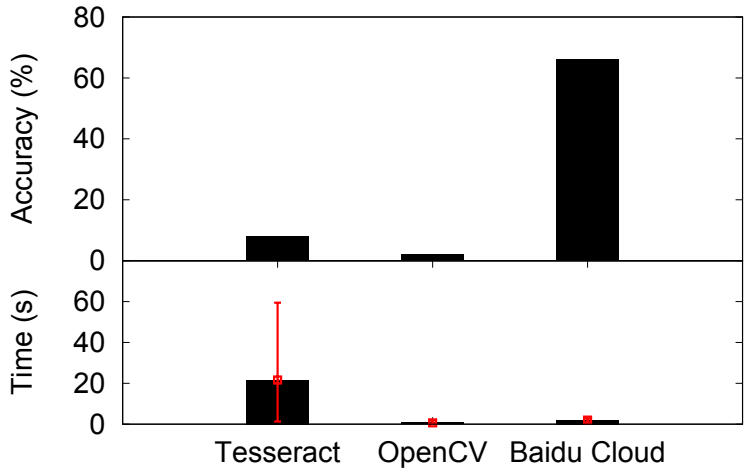


Figure 2.14 Different text recognition approaches.

2.3.1.2 The influence of image resolution

We further explore the performance of Baidu Cloud by using different image resolutions. We vary the resolution of the 100 images from 180p to 1538p. The performance is shown in Figure 2.15. We can see that both text recognition accuracy and processing time increase with the increase of image resolution. When the image resolution is 720p, the text recognition accuracy and average processing time are 54% and 0.74s. In comparison, the text recognition accuracy and average processing time increase to 72% and 1.89s when the image resolution increases to 1536p. EyeLoc chooses I_p to keep the text recognition accuracy higher than 60%. Meanwhile, EyeLoc minimizes the expected time for successfully recognizing a POI which is the ratio between the average processing time and the recognition accuracy. Hence, we set I_p as 1080p. For outlier view image filtering, according to the observation of Figure 2.8 and Figure 2.9, we set Δ_{Lap} and Δ_{ω} as 80 and $10^{\circ}/s$.

2.3.2 Circle Shoot Operation

We set f_s as 2Hz, namely EyeLoc shoots 2 view images per second. The step length for sparse image processing Δ_s is set to be 3. According to our empirical experience, 20% of view images are observed to be blurred (Figure 2.8) and 37% of 1080p view images encounter text recognition

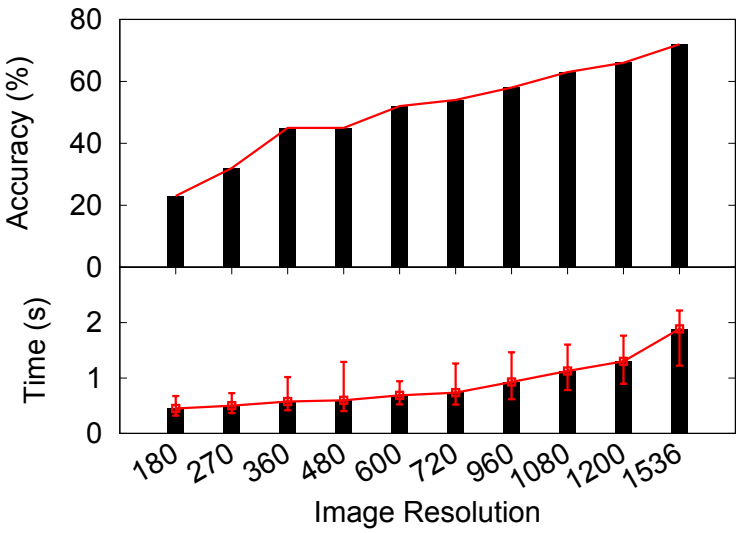


Figure 2.15 Different image resolution choices.

failure (Figure 2.15), it is better to obtain at least 6 view images of a POI (e.g., 3s) to ensure the reliability and efficiency of POI extraction. That means if there are 5 POIs around a user, the circle shoot will take 15s at least.

2.3.3 Floor-plan Images

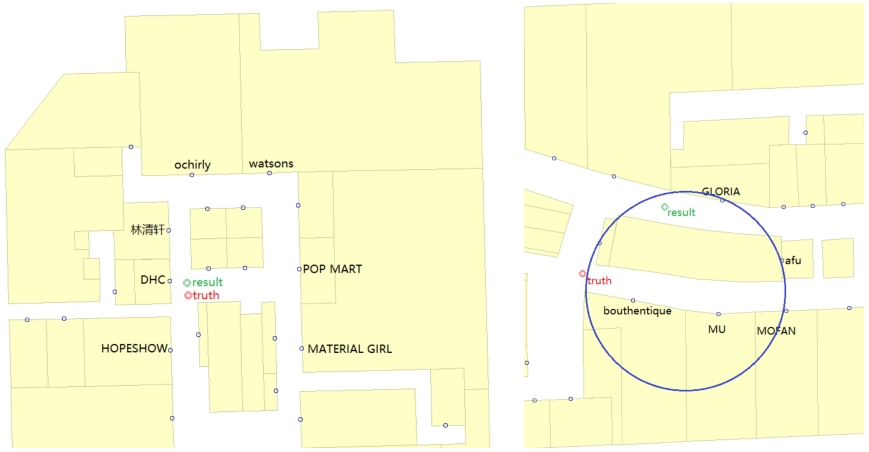


Figure 2.16 Two experiment positions in two shopping malls.

EyeLoc generates high-resolution indoor floor-plan images from Gaode Maps. Since the font of POI texts on floor-plan images is in regular print format, the text recognition accuracy is close to 100%. The resolution of the floor-plan image is set to 2560×1440. Using the floor-plan image, however, the texts of different shops are often placed in the middle of the shops' blocks, while the

POIs captured by EyeLoc are shop signs which are often located on the entrances. The caused direction and coordinate errors can further lead to possible localization errors. To mitigate the influence of the floor-plan error, we refine the shop coordinates with fine-grained floor-plan data fetched from Gaode Maps. The floor plans are shape files containing several image layers, depicting shops, roadmaps, and doors. In our experiments we directly use the coordinates of the "doors" as the location of POIs (e.g., the blue circles in Figure 2.16). The coordinates acquired this way are more accurate and it is beneficial to improve localization quality. Moreover, for error sensitivity estimation, we empirically set Δ_θ and R_e as 10° and 20 pixels. The threshold of text string similarity Δ_t is set to be 50%.

Overall, Table 2.1 summarizes all system parameters of EyeLoc.

Symbol	Description	Value
I_p	The resolution of view image	1080p
f_s	The sampling frequency of view image	2 Hz
Δ_{Lap}	The threshold of view image blur	80
Δ_ω	The threshold of angle velocity difference	$10^\circ/s$
Δ_t	The threshold of text string similarity	50%
Δ_s	The step length of sparse image processing	3
Δ_θ	The error of intersection angle estimation	10°
R_e	The error of extracted POI coordinates	20 pixels

Table 2.1 Summary of system parameters.

2.4 Evaluation

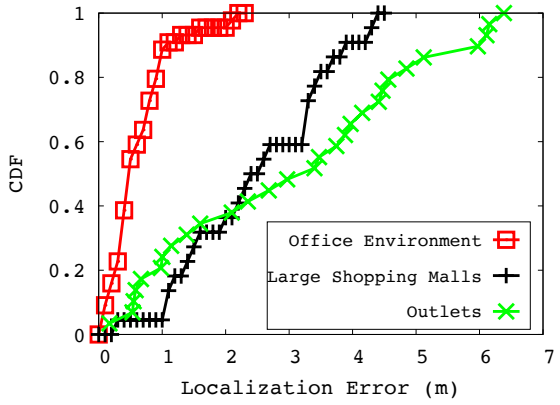
We evaluate EyeLoc with different smartphones (e.g., MI 5 and Huawei Mate 7) in an office environment[1] and two large shopping malls[2]. The office environment is a 7m×9m office room. We print 6 shop signs such as NIKE on A4 papers, then hang them on the wall or curtains in clockwise order. The area of each floor in two large shopping malls is 7,500 m^2 and 10,000 m^2 respectively. The area of the semi-outdoor Outlets is about 70,000 m^2. The distance between adjacent shops and the width of corridors are much larger than office and shopping malls. We invited two volunteers (Male, 20-30 years old) to complete all the experiments both in daytime and at night. User 1 uses

[1]Demos in the office environment: https://youtu.be/v7CT6gTBNEc, https://youtu.be/wCu8STdRG_c, https://youtu.be/iT5pdjO6RVk

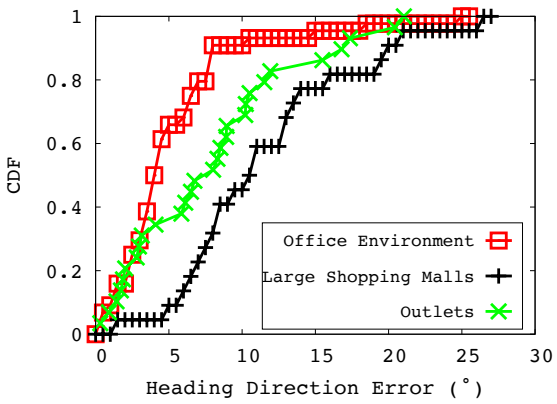
[2]A demo in a shopping mall: https://youtu.be/iHh0R8TkNLo

MI 5, User 2 uses Huawei Mate 7. The two users exhibit different habits as User 1 turns faster than User 2.

In the office environment, we uniformly split the office into 18 areas. Users stand at the center of each area to perform circle shoots. The ground truth is obtained through a laser rangefinder. In the two large shopping malls and the semi-outdoor Outlets, we mainly select the positions near entrances, elevators and bathrooms where users have high localization demands. For each of the 16 positions we evaluated, we also use the laser rangefinder to measure its ground truth. The minimum and maximum distances between the user and a POI are 2.26m and 37.4m in our experiments.



(a) The CDF of localization error in different environments



(b) The CDF of facing direction error in different environments

Figure 2.17 The reliability of EyeLoc.

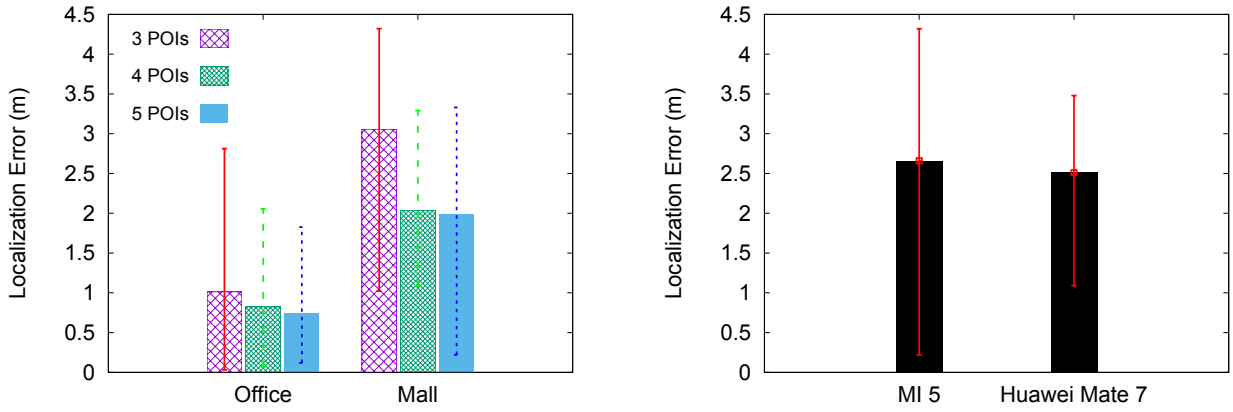
2.4.1 The Accuracy of Localization and Heading Direction Estimation

We first discuss the reliability of EyeLoc. As shown in Figure 2.17a and Figure 2.17b, in the two large shopping malls, the median errors of localization and heading direction are 2.6m and 10.5°. The 90-percentile errors of localization and facing direction increase to 4m and 20°. In the office environment, the 90-percentile errors of localization and facing direction are 1.1m and 8°, which are much better than the performances in large shopping malls. There are two reasons. First, in the office environment, we can precisely measure the POI coordinates on the floor-plan images. However, the POI coordinates suffer larger error due to the position mismatch between the text bounding boxes and the observed POI signs on the floor-plan images obtained from indoor map

providers. Moreover, in the office environment, usually we have only one POI in a view image. However, in large shopping malls, several signs of the same POI may appear in a view image, which will introduce error into the sightline estimation. Given the area as large as 7,500m^2 and 10,000m^2, 4m and 20° is still relatively accurate for the most location-based services.

In 70,000 m^2 semi-outdoor Outlets, the 90-percentile errors of localization and facing direction are 5.97m and 20°. The error of facing direction is comparable with that in shopping malls. The localization error, however, is getting large. The reason is that the distance between user and POIs is larger in Outlets than that in shopping malls. A small estimation error of POI direction can lead to more position estimation errors. Hence, although the facing direction accuracy is similar, the localization error increases in a larger space.

2.4.1.1 The influence of the number of observed POIs



(a) The influence of the number of observed POIs

(b) The influence of smartphone hardware

Figure 2.18 Different influencing factors.

In position matching, EyeLoc utilizes the POI redundancy to mitigate the measurement errors of POI coordinates and EyeLoc sightlines. Figure 2.18a shows the relationship between the number of observed POIs and the localization error. We can see the average localization error decreases as the number of observed POIs increases. Specifically, the localization error decreases by 33.7%/34.9% when the number of observed POIs increases from 3 to 4/5 in large shopping malls respectively. This indicates the error mitigation approaches are effective for improving the

localization accuracy. Moreover, 5 or more POIs cannot provides more useful information for improving the localization accuracy rather than 4 POIs.

2.4.1.2 The influence of POI distribution

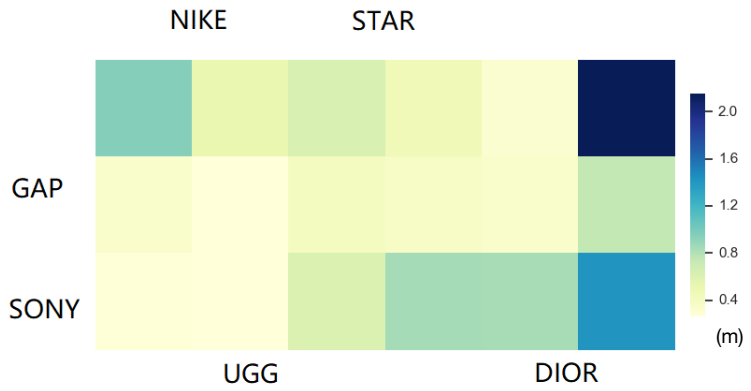


Figure 2.19 The distribution of the localization error in office environment.

We evaluate the influence of POI distribution on the localization error. Figure 2.19 shows the localization error distribution of 18 experiment positions given the positions of 6 POIs. The darker the color is, the higher the localization error is. We can see the localization error is getting higher when the distance between the user's position and POIs is large. Moreover, the top left area is higher than its surrounding area, that is because it is close to the circle formed by several POIs. According to Section 2.2.2, when the user's position and POIs tend to be on the same circle, the accurate localization will be hard to achieve. Moreover, we give two experiment positions in shopping mall 1 (Figure 2.16(a)) and shopping mall 2 (Figure 2.16(b)). The white areas are roads and the yellow blocks are shops. The green points are the results of EyeLoc localization. The red points are the ground truths. The localization error of the position in shopping mall 1 is 1.56m, but that of the other one is 3.64m. From the POI distribution on the floor-plan images, we can see the large error in Figure 2.16(b) is because the true position, GLORIA, AFU and MOFAN tend to on the same circle. In contrast, the position in Figure 2.16 has more POIs which are close to it and uniformly distributed. Hence, the results suggest that the localization error is indeed related to the distribution of surrounding POIs, especially when the user's position and observed POIs are on the same circle. However, the situation only happens twice among total 29 positions. If the situation

happens, we will ask the user to walk a short distance and relocalize himself/herself again.

2.4.1.3 The influence of smartphone hardware

We further evaluate the localization error by using different smartphones at the same positions in large shopping malls. The results are shown in Figure 2.18b. We can see the average localization error is 2.66m for MI 5 and 2.51m for Huawei Mate 7. Since User 2 turns slower than User 1, the more redundant view images make the localization error variance of Huawei smartphone is smaller than MI 5. Overall, EyeLoc works well on both smartphones and does not depends on any smartphone specific hardware and parameters.

2.4.2 Processing Efficiency

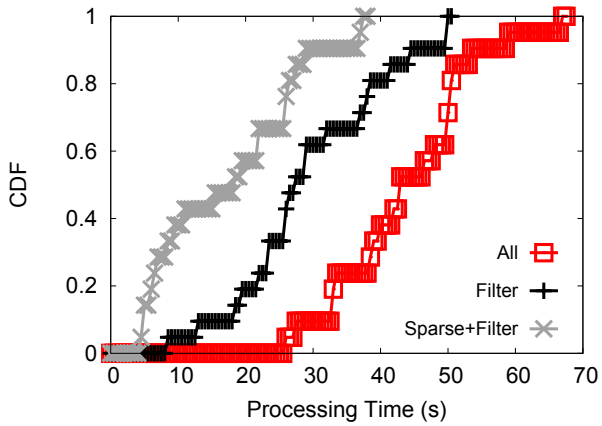


Figure 2.20 The processing efficiency with outlier filtering and sparse processing.

Another important metric is the processing time, which is from the end of circle shoot to the user's position is shown on the screen. Since the view image processing dominates the overall processing time, we designed two approaches, namely, outlier image filtering and sparse image processing. We evaluate the processing time of three methods: "All" indicates we process all view images; "Filter" indicates we only process the view images after filtering the outliers. "Filter+Sparse" indicates we combine outlier filtering and sparse processing approaches. As shown in Figure 2.20, we can see "Filter+Sparse" outperforms the other two methods and its median processing time is 18.2s which is 55.5% shorter than "All". Moreover, the median processing time of "Filter" is 27.3s which is 36.2% shorter than "All". That verifies both outlier filtering and sparse

processing are effective to improve the processing efficiency. In the worst case, the processing time of "Filter+Sparse" is 37.4s. That means the user can obtain the localization result in no more than half a minute after circle shoot. Figure 2.21 further shows the processing time on different smartphones.

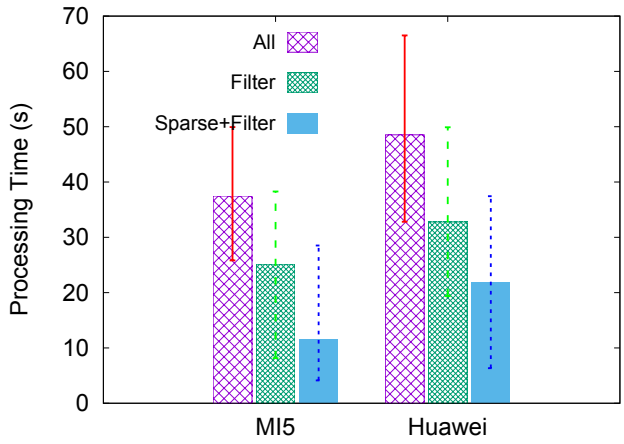


Figure 2.21 The comparison of processing time on different smartphones.

2.4.3 Energy Efficiency

We evaluate the energy efficiency of EyeLoc with Battery Historian [39], which is a tool to analyze battery consumption using Android "bugreport" files. We decompose EyeLoc to 3 function modules which are image recording, image processing and location calculation. To evaluate the energy efficiency of each function module, we set them in infinite loops and continuously run them for an hour. For each experiment, we charge MI 5 to 100% at the beginning. We also evaluate the total energy efficiency of EyeLoc with the same method. For each experiment, we repeat it 3 times and obtain the average value. The results are shown in Table 2.2. We can see that among different function modules, the energy cost of image recording is 1.94 W which is much higher than that of image processing (0.34 W) and location calculation (0.65 W). Hence, the network communication of image processing and computation of location calculations is much more energy-efficient than shooting images with the camera. In our implementation of EyeLoc, we use multi-thread to overlap image recording and processing. The circle shooting usually dominates the operation time of EyeLoc. As a result, the total energy efficiency of EyeLoc is 2.17 W which

is mainly spent on imaging shooting.

Function Module	Energy Efficiency (W)
Image recording	1.94
Image processing	0.34
Location calculation	0.65
Total EyeLoc	2.17

Table 2.2 Energy efficiency of EyeLoc.

2.4.4 Localization Performance Comparison

To compare the localization performance, we implement Sextant [29] and evaluate its performance in the semi-outdoor Outlets. For each shop, we take 3 high-quality images to construct the visual clue and localization map. During evaluation, users are well trained and follow the rules of Sextant to shoot images. For example, users manually keep the POI in the middle of the image. The performance comparison is shown in Figure 2.22. We can see the 90-percentile localization error of Sextant is 6.6m which is 10% higher than EyeLoc. The reason is that EyeLoc can automatically capture the angle of POI which is less accurate by using the user dependent method of Sextant. If users are not well trained, the localization error of Sextant could be increased since the increasing error of POI angle estimation.

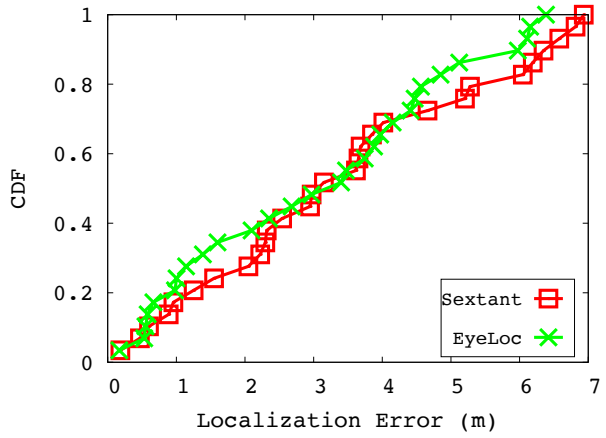


Figure 2.22 The localization performance comparison between EyeLoc and Sextant in semi-outdoor Outlets.

2.5 Related Work

In recent years, many works target on developing efficient indoor localization and positioning systems. However, most of them need some site surveying. Some of them even need custom hardware. According to different types of data sources, we divide existing works into four categories.

• **Wi-Fi Signal** Many indoor localization methods are proposed based on Wi-Fi signals. One approach is fingerprinting-based. The Wi-Fi signal patterns serve as the fingerprint that represents every location. The system manager builds a fingerprint database in the target areas to initialize localization service. The user's location is estimated by matching the measured fingerprint to database records. RADAR [40], Horus [41], Place Lab [42], PinLoc [43] and Smart^2 [44] use site survey to construct the fingerprint database. LIFS [45] and Zee [46] further utilize crowdsourcing to alleviate the burden of labor-intensive site survey. A theoretical analysis around how good a performance the RSS fingerprinting can achieve is provided in [47].

The other approach is model-based. The basic principle is the relationship between the geometric structure from Wi-Fi access point (AP) to user's location and the physical features of the received Wi-Fi signal can be modeled. If the location of Wi-Fi AP is pre-known, the user's location can be inferred. Based on the log-distance path loss (LDPL) model, EZ [48] uses the received signal strength (RSS) to estimate the signal propagation distance and combines several estimations of different APs to find the user's location. SpinLoc [49] and Borealis [50] observe if a user faces an AP, the RSS is usually higher than the user turns his back on the AP. After making a full 360° turn, SpinLoc and Borealis extract the angle-of-arrival (AoA) of several APs to determine the user's location. ArrayTrack [51] uses antenna array and Wi-Fi signal phase to obtain accurate AoA spectrum to calculate a user's location. CUPID [52], SAIL [53], Chronos [22] and Ubicarse [54] further refine the distance and AoA measurement methods to achieve high localization precision or adapt to COST Wi-Fi AP. SpotFi [55] proposes a super-resolution AoA algorithm to extract AoAs from Channel State Information (CSI).

• **Visible Light** In a typical visible light positioning (VLP) system, lamps (fluorescent and LED) are served as landmarks. After a light receiver (smartphone camera or photodiode) obtains sev-

eral lamps' location, the light receiver further measures the geometric structure from the observed lamps to find the user's location. Luxapose [24] takes an image which contains several LEDs as input and fetches LEDs' AoA to calculate the user's location. According to inherent and common optical emission features of both LED and fluorescent, iLAMP [25] and Pulsar [56] identify a lamp's location from a pre-configured database by feature matching. iLAMP further combines camera image and inertial sensors to infer user's location. Pulsar utilizes a custom device to measure the lamp's AoA, then determine user's location.

The other approach is to customize the lamp to establish a mapping function between the location of light receiver and the corresponding received light physical features. CELLI [57] develops a custom LED bulb which projects a large number of fine-grained light beams toward the service area. CELLI adopts a modulation method to encode the coordinate of a fine-grained cell into the corresponding light beam. Thus, the light receiver can obtain its location by visible light communication. SmartLight [26] uses LED array and a lens to form the light transmitter. On LED array, different LED lamps use different PWM frequencies. According to the frequencies of the received light, the coordinate of the observed LEDs circle can be inferred on LED array. The location of light receiver can be further calculated by optical geometric translation function of the lens.

• **Scene Image** Using scene images containing landmark details and architectural features is also a popular direction. SLAM (Simultaneous Localization and Mapping) aims to output a map as well as the user's real-time location. Both 2D and 3D positions are acceptable. We especially introduce SLAM with the camera as its main sensor in this paper. A single image and a floor plan is used as the input in [18] and the main technology is Markov random field. SfM (Structure from Motion) builds a 3D model from 2D images or video. Based on the 3D model, we can know the camera pose and shooting location given another image. iMoon [58] and Jigsaw [59] adopt SfM to construct indoor 3D model and enable localization services. Sextant [29] builds a geometrical model with static reference points and works for a lightweight site survey. All these papers need a collection of images for site survey, which EyeLoc avoids.

• **Others** Magicol [30] and FollowMe [60] combine the geomagnetic field and user trajectory as

the fingerprint to localize user's location. With acoustic speakers as the landmarks, Swadloon [61] and Guoguo [62] use acoustic signal based geometric model and inertial sensors to localize user's location. Shenlong Wang, etc. [18] utilize the floor-plan image and a scene image to localize a user in large shopping malls. Based on edge, text and layout features of a scene image, they use Markov random field model to infer the camera pose on the floor-plan image. However, they need to search through all possible positions which further incurs huge computation complexity. Hence, it is not practical on COTS smartphones.

To conclude, compared with these methods, as shown in Table 2.3, EyeLoc depends on neither pre-deployed infrastructure nor pre-collected information. Moreover, EyeLoc does not depend on any custom hardware and can be implemented as a smartphone application. EyeLoc can achieve the comparable accuracy of localization and heading direction in real large shopping malls which are much larger than the prototype deployment of the existing indoor localization systems.

Technology	**Site Survey**	**Custom Hardware**	**Range (m^2)**	**Localization Error (m)**	**Heading Error (°)**
LIFS [45]	WiFi fingerprint		1,600	9	NA
SAIL [53]	WiFi AP location	√	2,800	2.3	NA
SmartLight [26]	Lamp modification	√	16	0.5	NA
iLAMP [25]	Lamp fingerprint		8	0.032	2.6
iMoon [58]	Environment image		1,100	2	6
Sextant [29]	Environment image		11,250/60,000	20	NA
FollowMe [60]	Geomagnetic fingerprint		2,000	2	NA
EyeLoc	**FREE**		**7,500/10,000/70,000**	5.97	20

Table 2.3 Summary of existing indoor localization systems.

2.6 Conclusion

In this chapter, we propose EyeLoc, a plug-and-play localization system for large shopping malls without the burden of system bootstrap nor calibration. EyeLoc enables the smartphone to imitate human self-localization behavior. After a user opens the EyeLoc application, he/she carries out the circle shoot and the smartphone continuously shoots view images meanwhile. After that, EyeLoc automatically projects the user's position and heading direction onto the floor-plan image. The evaluation results show that the 90-percentile accuracy of localization and heading direction is 5.97m and 20°. Moreover, EyeLoc can be extended to other environments (e.g., office building, train station, airport) where floor-plan and indoor texts are available. We will extend EyeLoc to

these environments in the future.

EyeLoc achieves a good balance among cost, computational power and real-time responses. While alternative localization methods can be developed for large shopping malls, EyeLoc better fulfills the constraints of practical situations such as the low budget requirement of application providers and the low latency requirement of users.

CHAPTER 3

A ROBUST SOUND SOURCE LOCALIZATION SYSTEM FOR VOICE ASSISTANTS

Acoustic signals serve significant location information. For example, underwater sonar system can detect shipwrecks and fishes, medical ultrasonography images internal body structures. As IoT prospers, a new trend of acoustic localization application arises: sound source localization for voice assistants. To the benefit of Smart Home and Smart Office, machines need ears to perceive the environment. If voice assistants like Amazon Echo, Google Home or Apple HomePod can locate a person based on the speech he/she utters, they can better understand the context of commands and deliver more considerate tasks. For example, with user location known, voice assistants can send commands to TV screen or lights to adjust angles, volume or magnitude, after which the smart system provides better user experience and is more energy-saving. In some cases when voice assistants fail to understand commands directly from speech due to poor recording quality or ambiguous speech, location information can narrow down the possibilities and increase the accuracy of speech recognition. Last but not the least, passive localization woken up by speech is more friendly to user privacy and energy conservation compared to continuous camera monitoring, especially for places like fitting rooms and high-security labs.

Although sound source localization can empower voice assistants with fancy functions, it is challenging to determine the location of the sound source in three dimensions: azimuth, elevation and distance. Auditory distance perception is mainly achieved through monaural features like intensity loss or frequency loss, while Direction-of-Arrival (DoA) estimation uses binaural cues like phase difference or Time Difference of Arrival (TDoA) [63]. Although auditory distance perception and DoA estimation are developing into two subfields now and struggling with their own issues, they have a close relationship and sometimes share new ideas. Here we provide a comprehensive research survey to discuss the main challenges to implement a sound source localization system for voice assistants and the reasons behind, which can also be seen as the history of sound source localization:

• **Auditory distance perception is extremely challenging when the signal of interest is un-**

known. Theoretical research[64] has quantified loss of intensity and change of frequency as distance increases in the ideal case. However, to put the theory into practical applications, we need to know the initial intensity and spectral of the sound. Although voice assistants know their wake-up commands, they stay blind to the intensity and spectral shape of the original sound all the time. Experiments in [65] further shows that accurate distance judgements are unlikely to happen without intensity and spectral shape recorrelated with the environment on the basis of experience. Recent research make progress from three perspectives. (1) A machine learning model is proposed in [66] to memorize variation patterns contained in training data. The output falls into different distance classes (0 m, 0.5 m, ..., 3 m). However, the localization accuracy is constrained by the granularity of labels. Experiments further reveal the performance significantly decays when the system is trained on data collected from room A while tested on data of room B. (2) By combining auditory cues with visual cues, distance perception can be achieved. (3) Some IoT systems allow prior information of the transmitted signal. Earlier works [6, 67] achieve acoustic ranging between smartphones using impulses. Later Frequency-modulated Continuous-Wave (FMCW)[68, 69] is introduced to track smart devices.

• **The accuracy of DoA estimation shows instability due to multipath effect and background noise.** By analyzing the difference of signals received by two microphones, we can estimate the DoA according to Far-Field Effect[70, 71]. One single DoA will lead to Cone of Confusion[72]. With three precise DoAs from multiple non-collinear microphone pairs, we can retrace the 3D location through intersection of three cones. Unfortunately, a minor fluctuation of DoA estimation leads to non-ideal retrace. Over the years, the majority of research on sound source localization focus on improving the accuracy of DoA estimation. One category builds its foundation on cross-correlation[73–76]. It aligns two signals after delaying one signal. The best match corresponds to the most possible TDoA. In practice the existing of noise obfuscates the peak of cross-correlation and decreases the accuracy. Studies from[75, 77] assume the background noise follows Gaussian distribution and add a phase weight derived from magnitude squared coherence function. However, this method cannot deal with multipath effect. Another popular category[78–80] is based on

MUSIC[81]. MUSIC builds signal covariance matrix and extracts eigenvalues. Although it is capable of locating multiple sources, MUSIC assumes signals from different sources are uncorrelated as well as noise. With multipath effect existing, the assumption is compromised.

• **Voice assistants expect a system without extra deployment expense or prerequisite operations.** Due to two challenges above, current employable sound source localization systems turn to adding another sensor (depth sensor, camera, etc) or collecting prior information ((user height, environment space structure, etc). If multiple microphones[8, 62, 82–85] scatter around the place or multiple anchor speakers[9, 86–90] serve as beacons, geometric models can be built after strictly aligning the clocks across different microphones and speakers. Owlet[91] designs a 3D-printed metamaterial structure to obtain spatial information. Recently VoLoc[92] proposes to turn multipath effect from disturbance to information as it extracts location information from the second reflection path. After collecting the user height as prior information, VoLoc achieves 2D localization. However, its application scenario is limited as it requires voice assistants to be near a wall. Symphony[71] achieves multi-source localization but still requires the relative position of the microphone array to a nearby wall. MAVL[93] extends the application scenario of VoLoc as it does not require a wall nearby. However, MAVL needs to estimate the reflectors in the room by emitting wideband chirps before localization the sound source.

To solve the first challenge, we inherit the idea of TDoA. As long as TDoA between a pair of microphones is available, we can turn it into distance information and further 3D location. Instead of cross-correlation, we approach TDoA by formulating our problem into linear regression as $\theta = \omega \cdot \tau + \varepsilon$ where ω is angular frequency, τ is TDoA and ε is noise term. θ is the phase difference between two microphones obtained from cross spectrum. Under this formulation, we get a better visualization of acoustic data and the ways to remove outliers become more explicit, which further helps the system on solving the second challenge. To be more accurate, $\theta = \omega \cdot \tau + \varepsilon$ is a theoretical representation which only stands when there are only LoS path and uncorrelated noise. With the existence of multipath effect, the assumption is compromised and the performance of linear regression becomes unstable. To compress the disturbance of multipath effect, we extend

the idea of robust regression from [94]. Compared to linear regression, robust regression will dilute the impact of outliers on loss function. The main difference between [94] and our work is, [94] tests robust regression in simulated experiments. They modulate signals to simulate multipath environment and Gaussian background noise. However, our experiments are completely performed in real-life scenarios. The noises are more complicated and cannot be perfectly handled by robust regression. Moreover, the phase difference between two microphones in real life can be larger than 2π if the user has a high voice, but phase data from spectrum always falls into $[0, 2\pi]$. We make adjustments to accommodate robust regression to practical applications. As for hardware noise and background noise, we propose self-adjusting speech detection algorithm to predict the probability of existence of speech in a certain frequency bin. We also remove outliers which are in conflict with the known distances between each pair of microphones. The known distance projects certain constraints for TDoA estimation, which can be applied to filter out outrageous phase data.

We implement SoundFlower on a 6-mic circular array and Raspberry Pi as a simulation to Amazon Echo. We also implement state-of-the-art work VoLoc[92] as our baseline model. We collect 1,000 data points from different indoor environments and test on SoundFlower. The overall accuracy of 2D localization and 3D localization is 0.45m and 0.5m with consumption time of 3s and 5s, which is sufficient for voice assistants. Around 700 data points are collected next to a wall and are tested on both VoLoc and SoundFlower for 2D localization as VoLoc requires user height as input.

Our contributions can be summarized as:

• We propose the design of SoundFlower, a robust and real-time sound source localization system for voice assistants. We obtain phase difference from cross spectrum, extract TDoA from phase data, turn it into distance, and derive 3D localization through lightweight optimization.

• We formulate TDoA and phase shift into a robust regression problem. Robust regression shows great performance on compressing multipath effect. We also propose self-adjusting speech method to detect speech-involved phase data and an unwrapping scheme to rectify phase data from periodicity issue.

• We implement SoundFlower and compare it with state-of-the-art work. While extending previous work from 2D localization to 3D localization, we achieve comparable localization accuracy to state-of-the-art work.

The following content of this chapter is organized as follows: We discuss related work about acoustic localization in Section 3.1. Section 3.2 establishes mathematical preliminary about TDoA estimation and our motivation to SoundFlower. Section 3.3 is an overview of SoundFlower. We further introduce the system in detail in Section 3.4. Section 3.5 shows our implementation details as well as experiment setup. Section 3.6 presents experiment results. We analyze our limitations in Section 3.7. Finally we conclude our work in Section 3.8.

3.1 Related Work

Acoustic localization can be done passively or actively. Active acoustic localization involves the modulation and transmission of the signal of interest. Traveling along different paths leaves fingerprints on the specially-designed signal. The location information can be extracted through comparing the transmitted signal and the received signal. Passive acoustic localization waits for the sound from the target. Usually limited spectral information about the original sound is available. The location is usually obtained with the help of multiple microphones with known relative positions.

Active Acoustic Localization Active acoustic localization designs distinguishable signals. After the receiver captures the signal, several mechanisms can figure out the target location. Some systems[6, 8, 62, 87] are ranging-based by measuring TDoA. Some[68, 95–97] keep track of the moving velocity of the target through Doppler Effect. There are also some papers[89, 98–100] exploiting the phase shift of the signal and then derive its traveling distance. Some[69, 101, 102] combine acoustic signal with inertial sensors and achieve localization through tracking. Up to now, TDoA based models are constrained by the accuracy of timing, Doppler Effect based models are limited by the frequency resolution while phase shift based models only apply to scenarios whose phase shift is below 2π.

Passive Acoustic Localization Our paper falls into passive acoustic localization. Passive acoustic

localization is widely studied for applications like voice assistants and self-driving cars, where the target is located as soon as they make a sound like wake-up commands or ambulance sirens. Unlike active acoustic localization, the receiver usually does not have comprehensive knowledge about the spectrum of the sound of interest. Thus passive sound source localization uses multiple microphones with known positions to derive AoA of the sound and then locate the source. One classic technique is cross correlation[73–76, 103], which computes TDoA between microphones through cross correlating their signals. Another large category is based on MUSIC[78–81, 104], which analyzes spatial covariance matrix of microphone signals, extracts signal subspace from noise subspace and peeks the AoA which maximizes the energy. We specially introduce VoLoc[92] as we will use it as the baseline model. Multipath effect is believed to be one of the main interferences to acoustic localization. Unlike other models trying to get rid of it, VoLoc turns multipath effect into information by putting the voice assistant next to a wall. With wall distance and orientation known to the voice assistant, VoLoc successfully extracts the second AoA, namely the AoA of the wall reflection path, and achieves 2D localization after fusing two AoAs and the user height. MAVL[93] does not require voice assistants to be next to a wall, but before localization it has to transmit FMCW signals from 1 kHz to 3 kHz for AoA estimation.

3.2 Preliminary and Motivation

In Section 3.1, we summarized various categories of acoustic localization methods. Our paper falls into the category of TDoA estimation of passive acoustic localization. Suppose the signal transmitted from the source is $s(t)$, the signals captured by two microphones $x_1(t)$,$x_2(t)$ can be represented as:

$$\begin{aligned} x_1(t) &= h_1(t) * s(t) + n_1(t) \\ x_2(t) &= h_2(t) * s(t-\tau) + n_2(t) \end{aligned} \tag{3.1}$$

where $h_1(t)$ and $h_2(t)$ are channel state information, $n_1(t)$ and $n_2(t)$ are additive noises uncorrelated to our source, and τ is the TDoA of interest. As Figure 3.1 shows, once τ is at hand, we can derive the difference between the distances of two microphones to the source, which could further

contribute to the derivation of the source location.

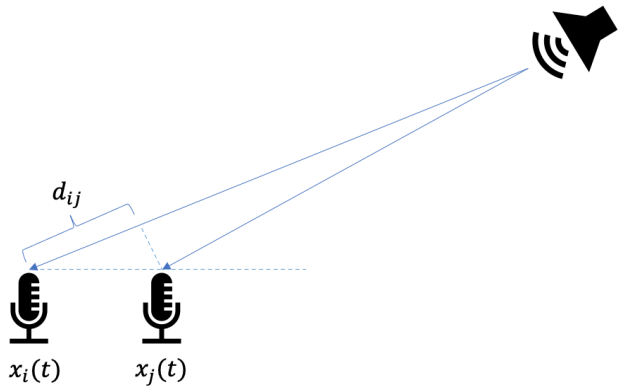


Figure 3.1 TDoA reveals the distance difference between two microphones to the source.

Now that we track our problem from location to TDoA, one fundamental and classic method for TDoA is General Cross Correlation (GCC). Roughly speaking, GCC tries all possible TDoAs and performs cross correlation on two signals with one signal delayed by the assumed TDoA, the TDoA which generates the peak cross correlation result is believed to be the actual TDoA $\hat{\tau}$. In equation 3.2 we use ω to represent angular frequency. Cross correlation $R_{x_1x_2}(\tau)$ is calculated through the inverse Fourier Transform of the cross spectrum $X_1(\omega)X_2^*(\omega)$. The weighting function $W(\omega)$ is to show emphasis on different frequencies in the presence of uncorrelated noise[75].

$$\begin{aligned} \hat{\tau} &= \underset{\tau}{\operatorname{argmax}} R_{x_1x_2} \\ R_{x_1x_2}(\tau) &= \int_{-\infty}^{\infty} W(\omega)X_1(\omega)X_2^*(\omega)e^{j\omega\tau}d\omega \end{aligned} \tag{3.2}$$

$W(\omega)$ brings up a challenge as effective cross correlation needs to focus on frequencies from speech rather than noise under the condition that we have no information about the original speech. Previous works like [75, 105] use magnitude squared signal coherence. The idea works well when the noise is ideal and follows Gaussian distribution but shows instability to non-stationary noise and multipath effect.

To work on non-stationary noise, we bypass $W(\omega)$ and turn to cross spectrum $X_1(\omega)X_2^*(\omega)$. The phase of cross spectrum can be represented:

$$\Theta(\omega) = \omega\tau + \varepsilon \tag{3.3}$$

where τ is TDoA and ε is noise term. As we can see, the problem has been transferred from TDoA to phase. If we have the phase of cross spectrum, we can figure out TDoA and then derive the location of source. However, equation 3.3 only stands when there is only LoS path existing and ω is from speech frequency. In real-life scenarios, multipath effect and other uncorrelated noise will taint the phase data. Facing the disturbance of multipath effect, VoLoc[92] proposes an idea by extracting the second path, which is speaker-wall-microphone reflection path, to turn disturbance into useful location information. However, the idea requires the voice assistant to be near a wall. Whenever the voice assistant is moved, VoLoc has to re-evaluate the distance and orientation from the voice assistant to wall, which takes hours according to the paper. In this paper, we propose robust regression to deal with multipath effect. Signals from NLoS is weak compared to LoS signal. Thus robust regression will see patterns from NLoS paths as outliers and concentrate on LoS pattern. As for uncorrelated noise, we filter them out before feeding phase data to robust regression.

To summarize, phase data from cross spectrum contains distance information which could lead to sound source location, the challenge is raw phase data from commercial microphones is very noisy. The cause of noises come from: (1) environmental noise; (2) internal hardware noise; (3) multipath effect. Facing the three challenges, we propose SoundFlower. Especially, we propose self-adjusting speech detection in Section 3.4.3.1 to recognize speech from environmental noise and internal hardware noise, and robust regression in Section 3.4.2.2 to deal with multipath effect.

3.3 System Overview

Figure 3.2 illustrates the overall architecture of SoundFlower. After the user speaks a command and the microphone array captures it, we perform self-adjusting speech detection to the sound samples collected by each microphone. During this step, we calculate spectrum and recognize frequency bins which are closely related to the speech. On the other hand, we calculate cross spectrum of each pair of microphones, and collect speech-involved phase data. After that, the filtered phase data is fed to robust regression. Robust regression outputs the TDoA of the speech to two microphones. With sound traveling speed 343 m/s, we obtain the distance difference from the

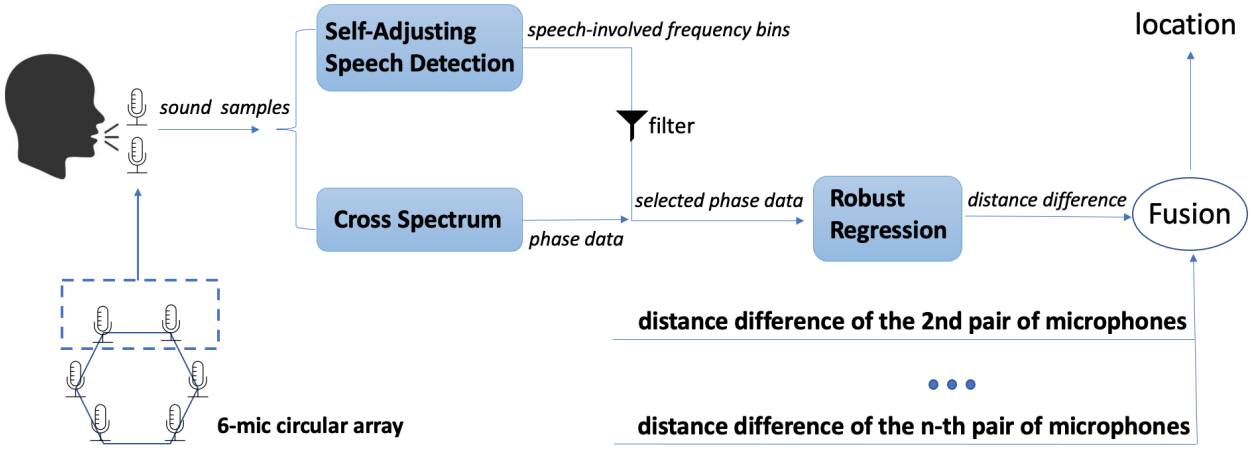


Figure 3.2 System architecture.

two microphones to the source. As we use a circular microphone array which has 6 microphones, we can have up to 15 distance differences. We fuse distance difference information from all pairs of microphones into an optimization model. By exhaustive search, we obtain the final user location.

3.4 Design

In this section, we elaborate the design details of SoundFlower and explain the motivation of each step. After the user utters a command, sound arrives at multiple microphones at slightly different times. The TDoA information encloses the location of sound source. We formulate TDoA estimation as regression problem where frequency is independent variable, phase difference is dependent variable and TDoA is weight. We also introduce cross spectrum to obtain the phase difference between each pair of microphones. In practice, the phase data is easily-polluted. We analyze the sources of noises and propose corresponding solutions. Finally, we locate the target in 3D space through optimization.

3.4.1 Phase from Cross Spectrum

Cross spectrum is the Fourier Transform of cross-correlation result. It describes the relationship between two time series as a function of frequency. Suppose we have two signals x, y captured by a pair of microphones. During cross-spectrum calculation, we first apply Fourier Transform to

time-domain signals and get X, Y. Then we multiple X with the conjugate of Y and get,

$$X \cdot Y^* = a_1(\cos\theta_1 + j * \sin\theta_1) \cdot a2(\cos\theta_2 - j * \sin\theta_2)$$
$$= a_1 a_2(\cos(\theta_1 - \theta_2) + j\sin(\theta_1 - \theta_2))$$

As we can see, the phase of the resulting cross spectrum is the phase shift between the original two signals as long as their phase shift is smaller than 2π. If the distance between two microphones is d and the sound speed is c, the phase shift could be correctly tracked if we only use frequency under $\frac{c}{d}$. For frequencies larger than $\frac{c}{d}$, we unwrapped the phase data as introduced in Section 3.4.3.2.

3.4.2 TDoA Estimation

After we have phase information, we use robust regression to extract TDoA between each pair of microphones, which later will be used in 3D localization.

3.4.2.1 Linear Regression

Before we discuss the technical details of our model, we use linear regression to formulate our problem and project phase data to time delay. Considering that the phase of the cross spectrum θ changes linearly with its frequency f, we have:

$$\theta(f) = 2\pi\tau \cdot f + \varepsilon$$

where τ is the TDoA of the signal arriving at two microphones and ε is the noise component. Under ideal scenarios, ε follows zero-mean, uncorrelated Gaussian distribution[77]. Thus we have the following optimization problem:

$$\hat{\tau} = \arg\min_{\tau} \int (\theta(f) - 2\pi f\tau)^2 df$$

Now we can see the original problem turns into a linear regression problem.

3.4.2.2 Robust Regression

Compared to linear regression, robust regression is more resistant to outliers. Theoretically linear regression is a perfect solution to TDoA estimation in Gaussian noise only scenarios, however in practice phase data is biased by multipath effect. The disturbance is extremely complicate as

noise incurred by multipath effect is not uncorrelated to the speech. Thus we use robust regression as its loss function is more insensitive to outliers. Robust regression will concentrate on the direct-path phase data as they are more weighty.

The objective function of robust regression[106, 107] is:

$$\hat{\tau} = \arg\min_{\tau} \int \rho\left(\frac{\theta(f) - 2\pi f\tau}{S(f)}\right)^2 df$$

where $S(f)$ is a scaling term of residual and $\rho(x)$ is a loss function of scaled residual x:

$$\rho(x) = \begin{cases} -\frac{(1-x^2)^3}{6}, & |x| \leq 1 \\ 0, & |x| > 1 \end{cases}$$

As we can see from Figure 3.3, when we have outliers caused by multipath effect or accumulated error from phase unwrapping (which we will illustrate in Section 3.4.3.2), robust regression can restrict the influence of outliers, focus on coherent data points, and generate output that is more related to ground truth, while linear regression covers all data points and outliers will distract the output from ground truth.

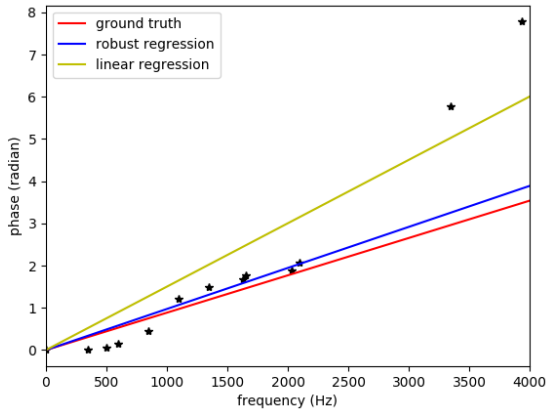


Figure 3.3 Comparison between robust regression and linear regression in presence of outliers.

Robust regression is proposed in [94] as a solution to reverberation. In their experiments, GCC-ML[108] shows great performance in the noise only situations, while robust regression outperforms GCC-ML when reverberation exists. However, their experiments use simulated data.

They simulate a room with plane reflective surfaces and frequency-independent reflections. To substantiate and quantize multipath effect, room impulse responses are modulated with the image technique[109]. Signal-to-noise (SNR) is controlled by adding zero-mean Gaussian noise with a fixed energy level. When we test robust regression in real-life scenarios, the phase data is more tainted and biased than we have expected. One significant reason is: the speech will not cover the whole frequency band (if the sampling rate of ADC is 16 kHz, then the frequency band is 0 - 8 kHz), so we should not use phase data from all frequencies. To accurately recognize the frequency bins in which speech exists, we design a self-adjusting speech detection method in 3.4.3.1 to predict the probability of presence of speech in each frequency bin. We are inspired by a noise estimation method MCRA[110] but modify it to make it simpler, computationally efficient and suitable with short speech like wake-up commands of voice assistants.

3.4.3 Data Pre-processing

When we acquire phase data from cross spectrum, we have phase data of all frequency bins. If we feed the complete phase data directly to robust regression, two issues arise: (1) Speech does not exist in all frequency bins. With too much phase data from irrelevant frequency bins, robust regression will be overwhelmed by outliers and fail to recognize the real pattern. (2) Phase information provided by spectrum is given in the format of complex number. We may transform it into angle in radians, but all the resulting radians will be in $[0, 2\pi]$. Thus we propose Self-Adjusting Speech Detection in Section 3.4.3.1 to clean phase data, and unwrap phase data in Section 3.4.3.2. After cleaning and reforming, it will be easier for robust regression to extract TDoA from the pre-processed phase data.

3.4.3.1 Self-Adjusting Speech Detection

Being blind to the speech makes it challenging to separate speech and noise from the captured signal. Due to hardware imperfection and the way ADC works, it is inevitable to have internal noise, and at this point it is almost impossible to estimate the internal noise. On the other hand, environmental noises, like heating noise and fridge noise, are quite common in the workplace of voice assistants. Previous works either uses magnitude squared signal coherence [75, 105] or

record silent intervals as prior information and use it to estimate SNR[108]. These methods either assume the noise to follow Gaussian distribution or require silent intervals as priori knowledge to estimate noise power spectrum. For non-stationary noise like heating noise and fridge noise, we need a self-adjusting speech detection method to recognize the speech.

In spite of noises, the signal is dominated by speech when the user starts talking, and the magnitude of speech frequency sharply decreases when the speech stops. Based on this phenomenon, a natural solution could be setting a threshold and filtering out the data points whose frequency magnitude is lower than the threshold. However, as the user may speak at different volumes, it is hard to decide on a numeric value for the threshold that works every time. Thus we propose Self-Adjusting Speech Detection to predict the probability of speech existing in a frequency bin.

Speech presence in a frequency bin of current time frame is determined by the ratio between the energy of the current frequency bin and its minimum within a specified time window. Suppose the magnitude of the i-th frequency bin is $M(f,t)$. We keep track of the last L frames of Short-Time Fourier Transform (STFT) results. The minimum magnitude of the frequency bin in the last L frames is $Mmin(f,t)$. By comparing the ratio $M(f,t)/Mmin(f,t)$ to a threshold δ, we can decide speech exists in the i-th frequency bin if $M(f,t)/Mmin(f,t) > \delta$. The number of frames L is determined empirically according to how fast the user speaks. In our experiments, we use a frame of 512 samples, a step of 128 samples and L is 10. As for δ, it is closely related to the ratio between the energy of speech and that of the noise. In practice, we coarsely detect the arrival and end of the speech by monitoring the energy increase and decrease, calculate the power when speech exists, and compare it with that of a silent interval. In this way, we obtain an estimation of δ.

Figure 3.4 shows how frequency 343.75 Hz varies as the user speaks the wake-up command. In spite of the fact that we have 91 data points on record from the whole speech window, we only collect 25 of them as self-adjusting speech algorithm thinks there is high chance that the other data points are from internal noise or environmental noise instead of the speech. You may notice from Figure 3.4 that there are some data points whose magnitude is at the peak but have not been

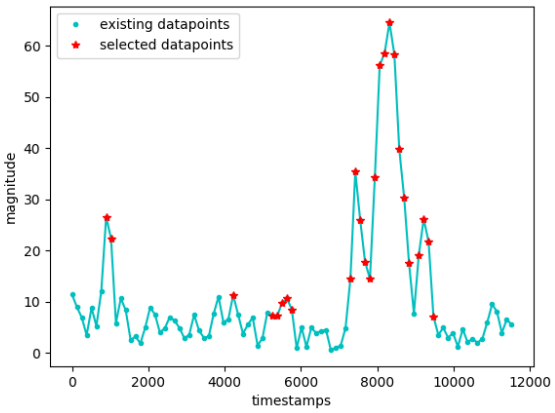


Figure 3.4 Self-adjusting speech detection.

selected. This is because the decision is made mutually from a pair of microphones. The speech detection result from the other microphone does not believe these points are from the speech.

3.4.3.2 Unwrapping Phase Data

The phase information is encapsulated in the form of complex number in cross spectrum as the consequence of FFT. Directly unwrapping the complex number through inverse tangent will result in a value between $[0, 2\pi]$. On the other hand, the maximum distance between a pair of microphones from our microphone array[111] is 0.092 m, which means the phase difference between two microphone could be larger than 2π if the upper-bound frequency bin is greater than 3731 Hz. Thus we add an unwrap step before feeding the phase data to robust regression in order to restore phase information larger than 2π. During the unwrapping, not only do we rectify the periodicity issue, but also more outliers are removed as they break certain rules under known distance constraints. After unwrapping, the data provided to robust regression is more clean and lightweight.

The prior information for unwrapping is the known distances between each pair of microphones, which leads to two rules: (1) there is an upper bound for phase difference between each pair of microphones. For example, if the distance between a pair of microphones is d, then their maximum phase difference is $\frac{2\pi fd}{c}$ where c is the sound speed. This rule is a general boundary for all phase data. Any phase data which is greatly larger than $\frac{2\pi fd}{c}$ or smaller than $-\frac{2\pi fd}{c}$ is

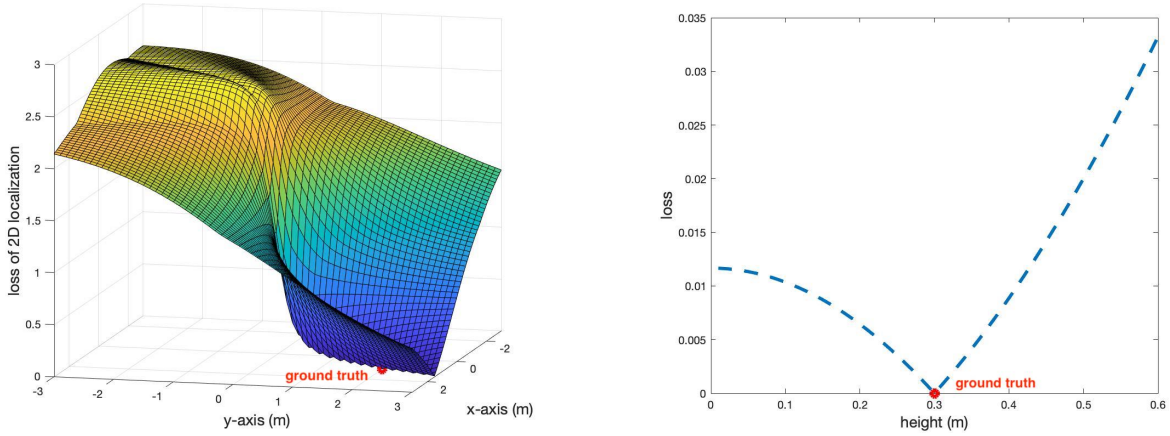
outlier. (2) The phase difference between two adjacent frequency bins $f_1, f_2 (f_1 > f_2)$ is between $[-\frac{2\pi(f_1-f_2)d}{c}, \frac{2\pi(f_1-f_2)d}{c}]$. If the rule is broken, we check if phase data could follow the rule after adding $2k\pi$ to it or minus $2k\pi$ ($k = 1, 2, 3, ...$), otherwise we label it as outlier and remove it. This rule especially works for phase data which are larger than 2π.

During implementation, we make two more practical adjustments. First, we slightly expand the theoretical threshold of two rules to leave certain error-tolerant space. We allow error space of $\frac{\pi}{36}$ to rule (1) and $\frac{\pi}{18}$ to rule (2). The rationale is, some experimental phase data is very close to ground truth but contains a little fluctuation. We keep them as they still provide valuable information. Second, we try our best to catch the information about which microphone receives signal first. This information, if reliable, directs unwrapping to either positive phase or negative, which further increase the accuracy of robust regression as it receives less outliers. Generally, we monitor the energy increase of each microphone. The first microphone which has an energy increase is considered to be closer to the speaker. However, this method is not always reliable as minor energy increase could be overwhelmed by noise fluctuations. So we only consider the result as informative but not trustworthy. On the other hand, we still unwrap phase data to both positive array and negative array. If the ground truth is positive, the length of positive array would be larger and with greater magnitude, and vice versa. If both energy increase and array length support the same result about which microphone receives the speech first, we consider the result as trustworthy and keep the corresponding phase array. Otherwise we believe the two mechanisms fail to recognize the information for reasons such as the speaker is on or near the median line of two microphones. We feed both positive data and negative data to robust regression so that it could make further decision.

3.4.4 3D Localization

In spite of plenty of papers[73–76, 78–81] studying AoA between two microphones, it is challenging to project AoA into location, especially 3D location. For a linear microphone array, AoA information narrows down potential space to a cone. Even with user height, theoretically there are still infinite points consistent with AoA results. For circular microphone arrays, we have one cone from each microphone, predicting the intersection of several cones easily gets stuck into an unsolv-

able situation in practice. Thus we choose optimization to obtain 3D localization from traveling distance differences.



(a) Loss Variation of 2D Localization

(b) Loss Variation of Height

Figure 3.5 Loss function minimizes at parameters that are close to the ground truth.

After TDoAs between each pair of microphones are available, we reformulate the 3D localization problem as an optimization problem and use grid search to find the optimal spot. If we use (x,y,z) to denote a random point within the searching area S, the optimal source location would be:

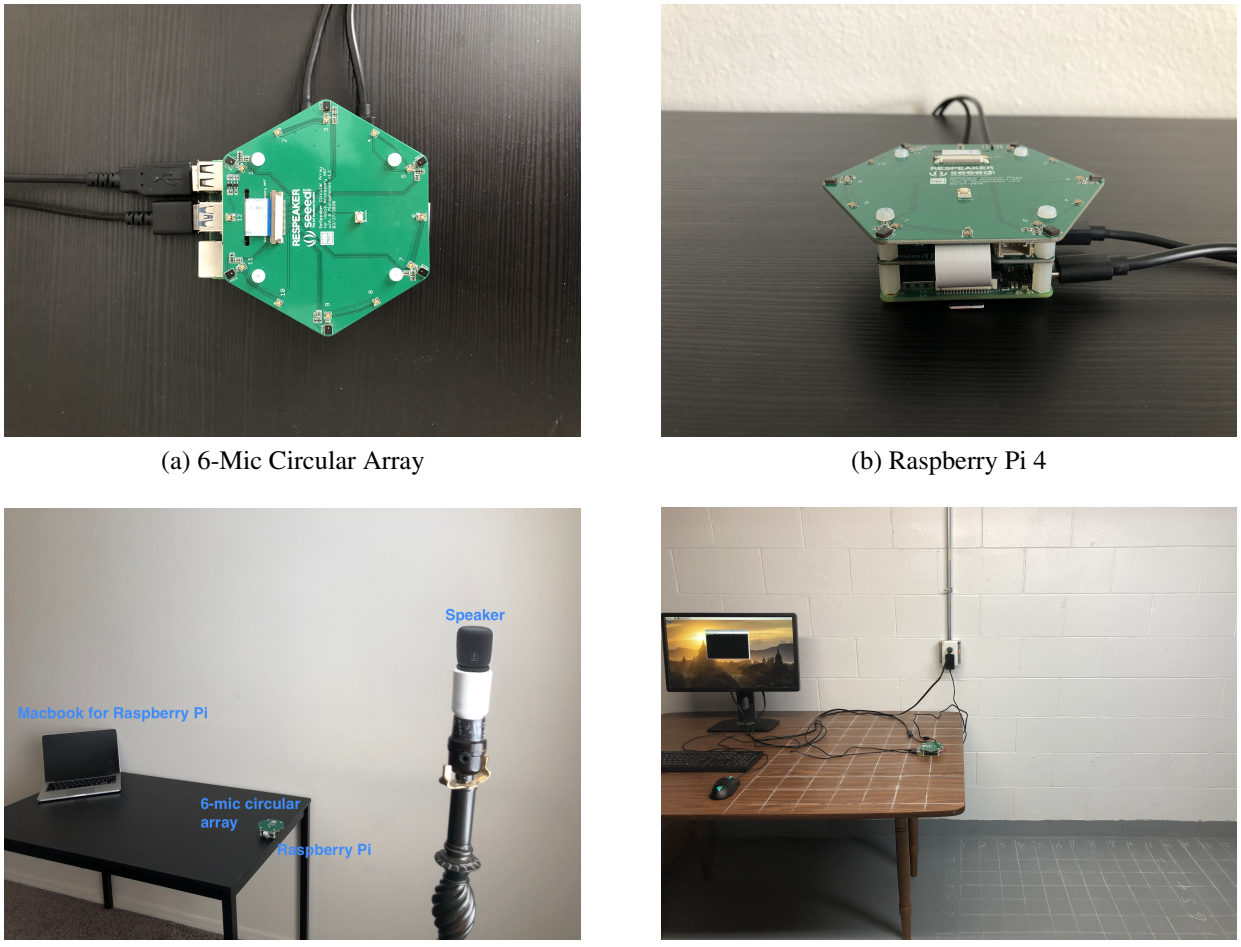
$$x_{opt}, y_{opt}, z_{opt} = \underset{(x,y,z)\in S}{\operatorname{argmin}} \sum_{i,j} |f_{i,j}(x,y,z) - d_{i,j}|$$

where $d_{i,j}$ is the distance difference between two microphones to the sound source derived from TDoA estimation, and $f_{i,j}(x,y,z)$ is the distance difference between two microphones to (x,y,z). Mean average error is convex function. Figure 3.5 shows an example of how objective function varies across the searching area. Figure 3.5a shows loss variation of 2D localization. The loss decreases sharply when approaching ground truth and reaches minimum at ground truth. However, Figure 3.5a also reveals points close to line $y = \frac{y_g}{x_g}x$ have close values to the minimum loss, where (x_g, y_g) is the ground truth and the microphone array is at the origin. Our experiments show similar conclusion, outputs of SoundFlower are around this line once they deviates from ground truth. Figure 3.5b shows the variation of loss function with height after SoundFlower finds the (x_g, y_g). We rationally assume the sound source is higher than the voice assistant as it is often the case in

real life. The technical need for the assumption is because two points symmetric to the horizontal plane of the voice assistant have the same TDoA estimation result.

We use mean absolute error (MAE) instead of mean square error (MSE) due to the fact that MAE is more robust to outliers. With several pairs of microphones available, it is possible some TDoA estimations deviate from ground truth. MAE will prevent loss function from greatly increasing by one or two offset TDoA estimations.

3.5 Implementation



(a) 6-Mic Circular Array

(b) Raspberry Pi 4

(c) Bedroom System Setup

(d) Basement System Setup

Figure 3.6 System setup.

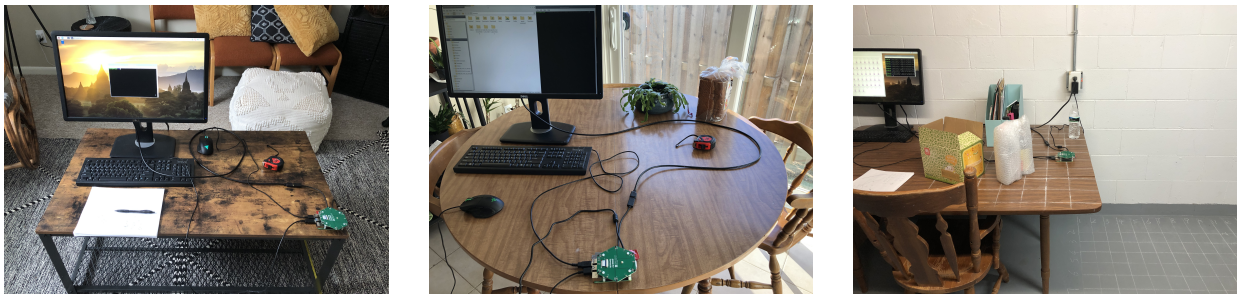
We implement SoundFlower using an assembly of a 6-mic circular array [111] in Figure 3.6a and Raspberry Pi 4 Model B [112] in Figure 3.6b. We use the simulation tool kit instead of off-the-

shelf voice assistants like Amazon Echo because raw acoustic signals are enclosed for commercial products. The sampling rate is set to be 16 kHz as the range could cover most of human voice frequency. Higher sampling rate actually may incur aliasing. The microphone array is mounted over the Raspberry Pi to connect acoustic samples. Afterwards the samples are sent from the Raspberry Pi to a laptop through wireless connection as Figure 3.6c and Figure 3.6d show. We use the laptop to run models and output locations.

We collected 1,000 data points from a bedroom, a basement, a kitchen and a living room like Figure 3.7 shows. To compare SoundFlower and VoLoc in Section 3.6.1, we use data points collected from scenarios like Figure 3.6c, Figure 3.6d and Figure 3.7c to assure the assumption of VoLoc, which is the reflection from the wall is the second sound traveling path. For study on how multipath effect influences the performance of SoundFlower in Section 3.6.2.2, we add objects in Figure 3.6d to complicate the surroundings and generate multipath environment. The overall results shown in Section 3.6.1 covers all different scenarios.

We record the wake-up commands by a small mobile device and place the mobile device in different locations as the sound source. We use prerecorded audio instead of human volunteers because (1) we want to remove volume as a variant and better study the influence of distance in Section 3.6.2.1 and the influence of multipath effect in Section 3.6.2.2. Uncontrollable voice volume uttered from human volunteers will introduce new variant and undermine the reliability of our conclusion. (2) We want to collect ground truth in a more refined manner. This is important as we suggest a practical searching step in Section 3.6.2.3. We aim to find a step size which is accurate enough for human shape size as well as being friendly to the computation capacity of voice assistants.

We use two different wake-up commands, 'Alexa' for Amazon Echo and 'Hi Siri' for Apple products. Both male voice and female voice are tested in our experiments. For evaluation metrics, we use localization error in meters to show the accuracy of SoundFlower, and consumption time in seconds to show the computation overload.



(a) Living Room (b) Kitchen (c) Bedroom System Setup

Figure 3.7 Different experiment scenarios.

3.6 Experiment

In this section, we present the performance of SoundFlower. Especially, we want to find answers to the following questions and evaluate the feasibility of SoundFlower as a sound source system for voice assistants:

• What is the overall localization accuracy of SoundFlower? How is it compared to the state-of-the-art baseline model VoLoc?

• What factors are influencing the overall performance of SoundFlower? What should we target to improve for future development of indoor sound source localization?

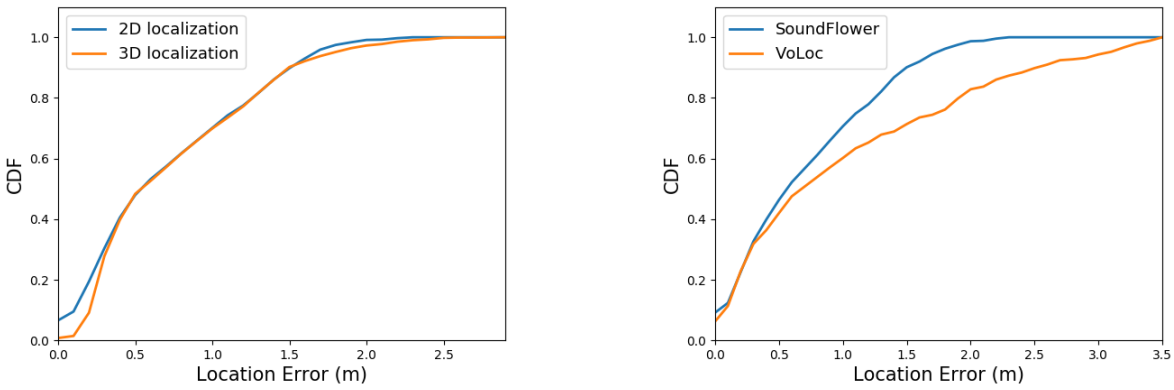
• What is running time of SoundFlower? Is it affordable for voice assistants?

3.6.1 Localization Accuracy

We test SoundFlower with both 2D localization and 3D localization. Considering our target indoor applications like Smart Home and Smart Office, 2D localization information is more of interest. Thus VoLoc chooses to collect user height as input and use it to narrow down the searching space. When there are several family members or workmates, voice assistants will have to recognize different users and choose different heights. Also, human body might lean or be on tiptoe. Having a fixed height value is not an ideal choice to serve our ultimate goal. Thus we choose to search a 3D space of $6m \times 6m \times 0.6m$, which is large enough to cover a basement or conference room and has a height variation of $0.6m$.

We first show our 2D and 3D localization accuracy in Figure 3.8a and then compare 2D localization with VoLoc in Figure 3.8b. The median error of 2D location is 0.45m, while that of 3D

localization is 0.5m. Considering the area covered by a single person, this accuracy is sufficient for location-based applications. Another point worth noting is, the 2D localization error is similar to 3D error. This is because at most locations, the loss function of SoundFlower is more sensitive to horizontal movement than vertical movement of independent parameters as Figure 3.5 implies. Thus with or without user height, SoundFlower is able to find the planar location of sound source. The minor higher increment of 3D localization error is usually from the height prediction offset.



(a) Overall Localization Accuracy

(b) 2D Localization Comparison with VoLoc

Figure 3.8 Localization accuracy.

Among all collected data points, we pick data points which are collected near a wall. VoLoc requires to know the distance and orientation of the voice assistant to the wall. The original paper shows their average error for wall estimation is 1.2cm and 1.4°, but the estimation takes hours every time the voice assistant has been moved. To be more efficient, we feed the information directly to VoLoc. The median error of SoundFlower is 0.5m while that of VoLoc is 0.65m. The accuracy of SoundFlower is slightly decreased because the second reflection path from wall is indeed strong. VoLoc can extract a second AoA information from it, but it will confuse SoundFlower from the LoS signal. If the magnitude of the second path is similar to the third and the fourth path, LoS signal will be more apparent to SoundFlower.

Through implementation and experiments, we analyzed several situations that could hurt the performance of VoLoc. (1) One important assumption of VoLoc is, the reflection of wall is the

second strongest component in the received signal apart from LoS component. For scenarios like 3.6d, the assumption stands and VoLoc receives great performance. However, real-life scenarios are more like 3.7c where different objects may be around the voice assistant. In these cases, the assumption is undermined and the performance goes down. (2) The very first step of VoLoc is to calculate the direct AoA. The resulting AoA will narrow down the searching space from 2D plane to a beam in the 2D plane. This step significantly reduces the running time of VoLoc and makes the computation overload affordable to voice assistants. The side effect is, the estimation of direct AoA needs clean direct-path signal. In other words, we need to clip the signal before the second path signal pollutes the direct path signal. VoLoc[92] states they use "tens of samples" right after detecting the rise of signal energy for direct-path DoA estimation with a sampling rate of 16 kHz. Following the instruction, we choose 32 samples and the same sampling rate in our implementation, which means we assume the second path is at least 0.686 m longer than the first path if the sound speed is 343 m/s. This assumption does not always hold. To extract effective direct path signal, we actually need coarse prior estimation about the distance difference between the direct path and the second path. Otherwise variation on the clipped samples could affect the direct-path AoA estimation, and further leads VoLoc to wrong searching area or take too much time to finally reach a location.

3.6.2 Influencing Factors

In this part we analyze the factors that influence the performance of SoundFlower. Distance and multipath effect are known to be influential to sound source systems[92, 94], our experiments show distance is still the most influential factor to localization performance. We also show the impact of switching searching step size. This is important as we make trade-offs between accuracy and consumption time. Our experiments show if a searching step size cannot cover the ground truth, usually it turns to the closest searching point. Switching to a larger searching step size will not significantly affect accuracy but can decrease consumption time.

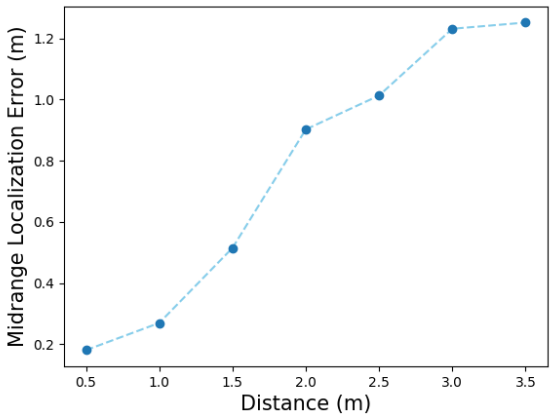


Figure 3.9 The influence of distance to localization accuracy.

3.6.2.1 Influence of Distance

Both VoLoc[92] and our experiments observe great impact of distance on localization accuracy. The further the speaker is to the microphone, the greater the attenuation of acoustics would be. SNR ratio would considerably decrease. After quantization of ADC, some information will be lost. Figure 3.9 shows the decrease of localization accuracy of SoundFlower as distance increases. We use mid-range error, namely the arithmetic mean of the largest and the smallest observed errors, to show the influence. For area within 1m to the microphones, the localization error is always smaller than 0.5m. For area around 3m - 4m to the microphones, the maximum observed error is 2.8m while the smallest error is 0.22m.

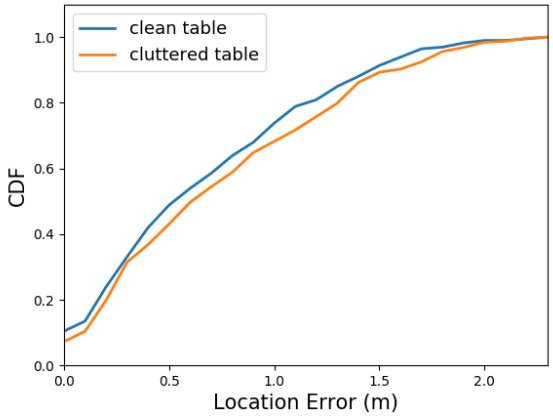


Figure 3.10 The influence of multipath effect to localization accuracy.

3.6.2.2 Influence of Multipath Effect

In this part, we explore the influence of multipath effect to the localization accuracy of SoundFlower. We create different degrees of multipath effect by adding objects around the microphone array as Figure 3.7c shows. As we can see from Figure 3.10, the accuracy of cluttered table is slightly lower than clean table. The median error is 0.5m if table is relatively clean and 0.6m if table is very cluttered.

3.6.2.3 Influence of Searching Step

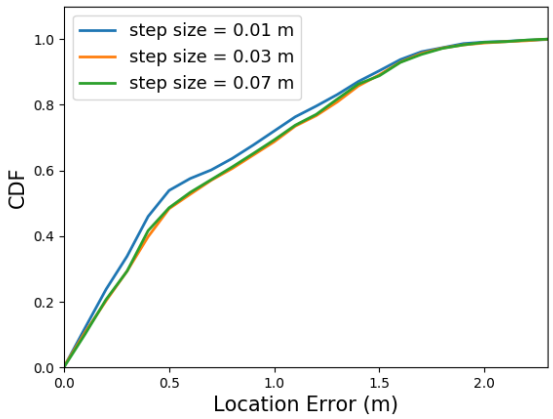


Figure 3.11 The influence of different searching steps to localization accuracy.

For experiments shown in Section 3.6.1, we use a searching step size of 0.1m to scan possible area. In this section, we further test our model on a step size of 0.01m, 0.03m and 0.07m. The result is shown in Figure 3.11. The increment of granularity does not lead to noteworthy accuracy increase. If the step size cannot cover the ground truth, usually SoundFlower will return the closest location to the ground truth.

3.6.3 Consumption Time

Figure 3.12 presents the consumption time for experiments of different scales. We run the model on a MacBook Pro (13-inch, Early 2015). The code is implemented in Python 3. For scanning area of 6m $\times$ 6m $\times$ 0.6m, 2D localization with a searching step size of 0.1m takes 3 - 4 seconds. 3D localization with step size of 0.1m takes around 5 seconds. 2D localization with a step size of 0.01m takes around 30 seconds. Due to the size of a normal person, we believe

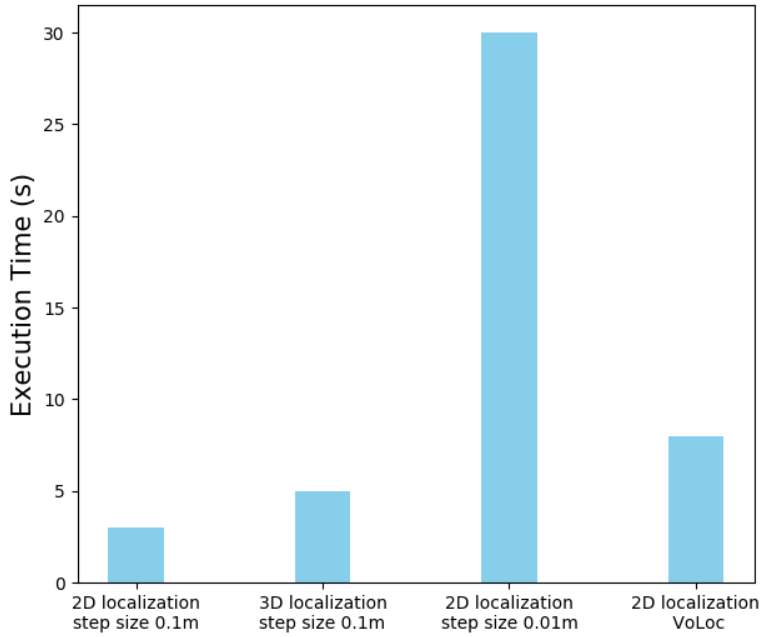


Figure 3.12 Consumption time.

a step size of 0.1m could achieve a good balance between consumption time and localization accuracy. According to [92], VoLoc takes 6 - 10 seconds to locate a person and hours to estimate the distance and orientation from the microphone array to the wall. We quote the time of VoLoc from the original paper in Figure 3.12 instead of measuring it by ourselves because when testing VoLoc, we found its consumption time is highly related to a preset parameter which narrows down the searching area after its initial estimation of direct-path AoA. If the parameter is small, the running takes seconds. But the algorithm may fail to find a location when the initial estimation of AoA deviates from the ground truth and the parameter filters out every possible location. If the parameter is large, VoLoc always finds an answer but the running takes minutes.

3.7 Discussion and Future Work

The main limitation of SoundFlower is, it cannot work when multiple people are speaking simultaneously. In practice, it is common when one person tries to wake up the voice assistant while other people are talking in the background. In this case, self-adjusting speech detection will return all frequency bins in which speeches exist, rather than the frequency bins in which the

wake-up command exists. To solve this issue, speech recognition method must be combined to recognize the wake-up command and truncate the time window of the wake-up command. We also need ideas like MUSIC[81] to explore the independence of multiple sound sources and separate the wake-up command from other speeches. We take these challenges as future work.

3.8 Conclusion

In this chapter, we present a robust system for voice assistants to obtain user location through user speech. After state-of-the-art work[92] shows the feasibility of such a model with the user height known and a wall next to the voice assistant, we extend the application scenario to 3D localization without the assumption of a second reflection path. We continue the idea of TDoA estimation between a pair of microphones, extract phase information from cross spectrum, design self-adjusting speech detection algorithm and unwrapping scheme to remove environmental noise and hardware noise, and apply robust regression to obtain TDoA against the disturbance of multi-path effect. We achieve similar localization accuracy to state-of-the-art work with less assumptions as well as less consumption time.

SoundFlower shows our efforts on overcoming diverse noises for sound source localization. Common sources of noises for IoT applications are imperfect hardware, the background noise and multipath effect. In this chapter, we show how to use statistical methods to compress the disturbance to system performance.

CHAPTER 4

PREVENTING UNAUTHORIZED SPEECH RECORDINGS WITH SUPPORT FOR SELECTIVE UNSCRAMBLING

Human beings have long used acoustic signals to exchange information with each other. Human beings now use acoustic signals, which is speech, to exchange information with ubiquitous smart devices such as smartphones, smartwatches, and digital assistants that are equipped with embedded microphones. While these speech detection and recognition capabilities make possible many convenient features, they also introduce many privacy risks such as secret, unauthorized recordings of our private speech [113, 114] that can have real world consequences. For example, the Ukrainian prime minister offered his resignation after an unauthorized recording was leaked [115].

Manufacturers claim that they are trying their best to protect users' privacy, but there is no effective and user-friendly technical anti-recording solution available despite the fact that anti-recording is not a new problem. One existing anti-recording solution is to talk near a white noise source, *e.g.* near an FM radio tuned to unused frequencies, so that the conversation cannot be clearly recorded. This approach is not user-friendly because the people having the conversation must put up with the white noise that interferes with their normal communication. A similar solution [116] emits high frequency noise near the upper bound of human sensitivity; most people do not notice the interference, but pets and infants may notice it [117], so this solution is not environment-friendly. Electromagnetic interference was an effective anti-recording solution [118] in the past, but modern microphones are immune to electromagnetic interference. Moreover, all of these traditional anti-recording approaches cannot allow authorized devices to clearly record conversations.

Any effective anti-recording solution must provide the following three key properties: (1) normal human conversation should be unaffected by the anti-recording solution meaning the anti-recording solution should not change what humans hear while having a conversation; (2) unauthorized devices should not be able to make a clear recording of any conversation protected by the anti-recording solution; (3) authorized devices should be able to make a clear recording of any

conversation protected by the anti-recording solution.

One potential solution that can satisfy all three properties is to generate multiple ultrasonic frequency sound waves because of the following two properties of ultrasonic waves. First, humans cannot hear ultrasonic sound waves. Second, commercial off-the-shelf (COTS) microphones exhibit nonlinear effects, which means that when these microphones receive multiple ultrasonic sound waves, they generate low-frequency sound waves that can be heard by humans and thus interfere with the clarity of recordings made with those microphones [17, 117, 119–123]. There are three main challenges that must be overcome in order to develop an ultrasonic anti-recording solutions that satisfies the three key properties:

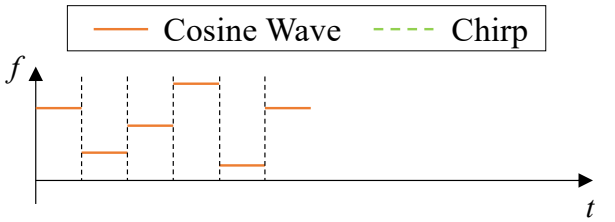
• First, any ultrasonic anti-recording solution must defend against potential attacks such as using Short-time Fourier transform (STFT) to analyze unauthorized recordings and using filters to cancel out the low-frequency sound waves that interfere with recording clarity.

• Second, ultrasound travels along a straight line [124], which means a single ultrasonic wave generator can only interfere with recording devices within a limited range of angles from the generator. In practice, it is difficult to design an ultrasonic anti-recording solution that can neutralize all recording devices within a large coverage area.

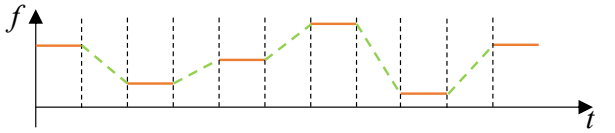
• Finally, the performance of authorized devices could be affected by the ringing effect due to electronic behaviors. Such ringing impulses are hard to be canceled and may remain in authorized recordings, severely downgrading the quality of the descrambled recordings.

In this paper, we present Patronus, an ultrasonic anti-recording system that satisfies the three key properties. Patronus has two key components: the *scramble* that is the pseudo-noise generated at all microphones, and *descrambling* that is the process to remove the scramble for authorized devices. We form the scramble by randomly picking frequencies from the human voice frequency band and then shifting them to the ultrasonic band. To thwart STFT attacks, we further fine-tune the period of the scramble so that it cannot be easily analyzed and canceled. We add a reflection layer with a curved surface to create a reflected ultrasonic wave that can cover a wider area. Finally, to mitigate ringing effects, *i.e.* sudden hardware impulses due to discrete frequency changes of

current waves, we use chirps to smooth the frequency changing components of the scramble, as shown in Figure 4.1.



(a) Discrete frequency scramble components.



(b) Continuous frequency changing scramble with chirps.

Figure 4.1 Using chirps to smooth the frequency changing components of the scramble.

Patronus lets authorized devices clearly record audio conversations by sending them the scramble pattern. With scramble pattern, the authorized device applies the Normalized Least-Mean-Square (NLMS) adaptive filter [125] to cancel the scramble and thus produce a clear audio recording of the conversation.

We implement a prototype of Patronus and conduct comprehensive experiments to evaluate its performance. We use the Perceptual Evaluation of Speech Quality (PESQ) [126], the Speech Recognition Vocabulary Accuracy (SRVA, see Section 4.5), and speech recognition error rates (1 - SRVA) to evaluate the performance of Patronus. Our results show that only 19.7% of the words protected by Patronus' scramble can be recognized by unauthorized devices. Furthermore, authorized recordings have 1.6x higher PESQ and, on average, 50% lower speech recognition error rates than unauthorized recordings.

In this paper, we provide several unique technical contributions when compared to existing works. First, to the best of our knowledge, Patronus is the first system to leverage the nonlinear effect of COTS microphones to prevent unauthorized recordings while allowing authorized recordings. Second, we perform a thorough study of the nonlinear effects of ultrasound frequencies including the effects of higher orders whereas recent works[17, 119, 120, 127] only consider

the order up to 2. This is critical for descrambling when the signal components with order higher than 2 will likely lie in the human voice frequency band, which means simply cutting off the high frequency components will result in message loss. Instead, our descrambling solution carefully removes these higher order frequencies using an NLMS filter. Third, we mitigate ringing effects by connecting scramble segments with chirps. This simplifies learning the coefficients of impulse response in existing work [17], especially when we deploy multiple ultrasonic transducers in a large space.

In general, our contributions are as follows:

• We propose a novel ultrasound modulation approach to provide privacy protection against unauthorized recordings that does not disturb normal conversation.

• We do a thorough study around the nonlinear effect of ultrasound on commercial microphones and propose an optimized configuration to generate the scramble.

• To overcome the fact that ultrasound travels in a straight line, we design a low cost reflection layer to effectively enlarge the coverage area of Patronus in a cost-effective way.

• We present Speech Recognition Vocabulary Accuracy, a new metric to measure the recording quality. Our experimental results with both PESQ and SRVA show that Patronus effectively prevents unauthorized devices from making secret recordings.

The organization of the rest of this paper is as follows. Section 4.1 introduces related work. Section 4.2 introduces the nonlinear effect of common microphones, which we analyze more thoroughly than existing works. Section 4.3 presents the design of Patronus. Section 4.4 presents the prototype implementation of Patronus. Section 4.5 presents our evaluation results of Patronus. Section 4.6 discusses the limitations of Patronus and future work, and Section 4.7 concludes this work.

4.1 Related Works

4.1.1 Nonlinear Effect of Microphones

There has been a lot of research into the nonlinear effect of microphones. For many years, the development of ultrasonic systems on smartphones was restricted due to being limited to a roughly

4 kHz range of frequencies between the high end of human hearing to the cutoff frequency of typical microphones. Furthermore, some infants and pets can actually perceive frequencies within this small band. Roy *et al.* [17] performed detailed research on the nonlinear effects of microphones to break through these limitations and expand the working frequency band for ultrasonic systems on smartphones. DolphinAttack [120] leverages the nonlinear effect to generate audio commands that are inaudible to humans. After being recorded by the microphone, the input ultrasonic signals would generate a shadow signal that could be recognized by VCS. Therefore, attackers can perform unauthorized commands without being discovered. SurfingAttack [123] uses oscillation of a surface such as a table to transmit inaudible commands. With this modality, attackers can deploy their speakers in hidden spots such as the back of the surface being used to transmit the secret commands. LipRead [119] extends the attack range by leveraging characteristics of human hearing. It also puts forward a model to filter out such commands generated by the nonlinear effect. Metamorph [121] injects inaudible commands into human-made commands to achieve unauthorized actions. AIC [127] presents a mechanism that fundamentally cancels inaudible commands against VCS, which we will discuss as an attack model in Section 4.3.2. NAuth [122] uses the nonlinear effect to authenticate devices. Unlike most of these methods, Patronus aims to preserve privacy by adding a removable scramble generated by ultrasonic signals to the recorded human speech. From a technical perspective, Patronus is unique in that it takes into account third and higher order terms from the nonlinear effect. Our experiments show those high order terms can affect recordings whereas most existing methods (e.g., AIC) only consider the second order term and assume the higher order sub-band of the microphone is clean.

4.1.2 Dual Channel Applications

Some applications leverage the difference between humans and devices. For example, human eyes and devices have different perceptions of flicker frequency. Technologies exist that use this phenomena to communicate between the screen and the camera without affecting human vision [128–131]. Likewise, some technologies modulate acoustic signals in ways that no human can detect to communicate between devices [132, 133].

The difference between the sensitivity of humans and devices is also used in privacy protection. Kaleido [134] protects a movie's copyright by adding a flashing distractor with very high frequency into movie frames that cannot be seen by human eyes. If such a protected movie is subsequently recorded by an unauthorized camera equipped with a rolling shutter, the distractor will be visible on the unauthorized recording because of its high sample rates making the pirated recording a low quality recording. LiShield [135] also uses the Rolling Shutter effect to reduce the quality of photos. Lights with different colors are set to flash in alternating high frequencies that provide normal lighting because human eyes cannot sense the flashing. However, cameras are influenced because the Rolling Shutter samples column by column meaning unexpected color stripes will appear on the photo. In the end, it prevents unauthorized cameras from taking photos. Although Patronus has a similar motivation to prevent unauthorized recordings, Patronus is different from the two papers as it targets acoustics rather than visuals.

4.2 Nonlinear Behavior of Common Microphones

In this section, we provide a brief primer about nonlinearity of common microphones; a more comprehensive introduction can be found in recent papers [17, 119]. Ideally, COTS microphones are linear systems. Given the input signal $s(t)$, the output signal $y(t)$ is expected to be linear combinations of the input signal, i.e., $y(t) = A_1 s(t)$ where A_1 is the complex gain quantifying the change of the phase and amplitude. Due to the physical properties of materials and variations in manufacturing, the components of a common microphone, such as the diaphragm and the pre-amplifier, are imperfect and typically do not constitute a linear system. As a result, COTS microphones, which are widely equipped on smartphones and smartwatches, typically exhibit nonlinear behavior. Specifically, the output signal $y(t)$ is under nonlinear effect, where $y(t) = A_1 s(t) + A_2 s^2(t) + A_3 s^3(t) + \cdots$, and the power gains of each component satisfy $|A_m| > |A_n| (m < n)$.

When the input signals are composed of two different ultrasonic frequencies, the output from a nonlinear microphone would contain several new shadow sounds with frequencies that are a linear combination of the two input frequencies. Assuming that the input signal is $s(t) = \cos(2\pi f_1 t) + \cos(2\pi f_2 t)$ where f_1 and f_2 are the ultrasonic frequencies, the output signal would be $y(t) =$

$\sum_{i=1}^{+\infty} A_i s^i(t)$. Without loss of generality, we assume $f_1 > f_2$ in the following discussion. For each component $A_i s^i(t)$,

$$\begin{aligned} s^i(t) &= (\cos(2\pi f_1 t) + \cos(2\pi f_2 t))^i \\ &= \mu + \sum_{j=1}^{i} [\alpha_j \cos(2\pi j f_1 t) + \beta_j \cos(2\pi j f_2 t)] \\ &+ \sum_{j=1}^{i-1} [\lambda_j \cos(2\pi (j f_1 - (i-j) f_2) t) + \gamma_j cos(2\pi (j f_1 + (i-j) f_2) t)], \end{aligned}$$

where α_j, β_j, λ_j and γ are coefficients of the polynomial expansion, and μ is the consequent constant.

After the pre-amplifer, the signals would pass through an embedded low-pass filter whose cut-off frequency is usually 24 kHz. Since f_1 and f_2 are both ultrasonic frequencies, jf_1 and jf_2 are all ultrasonic frequencies. However, if $i = 2j$, $jf_1 - (i-j)f_2 = j(f_1 - f_2)$ may be a non-ultrasonic frequency when j is small enough. Therefore, when the input signal is $s(t) = \cos(2\pi f_1 t) + \cos(2\pi f_2 t)$, new audible cosine waves $\cos(2\pi j(f_1 - f_2)t)$ appear, where $j = 1, 2, \ldots, k$, $k \leq i$, and $k(f_1 - f_2) \leq$ 24 kHz.

Existing works like BackDoor[17] and DolphinAttack[120] make use of $A_2 s^2(t)$ but ignore higher-order components; they essentially assume that for $i > 2$, $|A_i|$ is relatively small and has little effect on the output signal. However, in our experiments, we find that more high-order components should be taken into consideration as they do affect the output signal.

4.3 Design

4.3.1 Overview

As shown in Figure 4.2, there are three parties involved in Patronus: the Scramble Transmitter, authorized devices with descramble receivers, and unauthorized devices.

The Scramble Transmitter sends a series of scramble signals with randomly varying frequencies. To ensure that unauthorized voice recordings will be affected, the frequencies of the recorded scrambles should be located in the human voice band. Therefore, we use the Scramble Generator to generate random frequencies in the target range, store them as a secret key, and send them to

the Descramble Receivers through Wi-Fi, Bluetooth, or other media. The Scramble Generator then generates cosine wave segments according to these frequencies. The generated segments are then sent to the Frequency Shifter and their frequencies will be increased by f_0, which is an ultrasonic frequency. To ensure the scramble signal is picked up by microphones of unauthorized devices because of the nonlinear effect, we design a Constant Cosine Wave Generator to transmit a cosine wave with a constant ultrasonic frequency of f_0.

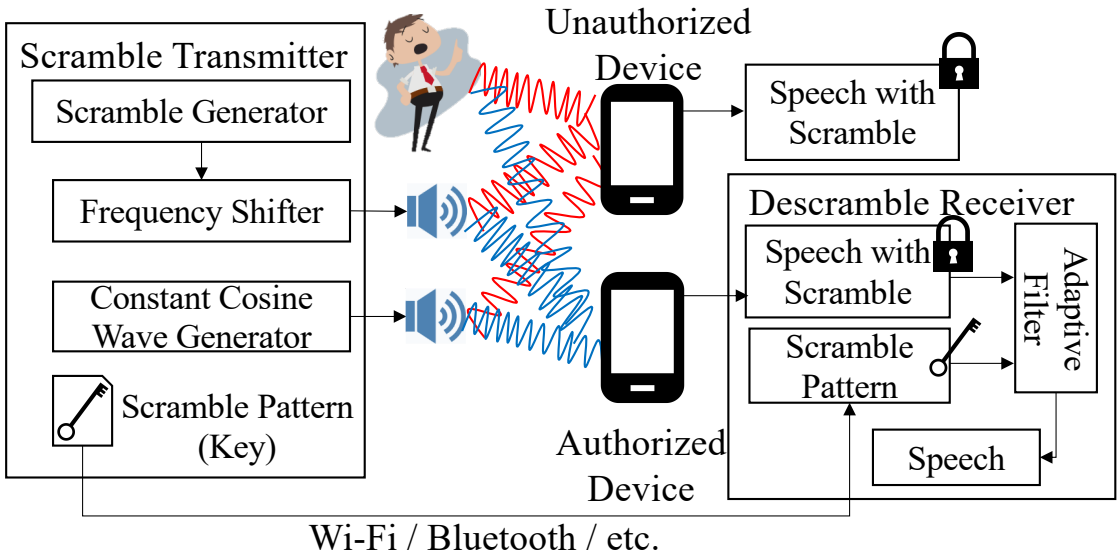


Figure 4.2 System overview.

During human talking protected by Patronus, the actual human conversation plus two ultrasonic signals will arrive essentially simultaneously at recorders (both authorized and unauthorized) and human ears. Human ears will not detect the ultrasonic signals and thus receive the human conversation with no additional noise. As discussed in Section 4.2, the two ultrasonic signals will generate a shadow audible signal that will be included in any recording made by a COTS microphone due to nonlinear effects. This applies to both authorized and unauthorized devices. Authorized devices, which receive a secret key from the Scrambling Transmitter, can generate the scramble waveform. They can then feed the scramble waveform along with the scrambled recording into an adaptive filter to extract clear speech from the scrambled speech. The details of descrambling will be discussed in Section 4.3.5.

We must overcome three challenges in order to design Patronus. First, we must design a system

whose working area is as large as possible. This is difficult because a sound wave of high frequency typically travels along a straight line meaning a straightforward implementation of ultrasonic generators will only cover a small area defined by a limited range of angles. Second, there is a trade-off between a shorter and a longer period of scramble frequencies. As the period increases, the system is more vulnerable to unauthorized recordings using STFT attacks. As the period decreases, the difficulty of descrambling increases. Our goal is to maximize the information recovered by authorized devices over unauthorized ones without exposing the scramble pattern to STFT. These details are discussed in Section 4.3.3.4. Third, when frequency changes frequently, a severe ringing effect (Section 4.3.3) occurs in the scrambled recording, which affects even the recordings made by authorized devices after descrambling. We use chirps to connect each frequency component of the scramble to eliminate the sudden change of the input to ultrasonic speakers, hence minimizing the ringing effect and enhancing the quality of the recovered speech by authorized devices.

4.3.2 Attack Model

Based on common acoustic processing technologies and known properties of nonlinearity effects, we consider the following types of attacks:

4.3.2.1 Short-Time Fourier Transform (STFT)

One natural way for an unauthorized device to try to extract a useful recording from its scrambled recording is to analyze the scrambled recording with STFT and filter out suspicious frequencies. We address this attack model by changing the scramble frequency according to a finely-tuned period model, making it impossible for the attacker to obtain each exact scramble frequency along with its start and end time. Detailed analysis is provided in Section 4.3.3.4. Even with the correct scramble frequencies available, bandpass filters will not work because the scramble frequencies are selected from the human voice band. The frequencies from chirps and those from human speaking are mixed together. To prove Patronus can defeat this attack model, we simulate the attack scenario when (1) the attacker is aware that our scramble pattern is varying continuous waves smoothed by chirps (2) the attacker calculates approximate scramble frequencies with STFT (3) the attacker applies NLMS adaptive filter (Section 4.3.5.4) to remove the scramble with the approx-

imate scramble frequencies they obtained from STFT. Our simulated attack experiments, provided in Section 4.5.8, show that this attack will fail because the approximate scramble frequencies are not accurate enough.

4.3.2.2 Extra Ultrasonic Transmitter Attack

After DolphinAttack[120] proposes to inject malicious commands into ultrasound, AIC [127] adds three more ultrasonic transmitters to cancel the malicious commands and protect Voice Control Systems (VCS). AIC assumes the legitimate as well as malicious commands are within the lower sub-band of the microphone sensible frequency band. Their added ultrasonic transmitters project only the malicious commands onto the higher sub-band, which can be used to filter the malicious commands in the low sub-band. With a fast changing of scramble frequencies, we can cover the whole frequency band, and make sure no clean band is left for attackers.

4.3.2.3 Wi-Fi/Bluetooth Snifing

Attackers can sniffer the Wi-Fi or Bluetooth channel to get the scramble pattern transmitted from the Scramble Transmitter to the authorized device. However, there are many cryptographic approaches to prevent attackers from sniffing channels. For example, we can encrypt the scramble pattern by AES-CTR using a pre-shared key and then directly send it to authorized devices.

4.3.2.4 Physical Attacking

There are also some physical attack models. First, attackers can place an obstacle before the Scramble Transmitter. However, attackers cannot do it secretly and nobody would like to do so. Second, attackers may just wrap a cover on their microphones. However, the cover itself may defeat the attackers objective of making a good recording. Although Patronus cannot perfectly handle such attack models, it enhances the difficulty of making an unauthorized recording. Finally, attackers may conduct experiments to discover where Patronus fails. This can be fixed by enlarging the working area through some methods that we will discuss later.

4.3.3 Ultrasonic Scramble Modulation

Two ultrasonic signals will be superimposed at the recorders to create the desired low-frequency component. In the design of the scramble using ultrasonic signals, we mainly consider the following issues:

4.3.3.1 Range of Frequency

The first issue is how to make it hard to cancel out the scramble without the key. Basically, the range of human speech frequency is from 85 Hz to 255 Hz [136, 137]. If the scramble consists of multiple random frequencies from this range, it is hard for attackers to cancel the scramble using linear filters. The application of a linear filter, *e.g.* highpass filter, will not only cancel the scramble, it will also change the original human speech. To ensure the scramble covers all human speech frequencies in practice, we modulate the scramble with a wider frequency band than $[85, 255]$ Hz.

4.3.3.2 Random Frequencies

If we always use specific frequencies to generate the scramble, attackers could analyze the frequency spectrum of their recordings to infer the scramble frequencies; with those, they could then recover the original audio signals.

To address this issue, we choose scramble frequencies randomly. We also periodically change the scramble frequencies over time. The sequence of scramble frequencies can be thought of as a one-time pad key. Without the sequence, it would be difficult for attackers to remove the scramble.

4.3.3.3 Ringing Effect

Frequent changing of the scramble frequencies produces a ringing effect [17] that makes it challenging for authorized devices to produce a high-quality descrambled recording. Specifically, the ringing effects incur heavy-tailed impulse responses that will remain in descrambled recordings as shown in Figure 4.3 (a) and (b). Since the ringing effect occurs when the input changes suddenly, we use a chirp signal to connect two adjacent segments with different frequencies in the scramble to smooth such a sudden change. Specifically, when the scramble changes from frequency A to frequency B, we add a transition signal that starts at frequency A and moves linearly to end with

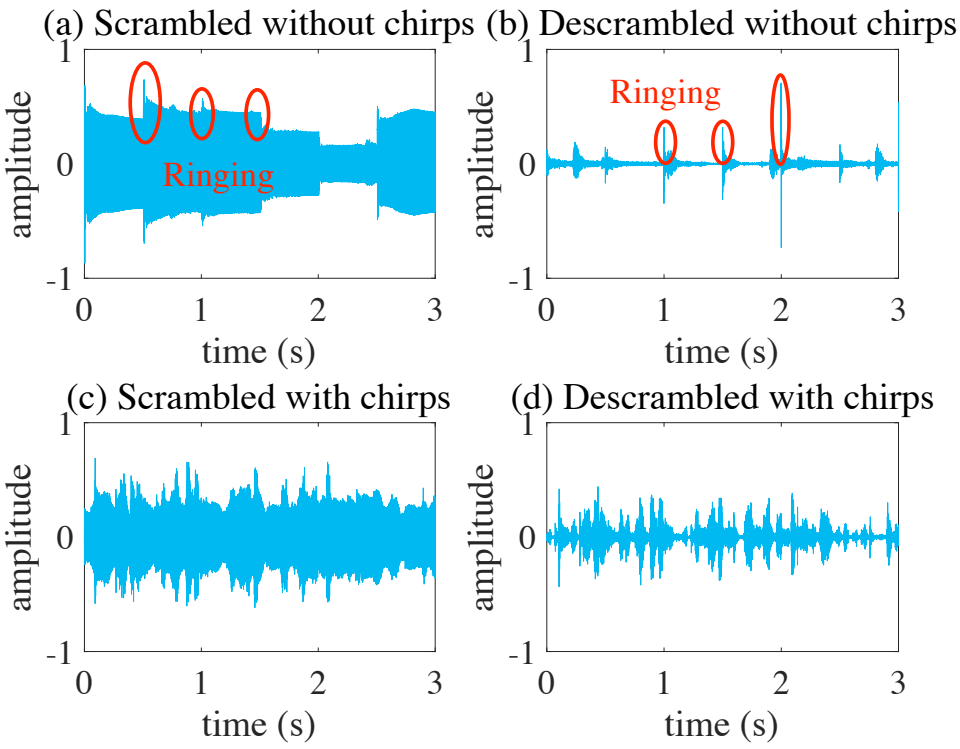


Figure 4.3 Illustration of how linear chirps mitigate the ringing effect.

frequency B.

The impulse incurred by ringing effects can have a very high amplitude or power. It will suppress other signals due to the microphone Passive Gain Suppression [17]. Figure 4.3 confirms that the ringing effect is mitigated by chirps. Figure 4.3 (a) shows a scrambled recording with no chirp, the resulting descrambled recording in Figure 4.3 (b) has many areas where most of the signal is suppressed. In contrast, Figure 4.3 (c) exhibits a scrambled recording with chirp signals, the resulting descrambled recording in Figure 4.3 (d) does not have the peak signals corresponding to the ringing effect and the rest of the signal is not suppressed.

4.3.3.4 Duration of each frequency

The next challenge is choosing the proper duration for each frequency in the sequence of scramble frequencies. Intuitively, if we give each frequency a long duration, unauthorized devices could easily split the record into multiple segments where each segment is only protected by a constant frequency scramble. They could then apply simple techniques such as using a linear bandpass filter to the scrambled recording to extract a clear speech recording.

More generally, there are two competing issues in choosing the duration of each scramble frequency, namely, defending against STFT attacks that are discussed in Section 4.3.2.1, and ensuring that authorized devices can obtain high-quality descrambled recordings. We first consider defending against STFT attacks. An STFT attack can successfully remove the scramble waveform if it can both accurately infer the frequencies and time periods for each scramble frequency in the sequence of frequencies. When the window length is n, the frequency resolution would be $\Delta f = \frac{f_s}{n} = \frac{f_s}{f_s \times t} = \frac{1}{t}$ where f_s is the sampling rate and t is the duration of the window. Taking 0.1s as an example, the offset of STFT can reach 10Hz. If the attacker tries to improve the frequency resolution by lengthening the window, the accuracy of the estimated time periods for the given scramble frequency will diminish. If the scramble frequency duration is long, scramble frequency will exhibit fewer changes within any given window, thus STFT attacks can use longer windows to accurately estimate the frequency with exact estimates of the frequency time period. Therefore, to thwart STFT attacks, we should make the frequency duration as short as possible. However, a too-short duration may misshape the scrambled recording due to imperfect hardware. A typical microphone and speaker use a diaphragm to sense and generate the vibration; this diaphragm moves continuously and can not change its position instantaneously. Circuit latency also makes it hard for the system to respond to frequent and instant changes. As a result, the scrambled waveform would be slightly distorted. This means the NLMS adaptive filter at authorized devices may not correctly descramble the scrambled waveform because it does not expect the distortion caused by frequent frequency changes. Therefore, the frequency duration cannot be too short. In summary, to balance these competing concerns, we must find a frequency duration that maximizes the information recovered by authorized devices compared to the information recovered by unauthorized devices. To identify a good frequency duration, we measure the descrambling performance with different frequency durations in Section 4.5.8.

4.3.3.5 Key Construction

We have two choices to construct the key for granting the privilege of recording the audio to authorized devices. One is directly using the scramble waveform generated by the Scramble Gen-

erator as the key. After getting the scramble waveform, authorized devices remove the scramble from the recorded audio. But there are some issues we need to consider. First, the sampling rate of authorized devices may vary from one to another. It means that in terms of the digital signal, devices having different sampling rates will get different presentations of the same scramble waveform. To grant the privilege to devices, the Scramble Transmitter should generate different digital scramble waveforms according to different sampling rates of authorized devices. This results in high computational overheads. Second, in addition to different sampling rates from different authorized devices, the sampling rates of the Scramble Generator and an authorized device may be also different. As a result, the scramble that the speaker emitted might have a different presentation of the recorded waveform.

In Patronus, we choose another way to construct the key. We select the frequency sequence used to generate the scramble as the key. After receiving the frequency sequence, an authorized device can reconstruct the scramble waveform with their sampling rates, which we discuss in more detail later. After that, an authorized device can use the reconstructed scramble waveform to remove the scramble from the recording and get the clear speech.

With the discussion above, we formally describe the scramble generation. We set one speaker to transmit an ultrasonic continuous wave $S_1(t) = \cos(2\pi f_0 t)$, while the other speaker transmits continuous waves linked by chirps $S_2(t) = \cos(2\pi f(t)t)$, where

$$f(t) = \begin{cases} f_i, & (2i-2)\Delta t \leq t < (2i-1)\Delta t, \\ f_i + \frac{f_{i+1}-f_i}{\Delta t}t, & (2i-1)\Delta t \leq t < 2i\Delta t, \end{cases} \tag{4.1}$$

and $f_i (i = 1, \ldots, n)$ are randomly generated constant frequencies. Δt is the duration of a single sine wave or a chirp. The induced low-frequency noise will be

$$R(t) = \cos(2\pi(f(t) - f_0)t). \tag{4.2}$$

To ensure $R(t)$ covers human voice, $f_i (i = 1, \ldots, n)$ are sampled from $[f_{low} + f_0, f_{high} + f_0]$ where $[f_{low}, f_{high}]$ covers the human voice band.

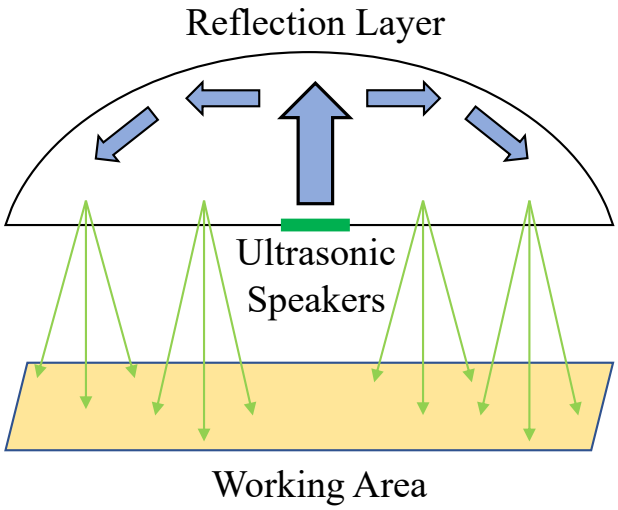


Figure 4.4 Enlarge working area with reflection.

4.3.4 Enlarge Scramble Working Area

The scramble signal is generated by two ultrasonic signals, which incurs another issue as the ultrasonic wave typically propagates in a straight line. In other words, if you want to prevent a certain device from recording, the ultrasonic speaker should be pointed directly towards that device. This results in a limited coverage area for ultrasonic anti-recording solutions.

Inspired by lamps that often use a bow-shaped cover to reflect the light beam in many directions, we build a reflection layer that reflects the ultrasonic wave in many directions. As Figure 4.4 shows, we put ultrasonic speakers near the center of the reflection layer and place the devices (authorized and unauthorized) in the working area. When the ultrasonic wave hits the reflection layer, it gets reflected in many directions leading to a much larger cover area.

4.3.5 Grant Recording Privilege

The goal of Patronus is not only to block unauthorized devices from recording audio, but also to provide authorized devices with a mechanism to recover speech. Patronus achieves this by creating a way for authorized devices to remove the scramble from the scrambled recording. Specifically, Patronus grants the clear recording privilege to authorized devices using the following steps.

4.3.5.1 Key Transmission

The Descramble Receiver needs the waveform of the scramble generated by the Scramble Generator before it can remove the scramble. Intuitively, if it had the pure scramble waveform, it could remove the scramble from the recorded audio by subtracting the scramble waveform from the recorded audio waveform. The scramble waveform here acts as the key for deciphering the recorded audio. We send the key through non-acoustic channels such as Wi-Fi or Bluetooth with cryptographic protection to prevent eavesdroppers from getting the key. Additionally, because of the randomness of scramble frequencies, they cannot get a usable scramble waveform by listening to the acoustic channel. Instead, they can get either the combination of interfered speech with scramble, or get the scramble without speech but independent of the successive scramble waveform.

4.3.5.2 Scramble Reconstruction

As discussed in Section 4.3.3, the Scramble Transmitter sends the random frequency sequence instead of the scramble waveform to authorized devices as the key. Patronus needs to use these frequencies to reconstruct the scramble waveform before removing the scramble. An authorized device uses Equation (4.2) and its recording sampling rate to generate the scramble waveform.

4.3.5.3 Synchronization

We need to synchronize the reconstructed scramble with the recorded scramble before removing it from recordings. Specifically, we choose a segment from the reconstructed scramble as the template, *e.g.* the beginning segment. Then we use cross-correlation to find the segment that is the most similar to the template. We then synchronize the recorded scramble and the reconstructed scramble by aligning the two segments.

4.3.5.4 Adaptive Filtering

Now we have the waveform of the scramble. The next task is to remove the scramble from the recorded audio with the known waveform of the scramble. Practically, we cannot directly subtract the scramble from the recorded audio because when the sound propagates through the air, it will be

distorted due to reflection and attenuation. We use adaptive filter to remove the waveform-known scramble.

Adaptive filter is widely used in Active Noise Cancellation (ANC) headsets. Technically, there is a reference microphone outside the headset. The reference microphone captures the noise, and the digital signal processor (DSP) generates the anti-noise wave according to the captured noise. When the noise wave and the anti-noise wave arrive at the ear, they eliminate each other. In Patronus, we denote the speech as x_1. It propagates through the acoustic channel h_1, arrives at the authorized device and becomes $h_1 * x_1$, where the operator $*$ denotes the convolution operation. Additionally, we denote the scramble waveform that is generated by non-linear effects and recorded by the authorized device as x_2. It propagates through another channel h_2, arrives at the authorized device and becomes $h_2 * x_2$. Therefore, the audio recorded by the authorized device is

$$y = h_1 * x_1 + h_2 * x_2. \tag{4.3}$$

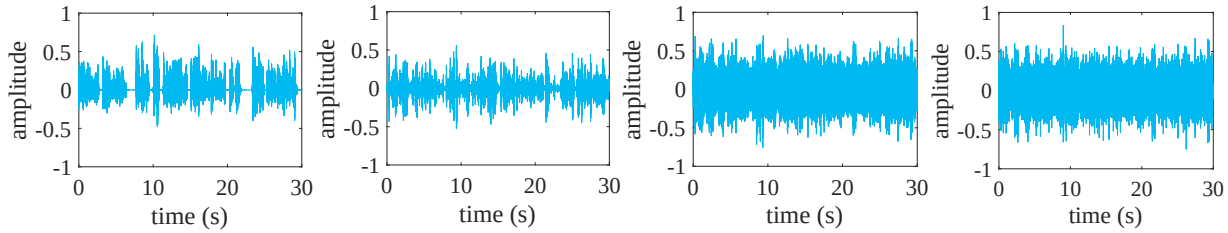
Similar to ANC headsets, here we see the scramble x_2 as the noise in ANC headsets. Different from ANC headsets, the noise here is generated from the key as we discussed in Section 4.3.5.2. Therefore, we can use the Normalized Least-Mean-Square (NLMS) Adaptive Filter [125] to remove the scramble. Formally, we are trying to find a channel vector h_2' to solve the optimization problem

$$\min E[(y - h_2' * x_2)^2]. \tag{4.4}$$

When the expectation in Equation (4.4) is minimized, $h_2 \approx h_2'$. Therefore, $h_1 * x_1 \approx y - h_2' * x_2$, and it can be regarded as the speech without the scramble. Stochastic gradient descent is usually adopted to solve the optimization problem defined by Equation (4.4), but it is hard to derive the gradient of the expectation. Researchers thus use $(y - h_2' * x_2)^2$ instead of the expectation to solve the problem. In this way, the noise gets canceled [138].

Following this design, we can develop a mechanism that prevents unauthorized recording while supporting authorized recording. The mechanism also prevents attackers from descrambling without authorization. Figure 4.5 gives an example. A piece of VOA news audio is used as the original

record, the attack result has severe scramble effects just like the unauthorized record, but the authorized record removes almost all scrambles.



(a) Original Waveform (b) Authorized Waveform (c) Unauthorized Waveform (d) Descrambled by STFT Attack

Figure 4.5 Illustration of original waveform, authorized waveform, unauthorized waveform, and descrambled waveform by STFT attack.

4.4 Implementation

This section discusses the details of the implementation of Patronus, which contains two parts, the Scramble Transmitter and the Descramble Receiver for authorized devices. We use an ordinary smartphone with its built-in audio recorder as the Unauthorized Device or Authorized Device.

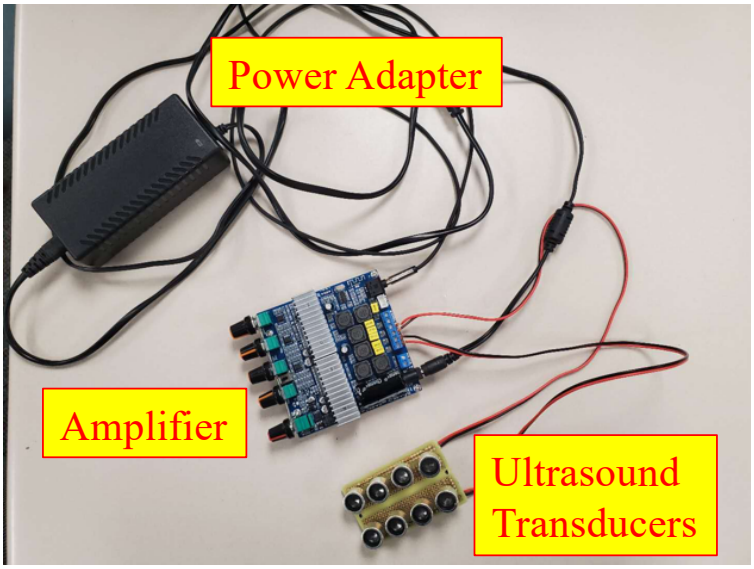


Figure 4.6 Implementation of scramble transmitter.

4.4.1 Scramble Transmitter

4.4.1.1 Hardware Implementation

As Figure 4.6 shows, we use eight TCT40-16R/T 16 mm ultrasonic transducers. Half of them play the frequency-shifted scramble and they are connected in parallel. The other half play the fixed-frequency cosine wave and are connected in parallel as well. We utilize an AOSHIKE DC12V-24V 2.1 Channel TPA3116 Subwoofer Amplifier Board to enhance the power of output ultrasonic signals. The two waveforms are played through a stereo channel. The frequency-shifted scramble uses the left channel, and the constant-frequency cosine wave uses the right channel.

As we have discussed in Section 4.3.4, we use a reflection layer to enlarge the working area. In this prototype, we use an iron wok as the reflection layer. The opening diameter of the iron wok is 30 cm, and the depth is 10 cm. As shown in Figure 4.7, the ultrasonic transducers are placed towards the center of the iron wok.

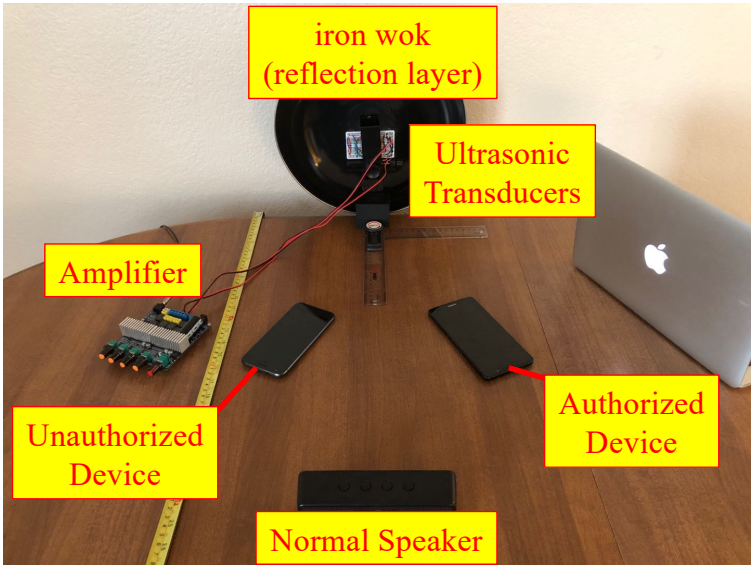


Figure 4.7 Prototype of Patronus.

4.4.1.2 Format of Key

As we have mentioned in Section 4.3, Patronus uses the frequency sequence as the key. This key must include the duration of each frequency in addition to the frequency itself in order for the

Descramble Receiver to generate the scramble waveform. Thus, our key file includes the frequency sequence plus the sample rate of the Scramble Transmitter and the number of samples of each frequency.

4.4.2 Descramble Receiver for Authorized Devices

We use an ordinary smartphone as an authorized device. The authorized device receives the key from the Scramble Transmitter. After the audio is recorded, the smartphone reconstructs the scramble waveform with the given key and leverages NLMS Adaptive filter to cancel the scramble. Formally, it takes the following steps:

4.4.2.1 Reconstruct Scramble Waveform

As we mentioned, in addition to the frequency sequence, the received key also contains the sampling rate of the Scramble Transmitter, which is denoted by f_{st}, as well as the number of samples of each frequency n_t. With the known sampling rate of the authorized device f_{sr}, the number of its recovered samples for each scramble frequency component can be calculated through the equation

$$n_r = \frac{f_{sr} n_t}{f_{st}}, \tag{4.5}$$

After getting n_r, the authorized device uses the same process as the Scramble Transmitter to generate the scramble, *i.e.* generating the discrete cosine signal with the frequency f_i and f_{i+1}, and connecting them by a chirp signal with start frequency f_i and end frequency f_{i+1}, where f_i and f_{i+1} are from the frequency sequence in the key.

4.4.2.2 Normalized Least-Mean-Square (NLMS) Adaptive Filter

After reconstructing the scramble waveform, we can use the Normalized Least-Mean-Square Adaptive Filter to cancel the scramble from the scrambled record. Specifically, we put the scrambled record rec_s and the scramble waveform s into the NLMS Adaptive Filter to get the descrambled waveform e by removing s from rec_s. According to the discussion in Section 4.2, the scramble wave is not only generated by frequencies in the given frequency sequence but also generated by high-order frequencies that are multiples of the target frequencies. Therefore, after getting e from

the NLMS Adaptive filter, we still need to iteratively remove the multiples of the frequency sequence scramble by NLMS Adaptive filter. It means that we iteratively put e and the scramble waveform generated by k-times multiple of the frequency sequence into NLMS Adaptive Filter, where $k = 2, 3, 4, 5, 6$ in our prototype.

In summary, the procedure of authorized devices for removing the scramble from the record is shown in Algorithm 4.1.

Algorithm 4.1 Remove Scramble from the record.

Input: rec_s, f_{sr}, f_{st}, n_t,
the frequency sequence $f[1..n]$
Output: Speech Record without Scramble e
1: $n_r \leftarrow f_{sr} n_t / f_{st}$
2: $e \leftarrow rec_s$
3: **for** $k = 1$ to 6 **do**
4: $s \leftarrow$ ScrambleGenerator($k \times f[1..n]$, n_r).
5: $e \leftarrow$ NLMS-Adaptive-Filter(e, s)
6: **end for**
7: e

The NLMS-Adaptive-Filter can be found in many open-source libraries, *e.g.* MATLAB, Python, *etc.*. Due to the selective frequency response of different smart devices, each model has its own parameter setting. In the implementation, we choose 500 as the number of taps and 0.005 as the step size for an iPhone, 100 as the number of taps and 0.003 as the step size for a Pixel, and 300 as the number of taps and 0.005 as the step size for a Galaxy S9.

4.4.3 Simulated STFT Attacker

We also simulate an STFT attacker to verify whether or not Patronus can prevent such an attack. Specifically, as discussed in Section 4.3.2.1, we apply STFT to the scrambled recording using the MATLAB function *stft* to infer its frequency sequence. We then feed the frequency sequence to an NLMS adaptive filter to get the descrambled recording. Experiment results are shown in Section 4.5.8. Here, we illustrate an example, which contains the original waveform, authorized waveform, unauthorized waveform and the waveform descrambled by STFT, in Figure 4.5. As illustrated by the figure, we observe that the authorized waveform is similar to the original wave-

form, the unauthorized waveform is different from the original one, and the unauthorized waveform is similar to the waveform descrambled by STFT attack. Therefore, our prototype proves that Patronus can block the unauthorized recording while allowing authorized recording, and it can prevent STFT attacks.

4.5 Evaluation

4.5.1 Overview

To evaluate the performance of Patronus, we select six news speech waveforms from Voice of America (VOA) and note these waveforms as A - F. The news speeches are read by a male, a female, or both alternatively, sometimes with background music.

A normal speaker (shown in Figure 4.7) is set to play these news waveforms, and we also read the news ourselves. While the news waveforms are played under different conditions, we start Patronus to interfere with the unauthorized recording device. Meanwhile, an authorized device is recording too. Later we apply scramble cancellation to recordings from the authorized device. After getting the scrambled recordings and scramble-canceled recordings, the following metrics are adopted to measure the performance of Patronus.

4.5.1.1 Perceptual Evaluation of Speech Quality (PESQ)

PESQ is a common-used metric of speech quality [126]. It is widely adopted by phone manufacturers, network equipment vendors, and telecom operators. Technically, the inputs include a clear speech signal as the reference and a signal that needs to be measured. The output is a Mean Opinion Score (MOS) [139] ranging from -0.5 to 4.5. A high PESQ score means that the corresponding speech has a high hearing quality and vice versa.

Typically, PESQ values ranging from 1.00 to 1.99 means "No meaning understood with any feasible effort" while those ranging from 3.80 to 4.50 meaning "Complete relaxation possible; no effort required". However, we cannot regard the audio recording as strict as lossless communication. To fit PESQ to characterize the performance of Patronus, we measure the PESQ of recordings without scrambling by turning off Patronus, and use that result as the baseline. As shown in Fig-

ure 4.8a, such recordings have PESQ between 2.2 and 2.7. We regard them as the upper bound of both unauthorized and authorized recordings.

In the following experiments, we use the PESQ implementation written in MATLAB [140] to compute the PESQ score.

4.5.1.2 Speech Recognition Vocabulary Accuracy (SRVA)

We also use a Speech Recognition service to measure the effectiveness of scrambling and descrambling. Specifically, we apply Google's Speech To Text (STT) service to transform the acoustic signals to text. We first use the STT service to recognize the original speech without interference and treat the recognized word sequence w_c as the ground truth. Then we use the STT service to recognize the scrambled speech and descrambled speech, and use w_s and w_d to denote their results, respectively. We name $\frac{\sum_{i \in w_s} isTrue(i \in w_c)}{|w_c|}$ (or $\frac{\sum_{i \in w_d} isTrue(i \in w_c)}{|w_c|}$) as the Speech Vocabulary Recognition Accuracy (SRVA) and use it to quantify the effectiveness of scrambling and descrambling. Note that $isTrue(i \in w_c)$ returns 1 when i is a word from w_c, and 0 when i is not a word from w_c. We define SRVA Error as $1 - \text{SRVA}$ which indicates the error rates of recognition with the STT service.

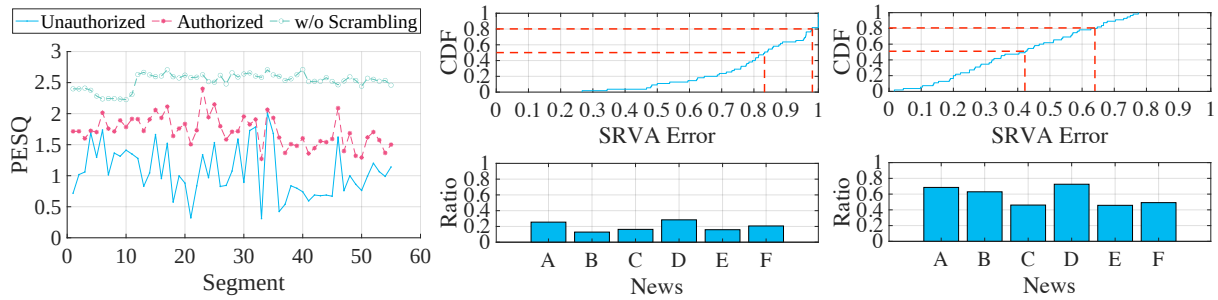
Using the above metrics, we try to answer the following questions:

- Can Patronus effectively scramble the unauthorized speech recordings?
- Can Patronus permit authorized devices to record the speech?
- Can Patronus work on different mobile devices?
- What is the impact of the distance between Patronus and a recorder?
- What is the impact of the reflection layer?
- What is the impact of the frequency switching time?
- Is it possible to perform real-time descrambling?

4.5.2 Effectiveness of Scrambling and Descrambling

We split the 6 news speech waveforms into 55 segments (1650 seconds in total), each 30 seconds long. Both the authorized and unauthorized device are Apple iPhone X in this experiment, so do the following experiments except that of Section 4.5.5. As shown in Figure 4.8a, with Patronus's scrambling, the hearing qualities of most segments are extremely low. Specifically, 44 out

of 55 (80.0%) segments have PESQ scores lower than 1.5. For SRVA, overall, only 551 out of 2796 (19.7%) words are recognized correctly. More detailed results are shown in Figure 4.8b. The upper half shows the CDF of the SRVA Error. We can know that 50% of the recordings have SRVA Error lower than 0.84, and 80% of the recordings have SRVA Error lower than 0.98. The lower half shows the ratio of SRVA between scrambled recordings and original waveforms. The results show that all of the news waveforms having a recognition rate lower than 0.3. Here we want to mention that if a word appears multiple times in a speech, SRVA would result in a high value or a low value compared to the actual word recognition rate. However, duplicated words have little impact because the duplicate rates of every segment, *i.e.* the ratio between the count of a specific word and the total count of words in the segment, are lower than 5%.



(a) PESQ (b) Scrambled (c) Descrambled

Figure 4.8 (a) PESQ of recordings captured by unauthorized and authorized devices, and PESQ of recordings without scrambling by turning off Patronus as the baseline. (b) Upper half: The CDF of SRVA Error of scrambled recordings from the unauthorized device. Lower half: The ratio of SRVA between scrambled recordings and original waveforms. (c) Upper half: The CDF of SRVA Error of descrambled recordings from the authorized device. Lower half: The ratio of SRVA between descrambled recordings and original waveforms.

To evaluate the effectiveness of descrambling, an authorized device records the speech under the scrambling from Patronus. The authorized device then cancels the scramble using the received key. As shown in Figure 4.8a, after descrambling, only 9 out of 55 (16.3%) segments having PESQ scores lower than 1.5. On average, descrambled recordings have 1.6x higher PESQ scores than their corresponding scrambled recordings. As for SRVA, we show the CDF of the SRVA Error in the upper half of Figure 4.8c. These results show that 50% of the descrambled recordings have SRVA Error lower than 0.43, which is 49% lower than scrambled recordings. Moreover, 80% of

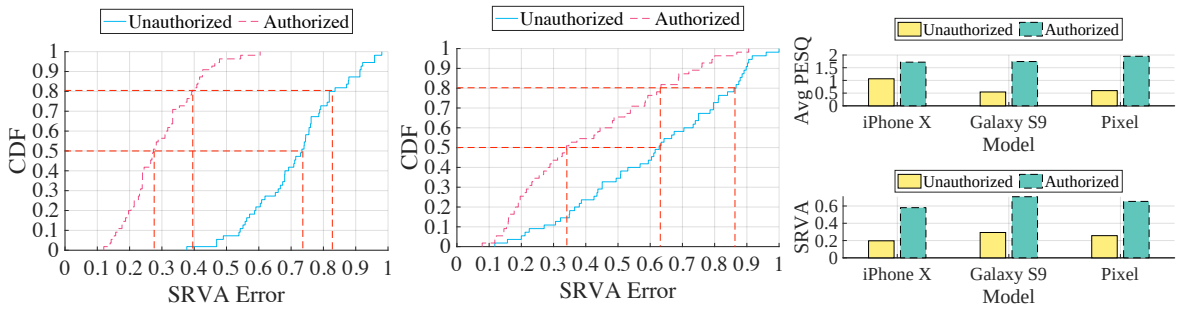
the descrambled recordings have SRVA Error lower than 0.64, which is 35% lower than scrambled recordings. As shown in the lower half of Figure 4.8c, ratios of SRVA between descrambled recordings and original waveforms are higher than 0.4 and lower than 0.8. They are at least 2x better than the scrambled recordings. The quality of the descrambled recordings is not as good as the original ones because there are residual components of the scramble after applying the NLMS adaptive filter. Moreover, background music and the volume of the original waveform also affects the quality of the descrambled recordings. For example, news C has a lower ratio after being descrambled by the authorized device compared to the other news clips because it has background music that could affect the performance of authorized devices. It also affects the SRVA of the record without scrambling, i.e., only 223 words are recognized from 295 in total. The reader of news E reads the news in a lower volume compared to others, so it has a lower ratio after being descrambled by the authorized device compared to the other news clips.

4.5.3 Effectiveness of Human Voice Scrambling and Descrambling

To verify whether Patronus works for real human speaking other than a sound player, we read the news and calculate SRVA. As shown in Figure 4.9a, Patronus can effectively scramble and descramble the human voice. Specifically, for the scrambled recordings, the median of SRVA Error is 0.74, and 80% of scrambled recordings have SRVA Error lower than 0.83. For the descrambled recordings, the median of SRVA Error is 0.27, and 80% of the descrambled recordings have SRVA Error lower than 0.4. The descrambling effectiveness of the human speaker is better than that of recorded sounds because recorded sounds from VOA sometimes play background music.

4.5.4 Effectiveness of Human Recognition to Scrambled Recordings and Descrambled Recordings

Because there might exist differences between machine learning-based speech recognition and human speech recognition, we invite 11 volunteers to write down words after listening to the 55 scrambled recordings and 55 descrambled ones. The results are shown in Figure 4.9b. People react differently to noise. Some people are very sensitive and the scrambled noise make them very uncomfortable. Note, the noise is generated by ultrasound speakers and only captured by the



(a) Human Reading (b) Human Hearing (c) Different Models

Figure 4.9 (a) Compare SRVA between before and after descrambling for the human voice. (b) Compare SRVA between before and after descrambling for human recognition. (c) Compare average PESQ and SRVA among different models.

nonlinear effects of microphones, so it will not disturb the people in the original conversation. It will only be heard after getting recorded by unauthorized devices. Further, authorized devices will be able to filter out such noises eliminating the discomfort for those listeners. The recovered information from humans listening to descrambled recordings is still better than that of humans listening to scrambled ones. 50% of the scrambled recordings have SRVA Error lower than 0.63, and 80% of the scrambled recordings have SRVA Error lower than 0.86. As a comparison, 50% of the descrambled recordings have SRVA Error lower than 0.34, and 80% of the descrambled recordings have SRVA Error lower than 0.63.

4.5.5 Effectiveness on Different Mobile Models

To verify whether Patronus works on different mobile models, we test it on three devices, an Apple iPhone X, a Samsung Galaxy S9, and a Google Pixel. We play all 55 segments using the normal speaker, and calculate average PESQs and SRVAs.

As shown in Figure 4.9c, less than 30% of words can be recognized by the STT service for all the unauthorized devices, and around 65% of words can be recognized for all the authorized devices. When the mobile devices are unauthorized, the average PESQ of iPhone X is 1.06, and the average PESQ of the other two models are even lower, roughly 0.5. When the mobile devices are authorized, they all achieve an average PESQ around 1.85. This demonstrates that Patronus works well for all devices; namely, it prevents all models from making good unauthorized recordings and

DT (ms) \ MSO / RT (s)	1	2	3	4	5	6
1	51	96	159	209	265	328
2	73	145	218	291	373	454
5	161	322	487	634	798	954
10	290	582	851	1108	1389	1653
20	548	1094	1653	2165	2695	3298
30	822	1617	2348	3088	3830	4563

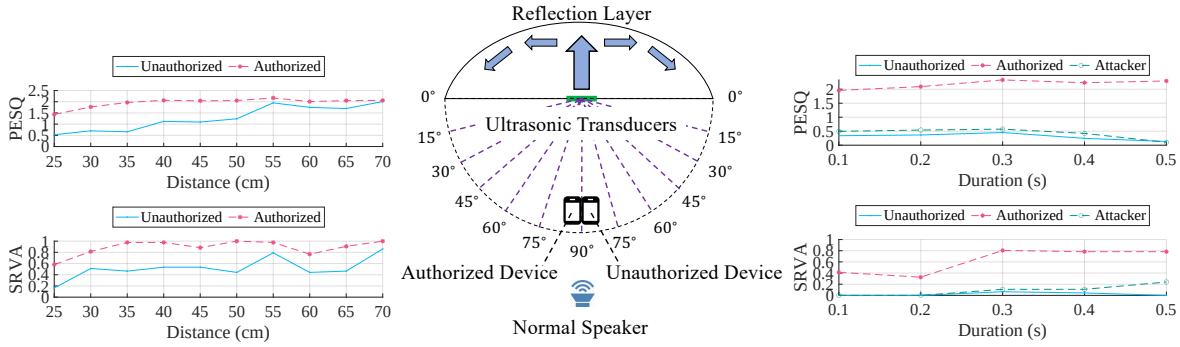
Table 4.1 Descramble time (DT) of different record times (RT) with different max scramble orders (MSO, the upper bound of k in Algorithm 4.1).

allows all models to make acceptable authorized recordings.

4.5.6 Impact of the Distance

We also characterize the impact of the distance between Patronus and the recording devices (both authorized and unauthorized). We put the Scramble Transmitter at the origin. A randomly-picked speech segment (which has 43 words) is played by a normal speaker, which simulates the talker. The authorized device and an unauthorized device are recording at the same time. Their distance to the Scramble Transmitter varies from 25 cm to 70 cm. Results of SRVA and PESQ between two devices are shown in Figure 4.10a. Overall, as the distance increases, the ultrasound would attenuate more. Therefore, the strength of the scramble decreases as the distance from the scramble transmitter increases. As a result, when the device is far enough away, both the authorized and unauthorized device can both record a clear speech. On the other hand, when devices are close enough, unauthorized devices produce recordings that are severely scrambled whereas authorized devices can recover much clearer speech using the secret key. The working area can be extended by using high power ultrasonic speakers, which we will discuss later. Here we want to mention that although there is a bump in Figure 4.10a at 55 cm with the SRVA, PESQs of 55cm and 60cm are close. This means that humans cannot see much difference between these two recordings, something we confirmed in person by listening to these recordings with this objective in mind. Thus, the SRVA bump at 55cm might be due to an error-correction mechanism of the Google STT engine; of course, since this is proprietary technology, we do not know how or why this error-

correction would produce such a performance bump for this recording.



(a) Different Distances (b) Angle Illustration (c) Different Durations

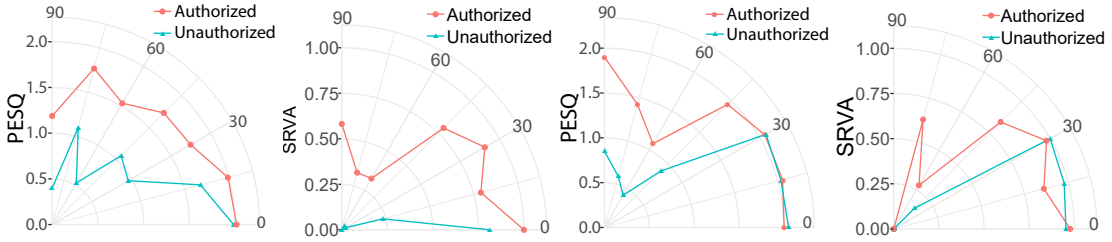
Figure 4.10 (a) Compare PESQ and SRVA at different distances. (b) Illustration of the reflection layer experiment. (c) Compare PESQ and SRVA with different frequency switching times.

4.5.7 Impact of the Reflection Layer

As we mentioned before, the ultrasound wave often propagates along a straight line. To enlarge the range of Patronus scrambling, we design a reflection layer. In this experiment, we apply the common speaker to play the chosen speech segment (43 words). As shown in Figure 4.10b, we point the ultrasonic speakers towards the reflection layer and change angles of both authorized and unauthorized devices to the ultrasonic speakers and measure Patronus' performance; in other experiments, the devices are always put at the 90° angle. We also measure the performance without using the reflection layer. We turn the ultrasonic speakers around so they face in the same direction as the normal speaker when we remove the reflection layer. The results when using the reflection layer are shown in Figure 4.11a and 4.11b, and the results without using the reflection layer are shown in Figure 4.11c and 4.11d. From the results, we see that with the reflection layer, Patronus can successfully scramble the unauthorized device when the angle is more than 15°, which is signficantly larger than the angle of more than 45° needed by Patronus without the reflection layer. Therefore, the reflection layer does significantly enlarge the scramble range of Patronus.

4.5.8 Impact of the Frequency Duration

We also measure the impact of the frequency duration. As we discussed in Section 4.3, we would like to make the duration of each frequency as short as possible. However, the shorter the



(a) PESQ (b) SRVA (c) PESQ (d) SRVA

Figure 4.11 (a) and (b): Compare PESQ and SRVA with the using of the reflection layer. (c) and (d): Compare PESQ and SRVA without the using of the reflection layer.

frequency duration is, the harder it is for authorized devices to descramble. To verify this feature, we put an authorized and an unauthorized device at 40 cm to Patronus and play the chosen segment (43 words) using the normal speaker. Both devices record the speech under Patronus using 5 different frequency durations: 0.1 s, 0.2 s, 0.3 s, 0.4 s and 0.5 s. We calculate PESQs and SRVAs for each duration. Moreover, we implement the attack model from Section 4.3.2, which first calculates approximate scramble frequencies using STFT and then attempts to cancel the scramble using an NLMS adaptive filter. We calculate PESQs and SRVAs for each duration and all devices including the attack model.

As shown in Figure 4.10c, for all durations, SRVAs of the unauthorized device are lower than 0.1, and PESQs are lower than 0.5. The authorized device has higher SRVAs and PESQs than the unauthorized device. Specifically, when the duration comes to 0.3 s, the SRVA reaches roughly 0.8 and PESQ exceeds 2.0. This verifies our claim that authorized devices can successfully descramble when the frequency duration is long enough.

A shorter duration also makes it harder for attackers to crack the scrambled record, *e.g.* SRVAs for the attacker also increase as the duration increases. Although both SRVAs and PESQs are higher than those of the unauthorized device, they are still too low to extract useful information. The reason why the NLMS adaptive filter fails is that the attacker cannot identify the scramble frequencies with enough accuracy. NLMS adaptive filter solves the optimization problem defined by Equation (4.4), which estimates the weight vector h_2'. Since convolution does not change the frequency of the signal, the attacker cannot make up for any offset existing between the correct

frequency and the result from STFT. According to the frequency resolution problem of STFT as discussed in Section 4.3.3.4, the simulated attacker in our experiment gets an average frequency offset around 3 Hz, which makes it hard to descramble the recording.

4.5.9 Descramble Time

Sometimes when we grant recording permission to a specific speaker, the speaker would like to perform real-time descrambling. Patronus can achieve this working with real-time smart devices such as Amazon Alexa. To prove this, we measure the descramble time for records with different durations on a laptop with an Intel Core i7-4870HQ 2.5 GHz CPU. Since different high-order scramble waves (second-order component, third-order component, ...) may exist in a record simultaneously, we measure descramble time as a function of different max scramble orders, *i.e.* the upper bound of k in Algorithm 4.1. As shown in Table 4.1, Patronus can descramble the record quickly. Specifically, when the record time is 1 s, Patronus can finish descrambling in 328 ms, even when the max scramble order is 6. This means that Patronus supports real-time descrambling.

4.6 Limitations and Future Works

Range: In our implementation, we use cheap and low power ultrasonic transducers to build the Scramble Transmitter. The result is a short working distance, *i.e.* less than 70 cm. To enlarge the working area to a wider range of angles, we designed a reflection layer and verified that it could enlarge the working area by using an iron wok in our prototype. We can also use a high power ultrasonic speaker to protect a larger area. Some commercial off-the-shelf devices can emit ultrasound which could be sensed in a larger area. For example, UPS+ [117] uses an ultrasonic speaker with a working area of 50m $\times$ 50m. However, it is expensive. We can reduce the cost by deploying one expensive speaker and multiple transducers like UPS+[117]. Here we provide users with three options to deploy Patronus according to their requirements such as working area and budget. The first option is to use cheap transducers and a reflection layer to protect a small area. The second is combining an expensive speaker and multiple transducers to protect a larger area. The third is using multiple expensive speakers to protect the largest area.

Volume: In our implementation, we assume the talker uses a normal volume, *i.e.* not too loud or

too quiet. However, the performance of Patronus does vary as a function of the speaker volume. For example, if the talker speaks too loudly, the scramble cannot mess up the recording; in the opposite extreme, a quiet talker cannot be recovered using descrambling. To adapt to different volumes, we can add a microphone to measure the talker's volume. With multiple deployed ultrasonic speakers or transducers, we can first detect the position of recording devices and then adjust the power of ultrasound emitted from the nearest speakers according to the talker's volume. There are two challenges that need to be solved. First, the microphone we use to measure the talker's volume can also be scrambled. Second, we need to localize recording devices before emitting scrambles. We leave these challenges as future work.

4.7 Conclusion

Acoustic privacy protection has always been an important topic. In this chapter, we study the nonlinear effects on commercial off-the-shelf microphones. Based on our study, we propose Patronus, which leverages the nonlinear effects to disrupt unauthorized devices from recording the speech while simultaneously allowing authorized devices to record clear speech audio. We implement and evaluate Patronus in a wide variety of representative scenarios. Results show that Patronus effectively blocks unauthorized devices from making secret recordings while allowing authorized devices to successfully make clear recordings.

While pervasive sensors in mobile devices rise privacy concern, Patronus shows the possibility of making use of these sensors to defend user privacy.

CHAPTER 5

CONCLUSION

IoT utilizes sensors as the information source of machine intelligence and achieves comprehensive understanding of the environment. Further reactions of actuators could enable complete automation of large infrastructures like Smart City and Smart Home. In this dissertation, we push the limit of IoT system design on mobile devices. We particularly introduce three IoT systems to show our study on overcoming common challenges of IoT applications. EyeLoc is a localization system for large shopping malls. It shows our effort on achieving a good balance among cost, computational power and real-time responses. SoundFlower is a sound source localization system for voice assistants. It presents our study on dealing with pervasive noises from both internal hardware and the environment. Patronus provides acoustic privacy protection against unauthorized recordings. While sensors rise privacy concern, Patronus shows the possibility of defending privacy by exploiting existing sensors.

With the rapid development of machine learning and sensing technology, plenty of IoT designs are being implemented and will benefit our lives. With mobile devices being the most common computational systems, pushing the limit of IoT system design on mobile device will play an essential role in smart applications.

BIBLIOGRAPHY

[1] IEEE, "Towards a definition of the internet of things (iot)," http://iot.ieee.org/definition.html, 2015.

[2] B. L. R. Stojkoska and K. V. Trivodaliev, "A review of internet of things for smart home: Challenges and solutions," *Journal of Cleaner Production*, vol. 140, pp. 1454–1464, 2017.

[3] C. Le Gal, J. Martin, A. Lux, and J. L. Crowley, "Smart office: Design of an intelligent environment," *IEEE Intelligent Systems*, no. 4, pp. 60–66, 2001.

[4] E. Ngai, F. Dressler, V. Leung, and M. Li, "Guest editorial special section on internet-of-things for smart cities and urban informatics," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 2, pp. 748–750, 2017.

[5] A. Wang, J. E. Sunshine, and S. Gollakota, "Contactless infant monitoring using white noise," in *Proceedings of ACM MobiCom*, 2019.

[6] C. Peng, G. Shen, Y. Zhang, Y. Li, and K. Tan, "Beepbeep: A high accuracy acoustic ranging system using cots mobile devices," in *Proceedings of ACM SenSys*, 2007.

[7] K. Qian, C. Wu, Z. Yang, Y. Liu, and K. Jamieson, "Widar: Decimeter-level passive tracking via velocity monitoring with commodity wi-fi," in *Proceedings of ACM Mobihoc*, 2017.

[8] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan, "The cricket location-support system," in *Proceedings of ACM MobiCom*, 2000.

[9] H. Chen, F. Li, and Y. Wang, "Echotrack: Acoustic device-free hand tracking on smart phones," in *Proceeding of INFOCOM*. IEEE, 2017, pp. 1–9.

[10] Y. Zheng, Y. Zhang, K. Qian, G. Zhang, Y. Liu, C. Wu, and Z. Yang, "Zero-effort cross-domain gesture recognition with wi-fi," in *Proceedings of ACM MobiSys*, 2019.

[11] C. Li, M. Liu, and Z. Cao, "Wihf: Enable user identified gesture recognition with wifi," in *Proceddings of IEEE INFOCOM*, 2020.

[12] W. Jiang, C. Miao, F. Ma, S. Yao, Y. Wang, Y. Yuan, H. Xue, C. Song, X. Ma, D. Koutsonikolas, W. Xu, and L. Su, "Towards environment independent device free human activity recognition," in *Proceedings of ACM MobiCom*, 2018.

[13] C. Godard, O. Mac Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," in *Proceedings of CVPR*, 2017.

[14] L. Weng and G. Brockman, "Techniques for training large neural networks," https://openai.com/research/techniques-for-training-large-neural-networks, (Accessed on Dec. 4, 2023).

[15] M. Liu, J. Du, Q. Zhou, Z. Cao, and Y. Liu, "Eyeloc: Smartphone vision-enabled plug-n-play indoor localization in large shopping malls," *IEEE Internet of Things Journal*, pp. 5585–5598, 2021.

[16] L. Li, M. Liu, Y. Yao, F. Dang, Z. Cao, and Y. Liu, "Patronus: Preventing unauthorized speech recordings with support for selective unscrambling," in *Proceedings of ACM SenSys*, 2020, p. 245–257.

[17] N. Roy, H. Hassanieh, and R. Roy Choudhury, "Backdoor: Making microphones hear inaudible sounds," in *Proceedings of ACM MobiSys*, 2017.

[18] S. Wang, S. Fidler, and R. Urtasun, "Lost shopping! monocular localization in large indoor spaces," in *Proceedings of IEEE ICCV*, 2015.

[19] L. M. Ni, Y. Liu, Y. C. Lau, and A. P. Patil, "Landmarc: Indoor location sensing using active rfid," *Wireless Networks*, vol. 10, no. 6, pp. 701–710, 2004.

[20] A. Haeberlen, E. Flannery, A. M. Ladd, A. Rudys, D. S. Wallach, and L. E. Kavraki, "Practical robust localization over large-scale 802.11 wireless networks," in *Proceedings of ACM MobiCom*, 2004.

[21] H. Liu, Y. Gan, J. Yang, S. Sidhom, Y. Wang, Y. Chen, and F. Ye, "Push the limit of wifi based localization for smartphones," in *Proceedings of ACM MobiCom*, 2012.

[22] D. Vasisht, S. Kumar, and D. Katabi, "Decimeter-level localization with a single wifi access point," in *13th USENIX Symposium on Networked Systems Design and Implementation (NSDI 16)*, 2016.

[23] Z. Yang, L. Jian, C. Wu, and Y. Liu, "Beyond triangle inequality: Sifting noisy and outlier distance measurements for localization," *ACM Trans. Sen. Netw.*, vol. 9, no. 2, pp. 26:1–26:20, 2013.

[24] Y.-S. Kuo, P. Pannuto, K.-J. Hsiao, and P. Dutta, "Luxapose: Indoor positioning with mobile phones and visible light," in *Proceedings of ACM MobiCom*, 2014.

[25] S. Zhu and X. Zhang, "Enabling high-precision visible light localization in today's buildings," in *Proceedings of ACM MobiSys*, 2017.

[26] S. Lin and T. He, "Smartlight: Light-weight 3d indoor localization using a single led lamp," in *Proceedings of ACM SenSys*, 2017.

[27] Q. Niu, M. Li, S. He, C. Gao, S. H. Gary Chan, and X. Luo, "Resource-efficient and automated image-based indoor localization," *ACM Trans. Sen. Netw.*, vol. 15, no. 2, pp. 19:1–19:31, 2019.

[28] Y. Tian, R. Gao, K. Bian, F. Ye, T. Wang, Y. Wang, and X. Li, "Towards ubiquitous indoor localization service leveraging environmental physical features," in *Proceedings of IEEE INFOCOM*, 2014.

[29] R. Gao, Y. Tian, F. Ye, G. Luo, K. Bian, Y. Wang, T. Wang, and X. Li, "Sextant: Towards ubiquitous indoor localization service by photo-taking of the environment," *IEEE Transactions on Mobile Computing*, vol. 15, no. 2, pp. 460–474, 2016.

[30] Y. Shu, C. Bo, G. Shen, C. Zhao, L. Li, and F. Zhao, "Magicol: Indoor localization using pervasive magnetic field and opportunistic wifi sensing," *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 7, pp. 1443–1457, 2015.

[31] H. Wu, Z. Mo, J. Tan, S. He, and S.-H. G. Chan, "Efficient indoor localization based on geomagnetism," *ACM Trans. Sen. Netw.*, vol. 15, no. 4, pp. 42:1–42:25, 2019.

[32] G. Bradski and A. Kaehler, *Learning OpenCV: Computer vision with the OpenCV library*. " O'Reilly Media, Inc.", 2008.

[33] R. Smith, "An overview of the tesseract ocr engine," in *Proceedings of ICDAR*, 2007.

[34] R. Jain, R. Kasturi, and B. G. Schunck, *Machine Vision*. McGraw-Hill, 1995.

[35] P. Zhou, M. Li, and G. Shen, "Use it free: instantly knowing your phone attitude," in *Proceedings of ACM MobiCom*, 2014.

[36] N. Roy, H. Wang, and R. Roy Choudhury, "I am a smartphone and i can tell my user's walking direction," in *Proceedings of ACM MobiSys*, 2014.

[37] S. Pertuz, D. Puig, and M. A. Garcia, "Analysis of focus measure operators for shape-from-focus," *Pattern Recognition*, vol. 46, no. 5, pp. 1415–1432, 2013.

[38] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions and reversals," in *Soviet Physics Doklady*, 1966.

[39] Google, "Battery historian," https://github.com/google/battery-historian, 18-Aug-2020.

[40] P. Bahl and V. N. Padmanabhan, "Radar: An in-building rf-based user location and tracking system," in *Proceedings of IEEE INFOCOM*, 2000.

[41] M. Youssef and A. Agrawala, "The horus wlan location determination system," in *Proceedings of ACM MobiSys*, 2005.

[42] Y.-C. Cheng, Y. Chawathe, A. LaMarca, and J. Krumm, "Accuracy characterization for metropolitan-scale wi-fi localization," in *Proceedings of ACM MobiSys*, 2005.

[43] S. Sen, B. Radunovic, R. R. Choudhury, and T. Minka, "You are facing the mona lisa: spot localization using phy layer information," in *Proceedings of ACM MobiSys*, 2012.

[44] I. Bisio, F. Lavagetto, A. Sciarrone, and S. Yiu, "A smart2 gaussian process approach for indoor localization with rssi fingerprints," in *2017 IEEE International Conference on Communications (ICC)*, 2017.

[45] Z. Yang, C. Wu, and Y. Liu, "Locating in fingerprint space: wireless indoor localization with little human intervention," in *Proceedings of ACM MobiCom*, 2012.

[46] A. Rai, K. K. Chintalapudi, V. N. Padmanabhan, and R. Sen, "Zee: zero-effort crowdsourcing for indoor localization," in *Proceedings of ACM MobiCom*, 2012.

[47] X. Tian, R. Shen, D. Liu, Y. Wen, and X. Wang, "Performance analysis of rss fingerprinting based indoor localization," *IEEE Transactions on Mobile Computing*, vol. 16, no. 10, pp. 2847–2861, 2017.

[48] K. Chintalapudi, A. Padmanabha Iyer, and V. N. Padmanabhan, "Indoor localization without the pain," in *Proceedings of ACM MobiCom*, 2010.

[49] S. Sen, R. R. Choudhury, and S. Nelakuditi, "Spinloc: Spin once to know your location," in *Proceedings of ACM HotMobile Workshop*, 2012.

[50] Z. Zhang, X. Zhou, W. Zhang, Y. Zhang, G. Wang, B. Y. Zhao, and H. Zheng, "I am the antenna: accurate outdoor ap location using smartphones," in *Proceedings of ACM MobiCom*, 2011.

[51] J. Xiong and K. Jamieson, "Arraytrack: A fine-grained indoor location system," in *Proceedings of NSDI*, 2013.

[52] S. Sen, J. Lee, K.-H. Kim, and P. Congdon, "Avoiding multipath to revive inbuilding wifi localization," in *Proceeding of ACM MobiSys*, 2013.

[53] A. T. Mariakakis, S. Sen, J. Lee, and K.-H. Kim, "Sail: Single access point-based indoor localization," in *Proceedings of ACM MobiSys*, 2014.

[54] S. Kumar, S. Gil, D. Katabi, and D. Rus, "Accurate indoor localization with zero start-up cost," in *Proceedings of ACM MobiCom*, 2014.

[55] M. Kotaru, K. Joshi, D. Bharadia, and S. Katti, "Spotfi: Decimeter level localization using wifi," in *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, 2015.

[56] C. Zhang and X. Zhang, "Pulsar: Towards ubiquitous visible light localization," in *Proceedings of ACM MobiCom*, 2017.

[57] Y.-L. Wei, C.-J. Huang, H.-M. Tsai, and K. C.-J. Lin, "Celli: Indoor positioning using polarized sweeping light beams," in *Proceedings of ACM MobiSys*, 2017.

[58] J. Dong, Y. Xiao, M. Noreikis, Z. Ou, and A. Ylä-Jääski, "imoon: Using smartphones for image-based indoor navigation," in *Proceedings of ACM Sensys*, 2015.

[59] R. Gao, M. Zhao, T. Ye, F. Ye, Y. Wang, K. Bian, T. Wang, and X. Li, "Jigsaw: Indoor floor plan reconstruction via mobile crowdsensing," in *Proceedings of ACM MobiCom*, 2014.

[60] Y. Shu, K. G. Shin, T. He, and J. Chen, "Last-mile navigation using smartphones," in *Proceedings of ACM MobiCom*, 2015.

[61] W. Huang, Y. Xiong, X.-Y. Li, H. Lin, X. Mao, P. Yang, and Y. Liu, "Shake and walk: Acoustic direction finding and fine-grained indoor localization using smartphones," in *Proceedings of IEEE INFOCOM*, 2014.

[62] K. Liu, X. Liu, and X. Li, "Guoguo: Enabling fine-grained indoor localization via smartphone," in *Proceeding of ACM MobiSys*, 2013.

[63] M. Risoud, J.-N. Hanson, F. Gauvrit, C. Renard, P.-E. Lemesre, N.-X. Bonne, and C. Vincent, "Sound source localization," *European annals of otorhinolaryngology, head and neck diseases*, pp. 259–264, 2018.

[64] P. D. Coleman, "An analysis of cues to auditory depth perception in free space," *Psychological bulletin*, p. 302, 1963.

[65] ——, "Failure to localize the source distance of an unfamiliar sound," *The Journal of the Acoustical Society of America*, pp. 345–346, 1962.

[66] E. Georganti, T. May, S. van de Par, A. Harma, and J. Mourjopoulos, "Speaker distance detection using a single microphone," *IEEE Transactions on Audio, Speech, and Language Processing*, pp. 1949–1961, 2011.

[67] K. Liu, X. Liu, and X. Li, "Acoustic ranging and communication via microphone channel," in *Proceeding of GLOBECOM*. IEEE, 2012, pp. 291–296.

[68] S. Yun, Y.-C. Chen, and L. Qiu, "Turning a mobile device into a mouse in the air," in *Proceeding of the 13th MobiSys*. ACM, 2015, pp. 15–29.

[69] W. Mao, J. He, and L. Qiu, "Cat: High-precision acoustic motion tracking," in *Proceeding of the 22nd MobiCom*. ACM, 2016, p. 69–81.

[70] J. C. Curlander and R. N. McDonough, *Synthetic aperture radar*. Wiley, 1991.

[71] W. Wang, J. Li, Y. He, and Y. Liu, "Symphony: Localizing multiple acoustic sources with a

single microphone array," in *Proceeding of the 18th SenSys*. ACM, 2020, p. 82–94.

[72] E. B. Newman, "Experimental psychology," 1954.

[73] J. Benesty, J. Chen, and Y. Huang, "Time-delay estimation via linear interpolation and cross correlation," *IEEE Transactions on Speech and Audio Processing*, pp. 509–519, 2004.

[74] I. L. Freire and J. A. Apolinário, "Doa of gunshot signals in a spatial microphone array: Performance of the interpolated generalized cross-correlation method," in *2011 Argentine School of Micro-Nanoelectronics, Technology and Applications*. IEEE, 2011, pp. 1–6.

[75] C. Knapp and G. Carter, "The generalized correlation method for estimation of time delay," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, pp. 320–327, 1976.

[76] B. Van Den Broeck, A. Bertrand, P. Karsmakers, B. Vanrumste, M. Moonen *et al.*, "Time-domain generalized cross correlation phase transform sound source localization for small microphone arrays," in *2012 5th European DSP Education and Research Conference*. IEEE, 2012, pp. 76–80.

[77] Y. T. Chan, R. Hattin, and J. Plant, "The least squares estimation of time delay and its use in signal detection," in *IEEE Transactions on Acoustics, Speech, and Signal Processing*. IEEE, 1978, pp. 217–222.

[78] R. Roy and T. Kailath, "Esprit-estimation of signal parameters via rotational invariance techniques," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, pp. 984–995, 1989.

[79] S. Shahbazpanahi, S. Valaee, and M. H. Bastani, "Distributed source localization using esprit algorithm," *IEEE Transactions on Signal Processing*, pp. 2169–2178, 2001.

[80] Y. L. Sit, C. Sturm, J. Baier, and T. Zwick, "Direction of arrival estimation using the music algorithm for a mimo ofdm radar," in *2012 IEEE Radar Conference*. IEEE, 2012, pp. 0226–0229.

[81] R. Schmidt, "Multiple emitter location and signal parameter estimation," *IEEE Transactions on Antennas and Propagation*, pp. 276–280, 1986.

[82] T. C. Collier, A. N. Kirschel, and C. E. Taylor, "Acoustic localization of antbirds in a mexican rainforest using a wireless sensor network," *The Journal of the Acoustical Society of America*, pp. 182–189, 2010.

[83] I. Constandache, S. Agarwal, I. Tashev, and R. R. Choudhury, "Daredevil: Indoor location using sound," *ACM SIGMOBILE Mobile Computing and Communications Review*, pp. 9–19, 2014.

[84] C. T. Ishi, J. Even, and N. Hagita, "Using multiple microphone arrays and reflections for 3d localization of sound sources," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 3937–3942.

[85] J. Sallai, W. Hedgecock, P. Volgyesi, A. Nadas, G. Balogh, and A. Ledeczi, "Weapon classification and shooter localization using distributed multichannel acoustic sensors," *Journal of Systems Architecture*, pp. 869–885, 2011.

[86] J. Yang, S. Sidhom, G. Chandrasekaran, T. Vu, H. Liu, N. Cecan, Y. Chen, M. Gruteser, and R. P. Martin, "Detecting driver phone use leveraging car speakers," in *Proceeding of the 17th MobiCom*. ACM, 2011, pp. 97–108.

[87] H. Chen, F. Li, and Y. Wang, "Echoloc: Accurate device-free hand localization using cots devices," in *Proceeding of the 45th ICPP*. IEEE, 2016, pp. 334–339.

[88] W. Huang, Y. Xiong, X.-Y. Li, H. Lin, X. Mao, P. Yang, Y. Liu, and X. Wang, "Swadloon: Direction finding and indoor localization using acoustic signal by shaking smartphones," *IEEE Transactions on Mobile Computing*, pp. 2145–2157, 2014.

[89] Y. Zhang, J. Wang, W. Wang, Z. Wang, and Y. Liu, "Vernier: Accurate and fast acoustic motion tracking using mobile devices," in *Proceeding of INFOCOM*. IEEE, 2018, pp. 1709–1717.

[90] P. Lazik and A. Rowe, "Indoor pseudo-ranging of mobile devices using ultrasonic chirps," in *Proceeding of the 10th SenSys*. ACM, 2012, pp. 99–112.

[91] N. Garg, Y. Bai, and N. Roy, "Owlet: Enabling spatial information in ubiquitous acoustic devices," in *Proceedings of the 19th MobiSys*. ACM, 2021, p. 255–268.

[92] S. Shen, D. Chen, Y.-L. Wei, Z. Yang, and R. R. Choudhury, "Voice localization using nearby wall reflections," in *Proceeding of the 26th MobiCom*. ACM, 2020, pp. 1–14.

[93] M. Wang, W. Sun, and L. Qiu, "MAVL: Multiresolution analysis of voice localization," in *Proceeding of the 18th NSDI)*. USENIX, 2021, pp. 845–858.

[94] M. S. Brandstein and H. F. Silverman, "A robust method for speech signal time-delay estimation in reverberant rooms," in *1997 IEEE International Conference on Acoustics, Speech, and Signal Processing*. IEEE, 1997, pp. 375–378.

[95] S. Gupta, D. Morris, S. Patel, and D. Tan, "Soundwave: Using the doppler effect to sense gestures," in *Proceeding of CHI*. ACM, 2012, pp. 1911–1914.

[96] B. Zhou, M. Elbadry, R. Gao, and F. Ye, "Battracker: High precision infrastructure-free mobile device tracking in indoor environments," in *Proceedings of ACM SenSys*, 2017.

[97] Z. Sun, A. Purohit, R. Bose, and P. Zhang, "Spartacus: Spatially-aware interaction for mobile devices through energy-efficient audio sensing," in *Proceeding of the 11th MobiSys*. ACM, 2013, pp. 263–276.

[98] K. Sun, T. Zhao, W. Wang, and L. Xie, "Vskin: Sensing touch gestures on surfaces of mobile devices using acoustic signals," in *Proceeding of the 24th MobiCom*. ACM, 2018, pp. 591–605.

[99] W. Wang, A. X. Liu, and K. Sun, "Device-free gesture tracking using acoustic signals," in *Proceeding of the 22nd MobiCom*. ACM, 2016, pp. 82–94.

[100] S. Yun, Y.-C. Chen, H. Zheng, L. Qiu, and W. Mao, "Strata: Fine-grained acoustic-based device-free tracking," in *Proceeding of the 15th MobiSys*. ACM, 2017, pp. 15–28.

[101] H. Jin, C. Holz, and K. Hornbæk, "Tracko: Ad-hoc mobile 3d tracking using bluetooth low energy and inaudible signals for cross-device interaction," in *Proceeding of the 28th UIST*. ACM, 2015, pp. 147–156.

[102] J. Qiu, D. Chu, X. Meng, and T. Moscibroda, "On the feasibility of real-time phone-to-phone 3d localization," in *Proceeding of the 9th SenSys*. ACM, 2011, pp. 190–203.

[103] J. H. DiBiase, *A High-accuracy, Low-latency Technique for Talker Localization in Reverberant Environments Using Microphone Arrays*. Brown University, 2000.

[104] B. D. Rao and K. S. Hari, "Performance analysis of root-music," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, pp. 1939–1949, 1989.

[105] G. Carter, C. Knapp, and A. Nuttall, "Estimation of the magnitude-squared coherence function via overlapped fast fourier transform processing," *IEEE Transactions on Audio and Electroacoustics*, pp. 337–344, 1973.

[106] A. E. Beaton and J. W. Tukey, "The fitting of power series, meaning polynomials, illustrated on band-spectroscopic data," *Technometrics*, pp. 147–185, 1974.

[107] S. Morgenthaler, "Fitting redescending m-estimators in regression," in *Robust Regression*. Routledge, 2019, pp. 105–128.

[108] M. S. Brandstein, J. E. Adcock, and H. F. Silverman, "A practical time-delay estimator for localizing speech sources with a microphone array," *Computer Speech & Language*, pp. 153–169, 1995.

[109] J. B. Allen and D. A. Berkley, "Image method for efficiently simulating small-room acoustics," *The Journal of the Acoustical Society of America*, pp. 943–950, 1979.

[110] I. Cohen and B. Berdugo, "Noise estimation by minima controlled recursive averaging for

robust speech enhancement," *IEEE Signal Processing Letters*, pp. 12–15, 2002.

[111] wiki.seeedstudio.com, "Respeaker 6-mic circular array kit for raspberry pi." [Online]. Available: https://wiki.seeedstudio.com/ReSpeaker_6-Mic_Circular_Array_kit_for_Raspberry_Pi/

[112] raspberrypi.com, "Raspberry pi 4 model b." [Online]. Available: https://www.raspberrypi.com/products/raspberry-pi-4-model-b

[113] T. Guardian, "Apple apologises for allowing workers to listen to siri recordings," https://www.theguardian.com/technology/2019/aug/29/apple-apologises-listen-siri-recordings, (Accessed on Feb. 28, 2020).

[114] CNBC, "Amazon echo recorded conversation, sent to random person: report," https://www.cnbc.com/2018/05/24/amazon-echo-recorded-conversation-sent-to-random-person-report.html, (Accessed on Feb. 28, 2020).

[115] V. News, "Ukrainian prime minister offers resignation," https://www.voanews.com/a/europe_ukrainian-prime-minister-offers-resignation/6182735.html, (Accessed on Dec. 5, 2023).

[116] Y.-C. Tung and K. G. Shin, "Exploiting sound masking for audio privacy in smartphones," in *Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security*, 2019.

[117] Q. Lin, Z. An, and L. Yang, "Rebooting ultrasonic positioning systems for ultrasound-incapable smart devices," in *Proceedings of ACM MobiCom*, 2019.

[118] "Anti-eavesdropping and recording blocker device," Patent, China Patent 201320228440, Oct. 2013.

[119] N. Roy, S. Shen, H. Hassanieh, and R. R. Choudhury, "Inaudible voice commands: The long-range attack and defense," in *Proceedings of NSDI*, 2018.

[120] G. Zhang, C. Yan, X. Ji, T. Zhang, T. Zhang, and W. Xu, "Dolphinattack: Inaudible voice commands," in *Proceedings of ACM CCS*, 2017.

[121] T. Chen, L. Shangguan, Z. Li, and K. Jamieson, "Metamorph: Injecting inaudible commands into over-the-air voice controlled systems," in *Network and Distributed Systems Security (NDSS) Symposium*, 2020.

[122] X. Zhou, X. Ji, C. Yan, J. Deng, and W. Xu, "Nauth: Secure face-to-face device authentication via nonlinearity," in *proceedings of IEEE INFOCOM*, 2019.

[123] Q. Yan, K. Liu, Q. Zhou, H. Guo, and N. Zhang, "Surfingattack: Interactive hidden attack on

voice assistants using ultrasonic guided wave," in *Network and Distributed Systems Security (NDSS) Symposium*, 2020.

[124] A. Rovner, "The principle of ultrasound," https://www.echopedia.org/wiki/The_principle_of_ultrasound, 2015.

[125] A. H. Sayed, *Fundamentals of adaptive filtering*. John Wiley & Sons, 2003.

[126] A. W. Rix, J. G. Beerends, M. P. Hollier, and A. P. Hekstra, "Perceptual evaluation of speech quality (pesq)-a new method for speech quality assessment of telephone networks and codecs," in *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No. 01CH37221)*, 2001.

[127] Y. He, J. Bian, X. Tong, Z. Qian, W. Zhu, X. Tian, and X. Wang, "Canceling inaudible voice commands against voice control systems," in *Proceedings of ACM MobiCom*, 2019.

[128] A. Wang, C. Peng, O. Zhang, G. Shen, and B. Zeng, "Inframe: Multiflexing full-frame visible communication channel for humans and devices," in *Proceedings of the 13th ACM Workshop on Hot Topics in Networks*, 2014.

[129] A. Wang, Z. Li, C. Peng, G. Shen, G. Fang, and B. Zeng, "Inframe++ achieve simultaneous screen-human viewing and hidden screen-camera communication," in *Proceedings of ACM MobiSys*, 2015.

[130] V. Nguyen, Y. Tang, A. Ashok, M. Gruteser, K. Dana, W. Hu, E. Wengrowski, and N. Mandayam, "High-rate flicker-free screen-camera communication with spatially adaptive embedding," in *Proceedings of IEEE INFOCOM*, 2016.

[131] K. Zhang, Y. Zhao, C. Wu, C. Yang, K. Huang, C. Peng, Y. Liu, and Z. Yang, "Chromacode: A fully imperceptible screen-camera communication system," *IEEE Transactions on Mobile Computing*, 2019.

[132] Q. Wang, K. Ren, M. Zhou, T. Lei, D. Koutsonikolas, and L. Su, "Messages behind the sound: real-time hidden acoustic signal capture with smartphones," in *Proceedings of ACM MobiCom*, 2016.

[133] M. Zhou, Q. Wang, K. Ren, D. Koutsonikolas, L. Su, and Y. Chen, "Dolphin: Real-time hidden acoustic signal capture with smartphones," *IEEE Transactions on Mobile Computing*, vol. 18, no. 3, pp. 560–573, 2018.

[134] L. Zhang, C. Bo, J. Hou, X.-Y. Li, Y. Wang, K. Liu, and Y. Liu, "Kaleido: You can watch it but cannot record it," in *Proceedings of ACM MobiCom*, 2015.

[135] S. Zhu, C. Zhang, and X. Zhang, "Automating visual privacy protection using a smart led," in *Proceedings of ACM MobiCom*, 2017.

[136] I. R. Titze and D. W. Martin, "Principles of voice production," 1998.

[137] R. J. Baken and R. F. Orlikoff, *Clinical measurement of speech and voice*. Cengage Learning, 2000.

[138] S. Shen, N. Roy, J. Guan, H. Hassanieh, and R. R. Choudhury, "Mute: bringing iot to noise cancellation," in *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*, 2018.

[139] I. Rec, "P. 800.1, mean opinion score (mos) terminology," *International Telecommunication Union, Geneva*, 2006.

[140] K. Wojcicki, "PESQ MATLAB wrapper," https://www.mathworks.com/matlabcentral/fileexchange/33820-pesq-matlab-wrapper, (Accessed on Dec. 6, 2023).