Sping 2018 CS755 Project Report Acoustic Sensing

Mingrui Han
George Mason University
4400 University Drive
Fairfax, VA
mhan8@gmu.com

Joshua Lilly
George Mason University
4400 University Drive
Fairfax, VA
jlilly3@gmu.com

ABSTRACT

More and more IoT devices emerge into our daily life, aiming to ease our living style. Those devices such as smart TV, smart fridge will have more functionality to provide a better user experience. A traditional controller that uses buttons cannot meet the expectation of the usage. In this paper, we implement AAMouse [13], a system that tracks the device using acoustic sensing. It enables users to use their mobile devices as the controller to draw freely in the air. AAMouse adopts inaudible sound wave and Doppler shift effect to track the device movement. Human being cannot hear the sound wave frequency above 17kHz. However, these frequencies are supported in commercial devices. AAMouse takes advantage of this unused frequencies. Also, AAMouse does not require specialized hardware to operate. We implemented AAMouse in two platforms, PC and Android tablet. To refine the tracking result, we applied maximum ratio combining and signal smoothing filter. Then we conducted the evaluation experiment. The actual error is larger than the AAMouse paper. We also discussed the possible reason of the error. In addition, the difference between our implementation and the original work is discussed in this work as well.

Keywords

AAMouse, acoustic sensing, inaudible sound wave

1. INTRODUCTION

In this work, we implemented AAMouse, a system that utilize acoustic signal and Doppler shift effect to track the device movement. AAMouse uses inaudible sound wave that operates above 17kHz. Researches indicate that human being cannot hear the sound wave above 17kHz. However, current commercial speakers support those frequencies. The unique property of AAMouse is that it can operates in our daily devices and it does not require any specialized hardware. A system that uses mobile device to track its movement can offer a variety of usage. User would be able to control the appliance anywhere they want by carrying their mobile devices and such system can also enable user to have a better user experience than the traditional remote controller. Moreover, mobile devices have IMU sensors built in, which can also be used to aid the accuracy of the tracking result.

The original AAMouse uses STFT, Doppler shift, maximum ratio combining (MRC) Kalman filter, and particle filter. Our system is different than the original work with minor changes.

Our system sends the sound wave and analyze the received signal using Doppler shift effect. The frequency shift will give information to calculate the velocity. We then used the velocity to calculate the distance that the device traveled. To refine the tracking result, we further apply MRC and signal smoothing filter.

We implemented the system in PC using matlab and in Android using Java. Some technical details we are using are different than the original paper, but the overall concepts are the same. During the experimental evaluation part, we observed that the actual system has larger error than the original paper. We provided detailed reasoning to explain what the potential causes are and what are the potential solutions.

The rest of this paper is organized as follows. Section 2 introduces some related works in this area. Section 3 provides some possible applications of this work in the future. Section 4 summarized the contribution of each group member in this work. Section 5 provides an overview of the project, compared to the original work. Section 6 includes all the technical details of the implementation. We evaluated the system in section 7.

2. RELATED WORK

AAMouse is the first tracking system using acoustic signals. It uses the unused inaudible sound wave that are available in commercial devices. Other than AAMouse, some other works also utilize acoustic signals. CAT [4] propose a high precision acoustic tracking using the combination of FMCW with Doppler shift. It modifies the traditional FMCW which sends a chirp signal and applied the modified FMCW to a distributed system. Then it combines with Doppler shift to refine the results. The evaluation shows that this system outperforms AAMouse. Wang [9] proposed a device free tracking using inaudible sound wave phase information. This work differs from previous ones in the fact that it does not track the movement of a device, instead, it tracks the movement of the finger. Such device free tracking will have more applications. For example, it can help to control the smart watch, and user can wear gloves. FinerIO [5] proposed a tracking system using OFDM and its echo profile. It is also a device free tracking system.

In RF-based schemes, ArrayTrack [11] uses WiFi to realize fine-grained tracking system with a median error of 23 cm using 16 antennas. RF-IDraw [8] adopts 8 RFID antennas with different spacing. This system has the median error of 3.7 cm. WiDraw [7] utilize the angle of arrive(AoA) based on CSI to enable hand-free drawing. Its median error is 5cm when sing 25 WiFi transmitters. mTrack [10] has a high accuracy using 60 GHz RF signals. Tagoram [12] is also a RFID based tracking system that utilizes commercial off-the-shelf devices. Its median error is 12 cm. Compared with RF based tracking systems, audio tracking can achieve a fine-grained system with high accuracy in mm level.

IMU sensors can also be used for localization tracking. For example, Microsoft X-box Kinect uses depth sensor and Nintendo Wii uses infrared cameras to track. Both system works for line-of-sight environment only and they suffer from error accumulation. Li [3] uses acceleration data from accelerator for localization tracking. However, it occurs significant error from measurement. IMU sensor can also be used to improve the tracking result. Zee [6] uses such approach to aid the WiFi tracking accuracy. Compared with sensor based tracking systems, audio tracking has a lower signal processing cost and it supports condition other than line- of-sight.

3. APPLICATIONS

IoT devices becomes more and more popular nowadays. Some appliances such as smart TV, smart fridge offer more functions than we ever had before. Merely using a remote controller with buttons cannot meet the expected user experience offered by those appliances. A mouse has been on of the best design for controllers. It enables a high level of freedom of controlling the devices. There are recently mouse developed for smart TV to allow users have a better exploration experience.

A traditional mouse requires a smooth surface to operate, which is feasible for a typical PC. However, when it comes to the smart TV or smart fridge, it is not suitable to have a flat surface near them and it is not suitable for user to maintain a fixed position when using the appliance. For example, an user may want to control a TV on couch, on desk, or on the dining table. Having a mouse in each one of the location is not feasible in real world. Our system can be applied in this scenario by enabling a mouse in the air on the go. User would be able to control the appliance wherever they want with rich control options.

4. ROLES AND COLLABORATION

We discussed the project implementation details together, including what we should implement, what the time line would be, what the issues are, and what we should do to solve them. For each technical concept we need, such as MRC, Kalman filter, STFT, Doppler Shift, and the particle filter, we discussed the implementation procedure, potential issues, and the possible solutions.

Joshua Lilly: Signal processing portion in MATLAB, Maximal Ratio Combining, Outlier removal Kalman Filtering, Distance and point Calculation.

Mingrui Han: Implement the system in Android tablet. Review and discuss matlab code. Implement MRC, distance calculation, following the matlab code. Research and implement FFT, signal filter, and other third party tools which java does not have.

5. PROJECT OVERVIEW

We implement a modified version of AAMouse in two platforms, PC (matlab) and Android tablet. Our project is different than the original work in some way but it has the same concept as AAMouse. The key techniques used in AAMouse are, Doppler shift, distance calculation, maximum ratio combining (MRC), Kalman filter, calibration phase, and particle filter. As we implement the work, we found that the paper omits some detail for implementation. Compared with AAMouse, we implemented a modified version of the system. Our system uses Doppler shift, distance calculation, MRC, and signal smoothing filter. Detail will be given in the later sections.

5.1 Original Work

The key idea of the original work is using Doppler shift as an estimate of the device velocity. The Doppler shift is a well known effect where the frequency of a signal changes as a sender or a receiver moves. In our case, the sender is fixed, so we only consider the movement of the receiver. Doppler shift is observed in both RF wave and acoustic wave. Because of its speed propagation speed and narrower bandwidth, acoustic signal can achieve a higher accuracy than RF signal. The detailed equation is shown below.

$$v = \frac{F^s}{F} \times C \tag{1}$$

F denotes the original frequency of the signal. F^s is the frequency shift of the signal. C is the signal speed, which is the sound wave speed in this case. After the calculation of the velocity v, AAMouse integrates it with time to obtain the distance of the device movement.

$$D = D + v \times t \tag{2}$$

In 1D tracking, D is the result. In 2D tracking, D is the distance for each speaker. We have 2 distances of the device to each speaker and we have the distance between the two speakers. We will be able to track the device movement using basic triangle properties.

To improve the accuracy, AAMouse uses maximum ratio combining (MRC). Measuring from a single frequency may not be reliable. Thus, AAMouse sends the signal in 10 frequencies, with a guarding frequency of 200Hz. Then it applied MRC to average the received signal weighted by the inverse of the noise variance. After performing MRC, AAMouse adopts Kalman filter to smooth the estimation with noise covariance and measurement noise covarince both to be 0.00001.

To find the distance between the speakers, AAMouse introduced a Doppler shift based calibration process. The TV emits the inaudible sound and user scans the TV with his hands holding the device. The users start from the left end of the TV to the right end of the TV. It finds the points where the value of Doppler shift change from positive to negative. Those points indicate the time that uses spent to move the device between the speakers.

AAMouse finds the initial device location by applying a particle filter. It generates many particles uniformly distributed in the area. Each particle indicates a possible initial location of the device. In the next Doppler shift interval, the system checks the device movement of each particle. If the movement is not feasible, such particle will be filtered out.

5.2 Project Summary

We have implemented a modified version of AAMouse. Our system includes the key concept of the original work, which is the Doppler shift. Doppler shift in our system is basically the same as it in the AAMouse, except in the Android part, where there is no STFT library available. Original AAMouse uses STFT to find the peak frequencies of the received signals. Our PC implementation does the same procedure as AAMouse, but our Android tablet uses FFT instead. Basic concept of Doppler shift is given in the previous section. Implementation detail will be given in section 6.2.

To refine the tracking result, original AAMouse uses maximum ratio combining. We also implemented such technique in both our system. We sent the signal in 10 different frequencies and average the received signal weighted by the inverse of the signal noise. The implementation detail will be given in section 6.2.

Then AAMouse applied Kalman filter after the MRC to further smooth the signal. We did not implement Kalman filter /*************** include the reason why we did not use it, and what is used in PC part. ***************/ In Android side, there are not a lot of options to choose from. For example, Kalman filter does not exist in Android platform. We used a signal filter from [2], called OneEuroFilter, to smooth the signal.

The distance calculation portion of the work are the same as AAMouse. We calculated the distance in each axis and then calculated the intersection of the two circles from two speakers with the radius as the distance calculated before. Section 6.2 will discuss the detail of the calculation.

AAMouse implements particle filter to find the initial location of the device. Due to the high complexity of particle filter concept and the missing detail of the implementation in the original paper. We were not able to replicate the particle filter from AAMouse. Therefore, our system does not have the particle filter implemented, which could cause the system to be error-prone.

6. DESIGN AND IMPLEMENTATION

We implemented the system in two platforms, PC using matlab and Android using Java. In this section, we will discuss the implement details of our system. It includes the hardware we used, the experimental setup we used, the implementation approach we applied, the possible problem we faced and the solutions we took.

6.1 Setup

/**************** I added two section for Android, so you can use below two section for you. *****************/ Scenarios and setup of your implementation, including location, hardware, software, and so on. If possible, include pictures and figures to be illustrative. Compare your setup to the original paper, and explain why you did so

/**************** Include some figures for best result, just the one from the slides are fine. I used the same setup with yours. *****************/

6.2 Technical Details

All the technical details related to your project should be placed here including, but not limited to, techniques and methodologies involved, theories behind thee, how they were implemented, what was the difference from the originial paper, what was not mentioned in the paper and how you overcame it, and so on. Be specific.

/**************** Include doppler shift parameters, MRC, distance calculation, signal filter *****************/

6.3 Setup-Android

We used one Samsung SM-T320 tablet and two Logitech S-120 speakers for experiment hardware. The experiment setup is the same as the one mentioned above for our PC implementation. We keep the distance between the speaker to the device to be one meter and the distance between two speakers also to be one meter.

6.4 Technical Details-Android

The sampling rate we used is the same as the original paper, 44.1 KHz. It is also the default sampling rate supported by the device. By default, we uses 1759 as the buffer size to store the audio samples. Such size is chosen by the convention from the Android AudioRecorder class. Because in Android system, there is no signal processing libraries to use, we cannot call STFT function in one line as we did in PC side. Thus the first issue is how to calculate frequency shift in Doppler shift. We used a third party FFT tool from [1]. This FFT function will return the same number of points as the buffer size of the stored signal. The even index of the FFT point are the imaginary part and the odd index of the FFT point are the real part. We then found the frequencies with the greatest FFT value and subtracted it with the sending frequency to obtain the frequency shift. Then we are able to calculate the velocity.

To improve the accuracy of the system, we also implement MRC in Android side, the technical detail is the same as above for the PC section. After applying MRC, the original work uses Kalman filter to smooth the data. However, there is no such filter available in Android system. Instead, we applied a third party filter from [2] called OneEuroFilter to smooth the data.

After the post processing, we were able to get the frequency shift in each axis, averaging five sound frequencies. Then we calculate the distance using the same equations mentioned above.

6.5 Lessons learned

What are the knowledge obtained from this project? Clearly state what you have learned from this project, beyond what is on the original paper. This includes practical issues youâĂŹve faced, how you were able to troubleshoot them. Even if you could not fix it, describe what you have tried, why you were unable to solve it, and your hypothesis on what would be the cause. If you were not able to implement the entire system in the originial paper, explain why. /**************** You can include your part here, before mine. I think we can use the same section. *****************/

In Android side, using buffer size of 1759 is no doubt too small for FFT. Such value will lead to a very low frequency resolution, which will cause the distance calculation to be very inaccurate. Same theory is also mentioned in the original work. The correct buffer size should be 44100. However, this will not work for the two following reasons. First, such number of the buffer size will cause the app to stop working as we tested using the tablet. Second, even if the app will not crash, this huge value will take one second to record. A long FFT does not allow us to track the device in real time. In one second, the device may move to other location. As a result, we think the small buffer size is the cause of the system tracking error, even though we were able to obtain the frequency shift.

7. EVALUATION

Performance evaluation and potential improvement. Should include below subsections. Feel free to add (do not remove) subsections and reorder them.

7.1 Performance and Analysis

Be illustrative with graphs and figures. However, donâĂŹt just simply enlist the results, but explain them and provide anal- ysis/insights obtained from the implementation and exper- iments. For example, why does your project performs bet- ter/worse than what is reported in the original paper? Why does it perform differently under various scenarios? Any un- expected results are found, and why

7.2 Potential Improvements

From your experiment experience, what do you think can be done to improve the performance. Please justify

8. REFERENCES

[1] P. N. Amsen. Noise. https://github.com/paramsen/noise, 2017.

[2] R. de Courville. Signal filter (beta). https://github.com/SableRaf/signalfilter, 2015.

[3] F. Li, C. Zhao, G. Ding, J. Gong, C. Liu, and F. Zhao. A reliable and accurate indoor localization method using phone inertial sensors. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, UbiComp '12, pages 421–430, New York, NY, USA, 2012. ACM.

[4] W. Mao, J. He, H. Zheng, Z. Zhang, and L. Qiu. High-precision acoustic motion tracking: Demo. In *Proceedings of the 22Nd Annual International Conference on Mobile Computing and Networking*, MobiCom '16, pages 491–492, New York, NY, USA, 2016. ACM.

[5] R. Nandakumar, V. Iyer, D. Tan, and S. Gollakota. Fingerio: Using active sonar for fine-grained finger tracking. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, pages 1515–1525, New York, NY, USA, 2016. ACM.

[6] A. Rai, K. K. Chintalapudi, V. N. Padmanabhan, and R. Sen. Zee: Zero-effort crowdsourcing for indoor localization. In *Proceedings of the 18th Annual International Conference on Mobile Computing and Networking*, Mobicom '12, pages 293–304, New York, NY, USA, 2012. ACM.

[7] L. Sun, S. Sen, D. Koutsonikolas, and K.-H. Kim. Widraw: Enabling hands-free drawing in the air on commodity wifi devices. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, MobiCom '15, pages 77–89, New York, NY, USA, 2015. ACM.

[8] D. Vasisht, J. Wang, and D. Katabi. Rf-idraw: Virtual touch screen in the air using rf signals. In *Proceedings of the 6th Annual Workshop on Wireless of the Students, by the Students, for the Students*, S3 '14, pages 1–4, New York, NY, USA, 2014. ACM.

[9] W. Wang, A. X. Liu, and K. Sun. Device-free gesture tracking using acoustic signals. In *Proceedings of the 22Nd Annual International Conference on Mobile Computing and Networking*, MobiCom '16, pages 82–94, New York, NY, USA, 2016. ACM.

[10] T. Wei and X. Zhang. mtrack: High-precision passive tracking using millimeter wave radios. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, MobiCom '15, pages 117–129, New York, NY, USA, 2015. ACM.

[11] J. Xiong and K. Jamieson. Arraytrack: A fine-grained indoor location system. In *Proceedings of the 10th USENIX Conference on Networked Systems Design and Implementation*, nsdi'13, pages 71–84, Berkeley, CA, USA, 2013. USENIX Association.

[12] L. Yang, Y. Chen, X.-Y. Li, C. Xiao, M. Li, and Y. Liu. Tagoram: Real-time tracking of mobile rfid tags to high precision using cots devices. In *Proceedings of the 20th Annual International Conference on Mobile Computing and Networking*, MobiCom '14, pages 237–248, New York, NY, USA, 2014. ACM.

[13] S. Yun, Y.-C. Chen, W. Mao, and L. Qiu. Turning a mobile device into a mouse in the air. In *MobiSys*, 2015.