南京大学

本 科 毕 业 论 文

院　　系　匡亚明学院

专　　业　计算机科学与技术

题　　目　基于超声波和 IMU 的室内定位系统

年　　级　**2016** 级　学　　号　**161240005**

学生姓名　陈勇虎

指导老师　谢磊　职　　称　副教授

提交日期　**2020** 年 **5** 月 **20** 日

南　京　大　学

本科生毕业论文（设计）指导情况记录

<table>
<tr><td rowspan="2">开题简况</td><td>论文题目：</td></tr>
<tr><td>1、选题质量（简述选题与专业培养目标、专业要求关系、题目难度、工作量、创新性、理论性、实用性）

2、开题意见：

指导教师签名：
年　　月　　日</td></tr>
<tr><td>中期检查</td><td>指导教师检查论文的进展情况：（指导和培养学生查阅文献资料、综合运用知识、研究方案设计、研究方法和手段运用和外文应用等能力简况）

指导教师签名：
年　　月　　日</td></tr>
</table>

南　京　大　学

本科生毕业论文（设计）指导教师评阅意见

指导教师评语：

指导教师签名：

年　　月　　日

南　京　大　学

本科生毕业论文（设计）评阅教师评阅意见

评阅教师评语：

评阅教师签名：

年　　月　　日

南　京　大　学

本科生毕业论文（设计）答辩记录、成绩评定

答辩记录：

答辩记录人签名：

答辩小组评语：

答辩小组成员：_____

成绩_____　　组长签名：

答辩时间：　　年　　月　　日

南京大学本科生毕业论文 (设计、作品) 中文摘要

题目：基于超声波和 **IMU** 的室内定位系统

院系：匡亚明学院
专业：计算机科学与技术
本科生姓名：陈勇虎
指导老师（姓名、职称）：谢磊副教授

摘要：

基于 IMU 和智能手机的室内定位系统

关键词： IMU; 智能手机;

南京大学本科生毕业论文 (设计、作品) 英文摘要

THESIS: IMU

DEPARTMENT: School of Computer Science

SPECIALIZATION: Computer Science and Technology

UNDERGRADUATE: Yonghu Chen

MENTOR: Professor Lei Xie

ABSTRACT:

IMU

KEY WORDS: IMU, smart phone,

目　　录

第一章　绪论

1.1　研究背景及意义

在室内环境 (办公室，住所) 中想要寻找一个小物品，诸如钥匙，硬币，可以说是一件令人费神的事情了[1]。在游戏世界和虚幻的电影场景中，我们都或多或少接触过类似于“寻宝罗盘”的神奇宝物，它可以类似一个掌上的指南针，时刻指示“宝物”的方向。随着各式各样移动智能设备的广泛涌现，例如智能手机，智能手表，这些设备不仅自带了诸多传感器，同时可以为用户提供很好的 UI 界面。因此，如果可以利用这些设备实现类似于寻宝罗盘的功能，用于寻找我们室内环境中的小物品，无疑是有趣且富有创造性的。

随着对位置精度要求的提高，对于定位技术的研究也越加深入。传统的全球定位系统 (Global Positioning System,GPS)[2] 是现代定位系统的首选，然而，在室内环境中，由于卫星信号受到建筑物的削弱，遮蔽, 以及在复杂的室内环境下引起的信号反射，折射，透射等造成的多径和非视距传输现象, 导致定位误差较大，所以卫星定位的精度已经无法满足室内定位的需求，因此以无线传感网为媒介的定位系统逐渐发展起来。

1.2　国内外研究现状

1.3　研究内容与本文主要工作

1.

2.

1.4　本文的整体结构

本文共分为个章节，每个章节具体安排如下:

第一章绪论。

第二章相关工作

第三章实验

第四章总结与讨论

。。。

第二章　声源定位相关研究工作

2.1　声源定位技术介绍

噪声和异响是一种日常生活和工业生产中所常见的现象，在很多情况下，这些声音是令人困扰的。至于如何去解决这些噪声，我们首先需要去识别噪声并且能够定位出噪声的方向和位置，这就是声源定位问题的一个实例。声源定位技术就是在有噪环境中，去识别某一个声音的空间来源位置的技术，当然，声源定位技术并不只局限于噪声的定位，在特定的场合中，也有不同的应用。例如在会议室中，定位正在说话人的位置同样是声源定位技术的应用。

日常生活中我们似乎很容易根据听到的声音来辨别声源的大致方向，比如从身边急驶而过的汽车的来向，甚至车辆距离自己大概的距离。经过专业的训练，我们甚至可以让盲人参与到踢足球的运动中。人类之所以可以做到“听声辨位”，主要是利用了声音到达双耳的时间差，声级差，相位差，音色差等。这种常见的“听声辨位”尽管有一定的局限性，但是依然可以满足我们生活中很多的定位需求。声音定位技术通过将麦克风类比人耳，通过其接收到的声音数据确定声源的位置，借助于双耳效应的启发，目前的声源技术大多数都是围绕这四个参数进行研究，其中最常用的还是声音到达人耳的时间差。然而，在实际生活中，由于环境中的噪声，多径效应，混响等环境不可控参数的影响，精确的声源定位变得十分困难。

为了提高声源定位的分辨力，声源定位技术提出了很多高效的定位方案。本章将对声源定位方案等相关研究工作进行调研。

2.2　声源定位方案

第三章　智能手机感知与定位原理

近些年来，基于声学传感器的定位系统越来越受到人们的关注，依赖于多个同步的麦克风，采集到的多路同步信号，用以后续的信号处理。随着无线传感网的不断发展进步，无线传感网定位问题也是一个比较热门的课题。而节点定位作为目标定位的前提条件，节点的定位精度也影响着目标定位的结果。然而，尽管目前有很多节点定位算法，但是因为功耗，成本等客观条件的限制，这些定位算法在很多场景中并不能很好的应用。因此本文提出了一种可以在室内环境下利用智能手机进行定位的方案。

随着智能设备的普及以及诸多传感器的集成，智能手机与硬件技术的协同发展有目共睹。无论是 ios 系统还是 Android 系统下的智能手机，内置传感器也越加丰富，也为智能手机应用的发展带来了更多的可能。

本文将会对利用智能手机进行双麦结点定位进行简要描述，并对本文中后续所涉及的智能手机内置传感器进行讲解，最后系统描述本文中所使用的的一些定位原理和算法设计。

3.1　双麦克风的节点定位

随着可移动智能设备的普及，尤以智能手机更为常见。自 2018 年起，大部分的智能手机都开始集成两个甚至多个麦克风，除了底部的通话话筒以外，智能手机的顶部都还有一个“小孔”(不同的手机可能有所区别，例如 iPhone 手机的次麦克风可能位于主摄像头的附近)(如图 3-1)，这就是手机的降噪麦克风，它的主要作用是实现有噪声环境下的高质量通话。

手机上设有的两个电容式麦克风尽管作用不同，但是其性能相同，对声音信号的收集能力相同。利用双麦克风系统的时间同步 (两个麦克风是集成于同一个设备的，因此完全时钟同步) 特性，即使两个麦克风 (如图 3-2) 之间的距离较短 (16cm 左右，不同手机略有差异)，也可以根据声音到达两个麦克风时刻的差异计算 TDOA，因此，利用双麦克风系统手机结点作为实验设备有效解决了研究过程中时钟同步的问题。

目前已经有很多利用双麦克风结点实现声源定位或者其他有趣的研究成果。例如利用双麦克风智能手机进行手势轨迹感知[3];Aarab 使用双麦克风节点

图 3-1: iPhone8 的次麦克风手机位于摄像头附近

图 3-2: Dual-microphone phone

定位的思路测算说话人的位置[4]。

3.2 智能手机的传感器

随着技术的进步，手机已经不再是一个简单的通信工具，而是具有综合功能的便携式电子设备。大多数的 ios 和 Android 设备都有许多内置传感器，用来测量运动，屏幕方向和各种环境条件，这些传感器能够提供高度精度的原始数据，非常适合用来测量设备的三维移动和定位，或者检测设备周围环境的变化，例如可以跟踪设备的重力传感器推断用户的手势和动作，同样天气应用中将会使用设备的温度传感器和湿度传感器。智能手机的内置传感器有硬件实现

和软件实现之分[5]。基于硬件的传感器是内置在手机或者平板设备中的物理组件，这类传感器通过直接测量待定的环境属性 (如加速度，地磁场强度或者角度变化等) 来采集数据。基于软件的传感器不是物理设备，从一个或者多个硬件传感器获取数据，例如线性加速度传感器。具体可参考表3-1。本章将会简述实验中涉及使用的几种传感器。

表 3-1: Android 平台支持的传感器类型

传感器	类型	说明	常见用途
ACCELEROMETER	硬件	加速力	动态检测
AMBIENT_TEMPERATURE	硬件	环境室温	监测气温
GRAVITY	软件或硬件	重力	动态检测
GYROSCOPE	硬件	旋转速率	旋转检测
LIGHT	硬件	环境光级（照度	控制屏幕亮度
LINEAR_ACCELERATION	软件	加速力	监测单个轴向上的加速度
MAGNETIC_FIELD	硬件	环境地磁场	创建罗盘
ORIENTATION	软件	旋转角度	确定设备位置
PRESSURE	硬件	环境气压	监测气压变化
PROXIMITY	硬件	物体到屏幕的距离	通话过程中手机的位置
RELATIVE_HUMIDITY	硬件	环境的相对湿度	露点，绝对湿度和相对湿度
ROTATION_VECTOR	软件或硬件	设备	动态检测和旋转检测
TEMPERATURE	硬件	设备	监测温度

在传感器框架下采用三轴坐标系，其坐标系是相对于设备屏幕设定的，其设定如图3-3所示。在后续的陀螺仪，加速度传感器，地磁场传感器中，上述坐标系均试用。

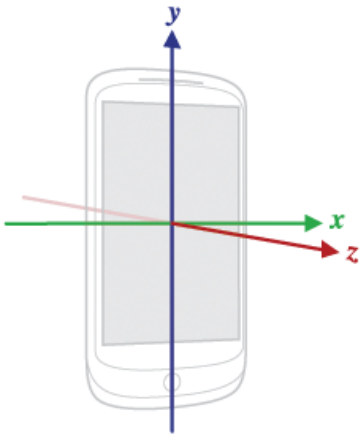


图 3-3: 传感器坐标系

3.2.1 陀螺仪

陀螺仪，是一种用于基于角动量守恒，用于感测和维持方向的装置。陀螺仪常用于导航，定位等系统。在 Android 系统中，陀螺仪测量的是围绕设备的 x，y 与 z 轴的旋转速率（弧度/秒）。通常情况下，我们利用陀螺仪的输出数据进行关于时间的一次积分，从而获得角度随时间变化累积的转角。但是，在判定设备是否“静止”的问题，对陀螺仪的输出设定一个硬阈值，当输出数据的 (绝对值) 超过阈值是，即可认为在设备处于运动中。从而避免积分中的漂移问题。

3.2.2 加速度传感器

加速度传感器又称为加速计，用于测量设备在运动状态下的加速度的传感器。高灵敏度的加速计用于飞机和导弹的惯性导航系统中，在无人飞行器中，加速计有利于保持稳定飞行。加速计的应用相当广泛，几乎所有的智能手机都包含加速计。尽管加速计的数据中包含了重力场的成分，但是在本文的使用中并不计较其细节，因此我们不对其进行深入解析。目前的智能手机下对加速计的应用很多，例如常见的计步器，微信“摇一摇”等。

3.2.3 磁力传感器

磁力传感器又称磁力仪，高斯计，是用于测量地球磁场的仪器。谈及磁力计，其主要的应用就是手机自带的指南针。在 Android 系统中，开发者不仅提供了获取磁力值的接口，同时也提供了磁力计与加速计结合获得方向的接口，因此，通过这方面的应用，将会给用户一个地理方位的指示。

3.2.4 屏幕方向传感器

方向传感器属于软件传感器，安卓平台提供方向传感器用于判断设备的位置。其数据是通过加速度传感器和磁场传感器共同获得的。至于其算法细节我们不必去关心。由于繁重的处理操作，因此方向传感器的准确度和精确度都是会降低的。具体而言，方向传感器在倾角为 0 时才可靠。根据官方文档的描述，Android2.2(API 级别 8) 已经弃用屏幕方向传感器，Android4.4W(API 级别 20) 已经弃用屏幕方向传感器类型。因此，Android 系统提供了另一种方法来来获取屏幕方向值，当我们注册好加速度传感器和地磁场传感器后，即可以通过

调用 getRotationMatrix() 方法和 getOrientation() 方法来计算屏幕方向值。屏幕方向传感器常应用 VR，AR，指南针等应用。

3.3 信号时延计算原理

智能手机的两个麦克风处于相同的环境下，因此采集到的的两路音频信号具有较强的相关性。当求解某一声源的位置时，声源信号到达两个麦克风的时延往往具有重要的作用。对于到达时延差 (TDOA) 的获取主要有两种方法。

第一种方法依赖于信号到达时间 (TOA) 的测量，通过 TOA 的差值来求得。TOA 对系统的时间同步要求就会很高，在信道传输特性相似的情况下，这种办法可以减少多径效应带来的误差。以二维空间为例，如图 (3-4), 根据 TOA 可以得到关于 TDOA 的方程如下:

$$\sqrt{(x-x_1)^2+(y-y_1)^2}-\sqrt{(x-x_3)^2+(y-y_3)^2}=c(t_1-t_3) \tag{3-1}$$

$$\sqrt{(x-x_2)^2+(y-y_2)^2}-\sqrt{(x-x_3)^2+(y-y_3)^2}=c(t_2-t_3) \tag{3-2}$$

由于依赖于 TOA 的值，微小的测量误差就会引起 TOA 较大的误差，进而可能导致 TDOA 的精度降低，因此在后续的实验中没有采用这种方式求解时延。

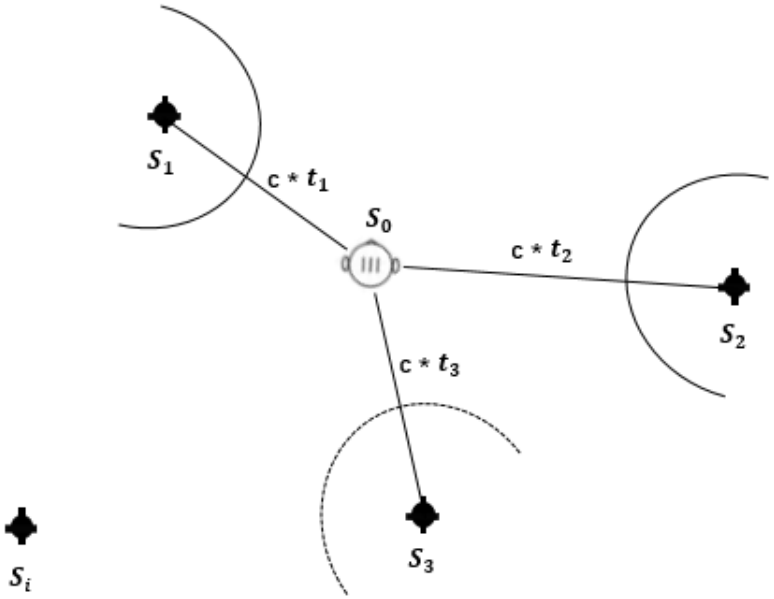


图 3-4: TDOA 算法示意图

第二种方法是将接受到的信号进行相关运算，从而得到 TDOA 的值。在实际应用中，系统往往很难做到严格的同步，因此常常用相关估计的方法求解 TDOA 的值，再进行后续的定位计算。在无线传感网中，这种方法对网络的要求比较低，因此更具有实际意义。

广义互相关的基本思想是对两路信号 $x_1(t)$ 和 $x_2(t)$ 进行预滤波，然后再求互相关函数，其原理框架如图 (3-5) 所示，公式表达为:

$$\tau_{\text{delay}} = \arg\max_{t\in\mathbb{R}}((f \star g)(t)) \tag{3-3}$$

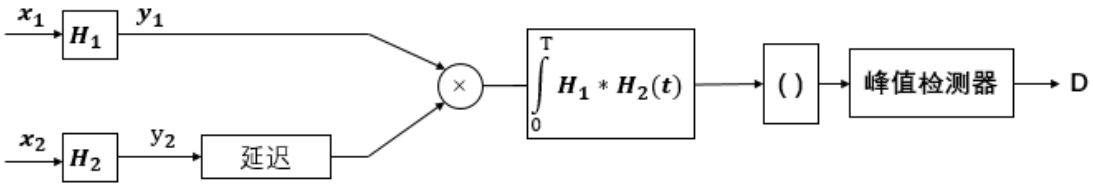


图 3-5: 互相关函数求解时延流程

在利用线性卷积计算时，常常可以借助线性卷积的关系来实现利用频域 FFT 的方式计算互相关，这种方法将比传统的时域计算法极大的减少计算量。根据维纳辛钦定理: 平稳信号的自相关函数与其功率谱密度互为傅里叶变换

对，因此相对应的，利用频域下的功率的互功率谱可以计算时域的互相关函数。最后通过峰值检测器求解出时延。需要说明的是，麦克风采集的信号是离散的，因此时延可以看做是几个采样点数的差距，通过麦克风的采样率，我们将可以把计算的采样点数的差距转换成时间的差值。

3.4 基于到达时间差的定位算法设计

上一节简要描述了利用互相关方法求解时延的方法，这一小节将会描述利用求解出的到达时间差 (TDOA) 进行后续的声源定位。

3.4.1 二维空间下的定位分析

我们首先从二维空间下进行定位分析。首先可以明确的是，根据到达时间差，结合声速 (v_{sound}) 我们可以求解出声源到两个麦克风的距离差 (Δd)，从而由方程 (3-4) 因此在二维空间下，相对于手机中心和手机坐标系，我们可以得到一个以智能手机麦克风节点为焦点的双曲线，而声源则在其中的某一支曲线上。同理，当我们移动手机至别的位置，那么就又可以得到一支曲线，如图 (3-6), 则两个曲线的交点就是声源的所在位置。

$$\begin{aligned}&\sqrt{(x-mic_{1x})^2+(y-mic_{1y})^2}-\sqrt{(x-mic_{2x})^2+(y-mic_{2y})^2}\\&=c\cdot\Delta t=d_1\end{aligned} \tag{3-4}$$

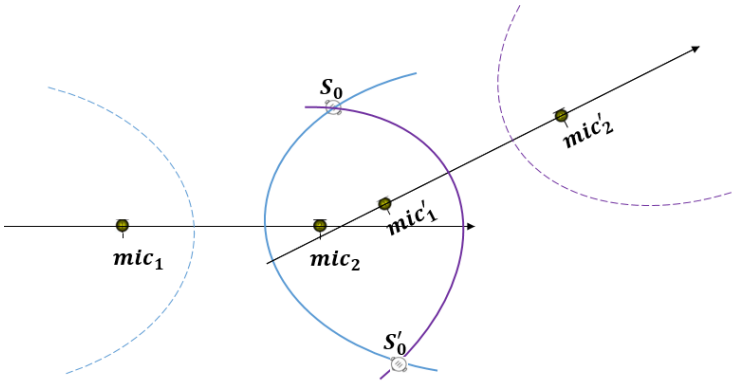


图 3-6: 2 维双曲线定位

3.4.2 三维空间下的定位分析

拓展至三维空间中，同样，我们可以根据到达时间差，计算出声源到两个麦克风的距离差，因此在三维空间中，如图 (3-7) 我们可以根据公式 (3-5) 得到一个双叶双曲面，而声源则可以在其中的某一个面上。同理，通过改变位置，我们可以获得更多的双曲面，进而求解出声源的所在位置。

$$\sqrt{(x-mic_{1x})^2+(y-mic_{1y})^2+(z-mic_{1z})^2}-\sqrt{(x-mic_{2x})^2+(y-mic_{2y})^2+(z-mic_{2z})^2} =c\cdot\Delta t=d_1 \tag{3-5}$$

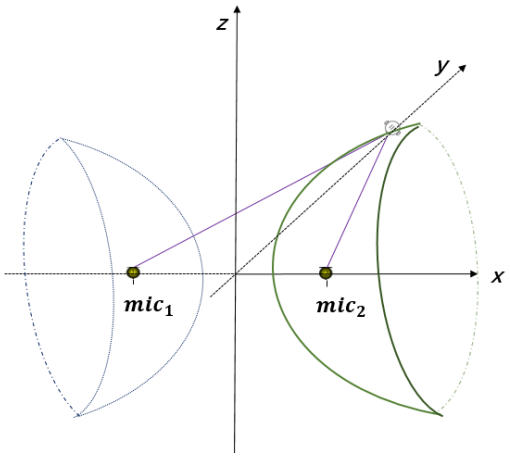


图 3-7: 3 维双曲面定位

3.4.3 离线计算下的定位算法分析和设计

通过公式3-4和公式3-5, 不难看出，定位方程组是非线性的，并且我们也不能保证到达时间差 (TDOA) 的测量总是不存在误差，另一方面，我们也知道声速也并非固定不变，正如公式3-6所描述的那样，因此对于到达时间差，总是存在或多或少的误差，这对后面定位精度的影响也是不可预测的。因此，由于我们可以变化智能手机的位置，因此就可以获得很多组数据，因此利用冗余的数据，该定位问题也就转换成了非线性的最优化问题。由于二维情况可以看做三维情况下的一种特殊情况，也就是 $z = mic_{iz}$，因此只对三维空间下的情况进行分析。

$$v_s = 331.3 + 0.606 * Temperature \tag{3-6}$$

为缩小搜索空间，在结点的布置上，采用固定其中一个麦克风结点的方式，如图 (3-8所示，因此根据到达时间差 (TDOA)，可以得到如下的方程组。

$$r_i^2 = (x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2 \tag{3-7}$$

$$r_0^2 = (x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2 \tag{3-8}$$

$$\Delta r_i = r_i - r_0 = c\Delta t_i, i = 1, 2, 3, ..., m \tag{3-9}$$

式中，r_i 为声源到不同位置 mic_2 结点的距离，Δr_i 为声源到固定的 mic_1 的距离，c 为声波的传播速度，Δt_i 为不同位置下 mic_2 与 mic_1 的到达时间差。

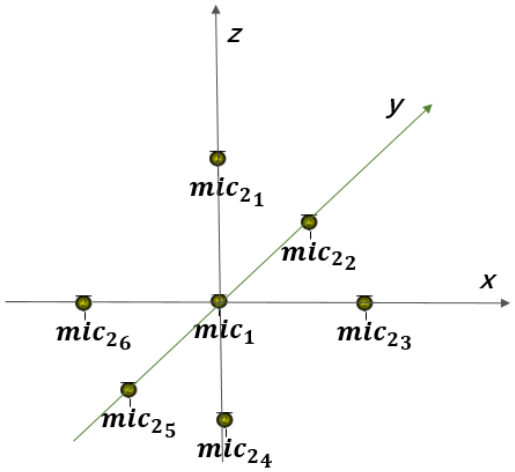


图 3-8: 3 维结点布置示意

下面简要分析在三维空间下，基于 Chan 的到达时间差定位算法[6]。由公式 3-7和 3-8不难得到:

$$\begin{aligned} R_{i,0}^2 &= r_i^2 - r_0^2 \\ &= (x - x_i)^2 + (y - y_i)^2 + (z - z_i)^3 - (x - x_0)^2 - (y - y_0)^2 - (z - z_0)^2 \end{aligned} \tag{3-10}$$

$$\begin{aligned} \Delta r_i &= r_{i,0} = r_i - r_0 \\ &= \sqrt{(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^3} - \sqrt{(x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2} \end{aligned} \tag{3-11}$$

$$\begin{aligned} r_i^2 &= K_i - 2x_i x - 2y_i y - 2z_i z + x^2 + y^2 + z^2 \\ K_i &= x_i^2 + y_i^2 + z_i^2 \end{aligned} \tag{3-12}$$

而由基本的定义则有:

$$r_i^2 = (r_{i,0} + r_0)^2 \tag{3-13}$$

将公式3-13代入3-12中，于是有

$$\begin{aligned} r_{i,0}^2 + 2r_{i,0}r_0 + r_0^2 = K_i - 2x_i x - 2y_i y - 2z_i z + x^2 + y^2 + z^2 \Longrightarrow \\ r_{i,0}^2 + 2r_{i,0}r_0 + (x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2 = K_i - 2x_i x - 2y_i y - 2z_i z + x^2 + y^2 + z^2 \Longrightarrow \\ r_{i,0}^2 + 2r_{i,0}r_0 = K_i - 2x_i x - 2y_i y - 2z_i z + 2x_0 x + 2y_0 y + 2z_0 z - (x_0^2 + y_0^2 + z_0^2) = r_i^2 - r_0^2 \Longrightarrow \\ r_{i,0}^2 + 2r_{i,0}r_0 = -2(X_{i,0} \cdot x + Y_{i,0} \cdot y + Zi, 0 \cdot z) + K_i - K_0 \Longrightarrow \\ X_{i,0} = x_i - x_0, Y_{i,0} = y_i - y_0, Z_{i,0} = z_i - z_0 \Longrightarrow \end{aligned} \tag{3-14}$$

将上式转换为矩阵形式

$$R_{i,0}^2 = r_i^2 - r_0^2 = -2\begin{bmatrix} X_{i,0} & Y_{i,0}i & Z_{1,0} \end{bmatrix}\begin{bmatrix} x \\ y \\ z \end{bmatrix} + K_i - K_0 \tag{3-15}$$

通过这种转化，消除了未知数的平方项，仅保留一次项, 从而由于 i 的不同取值，得到了一系列的线性方程组。

$$\begin{aligned} \frac{1}{2} * \begin{bmatrix} R_{1,0}^2 - K_1 + K_0 \\ R_{2,0}^2 - K_2 + K_0 \\ \cdots \end{bmatrix} &= -\begin{bmatrix} X_{1,0} & Y_{1,0} & Z_{1,0} \\ X_{2,0} & Y_{2,0} & Z_{2,0} \\ \cdots & \cdots & \cdots \end{bmatrix} * \begin{bmatrix} x \\ y \\ z \end{bmatrix} \\ h = \frac{1}{2} * \begin{bmatrix} R_{1,0}^2 - K_1 + K_0 \\ R_{2,0}^2 - K_2 + K_0 \\ \cdots \end{bmatrix}, G &= -\begin{bmatrix} X_{1,0} & Y_{1,0} & Z_{1,0} \\ X_{2,0} & Y_{2,0} & Z_{2,0} \\ \cdots & \cdots & \cdots \end{bmatrix}, h = Gz^0 \end{aligned} \tag{3-16}$$

其中 z^0 即为要求解的值。当然，由于误差的存在，所以 $Gz^0 \neq h$, 误差向量为 $\phi = h - Gz^0$。只需要 3 个 TDOA 测量 (计算) 值，就可以通过计算得到声源的

位置 (x,y,z)。但是充分利于冗余数据的话可以获取更好的拟合值。于是通过加权最小二乘法获得初始的估计解，再利用估计解和其他条件进行第二次加权平均，对得到的估计值进行优化。在[6] 中有对该方法的描述和分析，这一方面不是本文的重点，因此不做赘述。

3.4.4 在线指引下的寻迹方式

在前一小节中，我们简要描述了离线处理中，通过基于 Chan 的到达时间差算法计算出未知声源的方法。这种方法尽管操作方式复杂，但是却可以获得较为准确的结果。然而，在实际的定位中，在短距离的室内环境中，往往简单的方向指引就具有很好的参考意义。因此，对于三维的空间定位，在一定程度下，仅仅二维的指引就可以作为较好的参考值。

如图 (3-9) 中，我们获取了一个以麦克风为焦点的双曲线，同样，我们也就获得双曲线的渐进方向。而当声源距离麦克风中心较远时，渐进线的方向就是对声源方位的一个很好的指引。如果我们希望定位方式从离线状态下计算位置，转移到在线的指引我们去寻找的话，这种方向指引无疑是一种很好的途径。后文中，我们将对这个问题做具体描述。

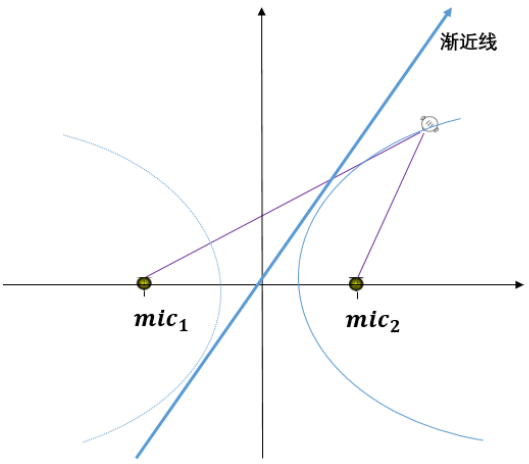


图 3-9: 2 维空间指引

第四章　基于双麦克风手机的声源定位方案

近些年来，智能手机已经成为人们生活中不可缺少的一部分，手机上集成的传感器的发展，使得智能手机的功能越来越强大。在前面我们已经简述到大部分智能手机已经配备两个麦克风，用于消除我们在日常通话中的环境噪声，并且可以很好的解决的时钟同步问题。因此，利用智能手机的优势，通过获取同一个手机两个麦克风的声音数据，我们从中过滤出我们想要的声音信号，得到两路同步的声音信号，利用智能手机计算出两个麦克风的到达时间差(TODA), 对计算出的 TDOA 信号进行后续的处理，就可以实现一些有意义的功能。

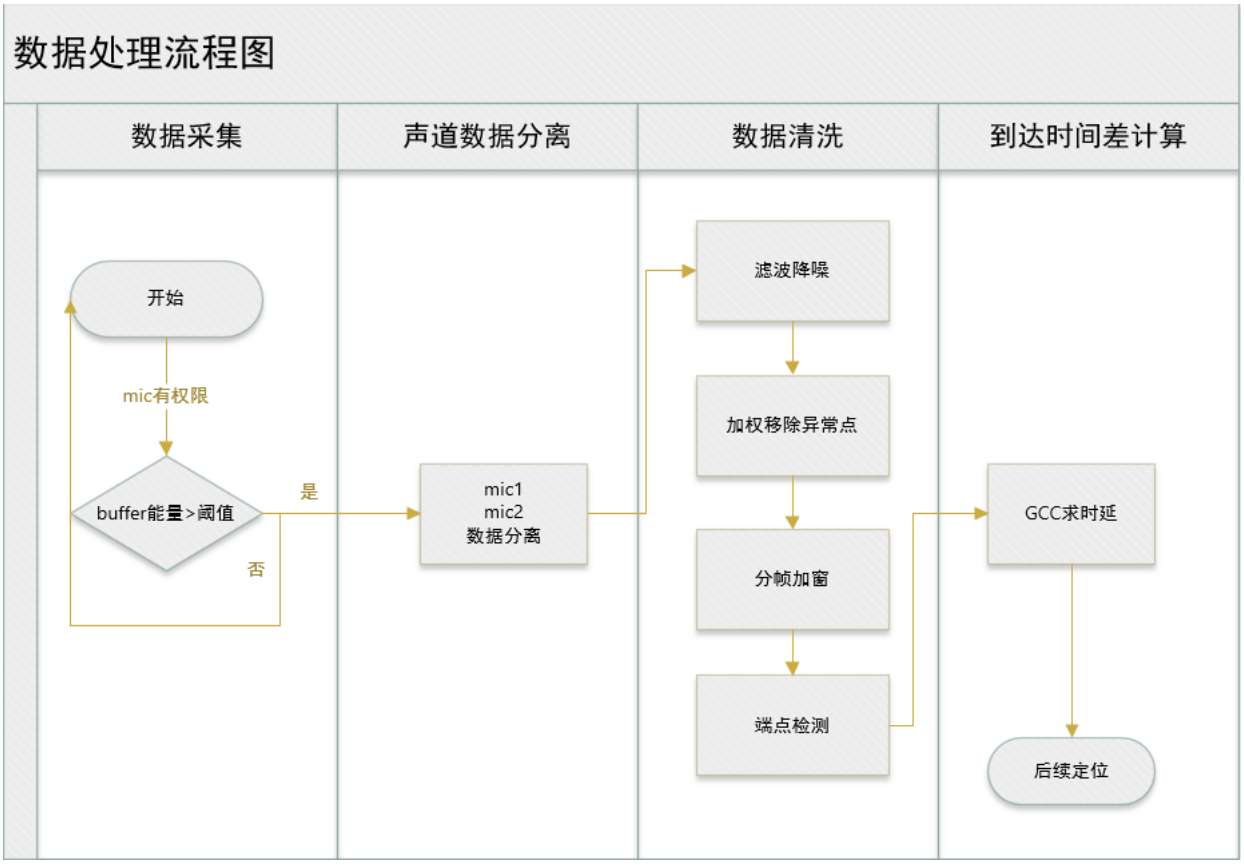
在前一章的基础上，本章详细描述对收集数据的预处理过程，并分析其特征。由于语音信号的非平稳性，因此预处理操作尤为重要。本章节将会描述利用双麦克风手机实现定位功能的诸多细节性工作，从而用于后续的定位分析。

4.1　离线数据预处理流程

本小节中，首先介绍离线状态下的数据预处理流程，并逐渐过渡到最终系统使用的在线运行的算法的处理流程。由于麦克风收集的信号是一种非平稳的带宽信号，因此需要对收集的信号进行预处理操作，否则，由于噪声等环境因素的影响，会使得实验效果很不理想。

在离线数据处理流程中，智能手机会收集声音文件并产生一个 wav 文件，通过 MATLAB 分析该 wav 文件，最终计算出到达时间差 (TDOA)。该离线数据算法处理流程是为了通过处理一段时间窗口内，设备没有发生大幅度状态变化 (包括角度，位置等参数) 的情况下，计算出静态声源相对于该位置的方向 (Speaker Direction Finding (SDF))。在分析阶段，包括滤波，加权移除异常点，加窗分帧，端点检测等必要操作，在一段时间窗口内的处理流程具体如下图所示 (4-1).

图 4-1: 离线数据处理流程



4.2 数据采集

本文利用具有双麦克风系统的智能手机 (Samsung 5) 作为传感器结点的进行声音信号的手机，在分析阶段，使用 MATLAB 科学工具对采集到的信号进行解析显示。我们利用另一台手机发出了一个频率在 2000Hz-2700Hz 变换的线性调频信号, 如图 4-2所示，图中蓝色的标记为 LY 的数据以及红色的标记为 RY 的数据，分别代指两个麦克风采集的数据。图中的上一部分图中，横坐标代指数据采样点，纵坐标是对数据进行标准化后的结果，因此无量纲。但是在下部分图中，横坐标是标准化的频率, 单位为 π*radians/sample(频率标准化的相关概念不做赘述，可以参考[7])，纵坐标为能量谱，这里我们很容易看出两个麦克风的采集的数据的能量谱是非常接近的，换句话说，两个麦克风的采集能力非常接近。这里我们对原始数据取样其中发生部分连续的 400 个点。如图 4-3所示，同样我们依然可以看出两个麦克风的采集能力是非常接近的，另一方面，即便我们暂时无法确定到达时间差 (TDOA) 的值，但是我们可以清晰看出两路声音数据是差了几个样本点的，因此只要计算出相差的样本点数，根据采样率

(44100Hz) 我们就可以求解出到达时间差 (TDOA) 的近似的结果。

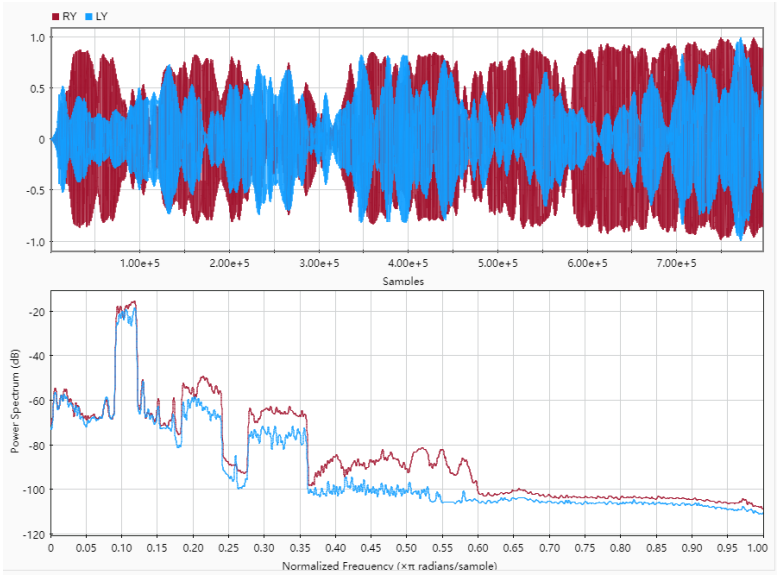


图 4-2: 双麦克风信号

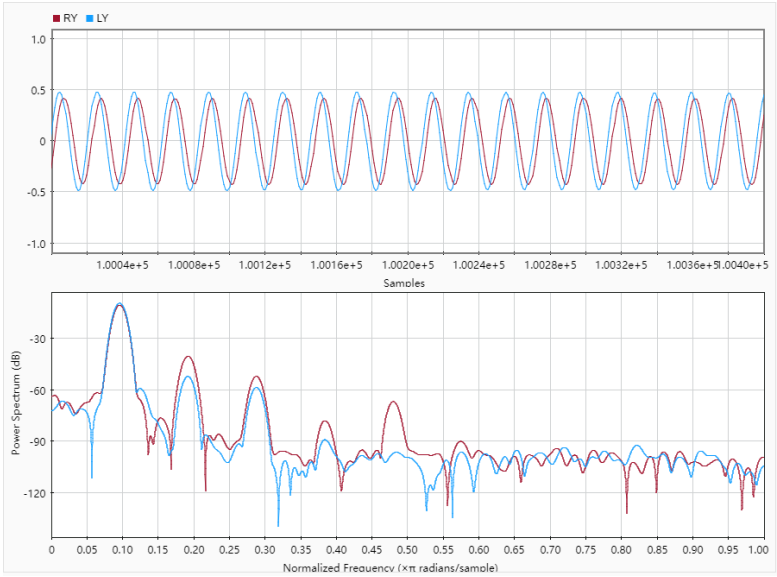


图 4-3: 双麦克风信号取样

下表中 4-1展示了采集声音信号的手机参数，实验中使用了 SamSung S5 手机作为实验设备，S5 集成了两个同步的麦克风，分别位于手机的底部和顶部，而这个两个麦克风的声音采集能力完全相同。目前的智能手机最高采样率都可以达到 44100Hz，每秒钟就可以获得 44100 个声音信号的样本值。

本文利用内置双麦克风的 S5 手机进行实时采集声音信号，在离线数据处理中，对信号进行汇总处理和分析。在底层获取的声音信号实际是个一维数组，当然利用 MATLAB 进行读取，会转换成两个一维数组，分别代表两个麦克风

表 4-1: SamSung S5 设备参数

参数	描述
灵敏度	-26dB
麦克风距离	16cm
采样频率	44100Hz

的数据。在原始的底层存储中，一维数组的值即是采集的到数据。如图4-4所示，实验中，我们采用 16bit 的 stero 的格式进行采集，因此根据图中的存储方式很容易分别获得两个麦克风的数据，并且两组数据是完全满足时钟同步的。为了环境噪声或者其他声音的干扰，本文中设置了一个基于能量的硬阈值，当每个 buffer 信号的能量低于该阈值时，则认为没有声源发生，当平均能量高于该阈值时，认为有声源发生，从而进行后面的处理。

图 4-4: 数据存储方式

4.3 声音数据处理

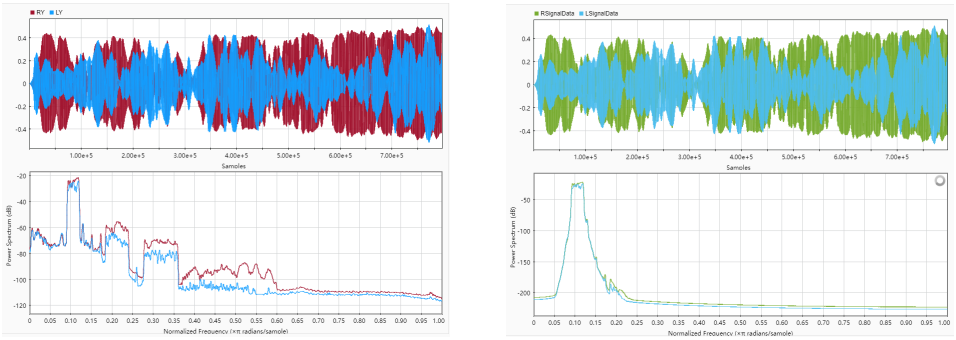
从底层获得的两路原始信号包括了真实声源信号，环境噪声以及其他各种异常点，本小节将会利用滤波器降低环境噪声的影响，同时使用加窗移动的方式去除采集中的异常值，对原始的信号进行清洗处理，后续将会进行分窗加帧，端点检测等为后续到达时间差做准备。

4.3.1 滤波除噪

由于信号采集中也会含有噪声成分，这对我们声源定位无疑会造成很大的影响，因此我们应该去除非声源发声的成分，提高声源的成分。常见的去噪方式就是使用滤波器，这里我们使用带通滤波器，降低环境噪声的干扰。在

MATLAB 通过 filter,designfilter 函数我们很容易实现我们需要的功能，在后续的在线算法处理流程中，因为程序运行在 Java 平台上，所以 Android 方面使用 dsp-collection 这一个 jar 包即可满足我们需要的功能。

例如当我们的声源频率范围在 2000Hz-2700Hz 时，那么在除噪的过程中，我们就需要通过一个带通滤波器，减益非该频率范围的成分。如图 (4-5) 所示，横纵坐标信息同图 (4-2)。通过滤波，不在制定频率范围的数据被大大减益，从而降低了对实验结果的影响。



((a)) 信号除噪之前 ((b)) 信号除噪以后

图 4-5: (a) 数据除噪之前。(b) 数据除噪以后。

4.3.2 异常点移除

对实验结果影响较大的还有异常数据，例如在稳定的信号中仍会有一些突变的声音信号。为防止信号之间的干扰，这里采用分别对两路信号进行处理。值得一提的是，由于采样率高达 44100Hz，所以处理过程中去掉一些数据并不会造成较大的影响。这里，我们采用滑动窗口的方式，使用加权平均移动 (WMA[8]) 的方式进行处理。加权平均移动认为: 同一个窗口内的不同的数据对目标值都有着不相同的权重因子，因此目标值就可以看成窗口内数据和一个固定的权重函数的卷积。由于数据本身异常点并不是很多，并且使用传统的WMA就可以获得较好的效果，因此记 signal_t 为 t 时刻的数据，采用下面的公式更新 t 时刻的信号值:

$$\begin{aligned}\text{signal}_{t_{new}} = \text{WMA}_t &= \frac{np_t + (n-1)p_{t-1} + \cdots + 2p_{(t-n+2)} + p_{(t-n+1)}}{n + (n-1) + \cdots + 2 + 1} \\ &= \frac{n \times \text{ signal }_{t-1} + (n-1) \times \text{signal}_{t-1} + \cdots + 1 \times \text{signal}_{t-n+1}}{n + (n-1) + \cdots + 2 + 1}\end{aligned} \tag{4-1}$$

经过上述处理后，数据变得更加光滑，并且从图4-6中已经可以隐约可见到达时间差 (TDOA) 的大致范围。这里，LSD为第一个麦克风降噪后是数据，LSignaData代表降噪后数据经过异常点去除操作后的数据。

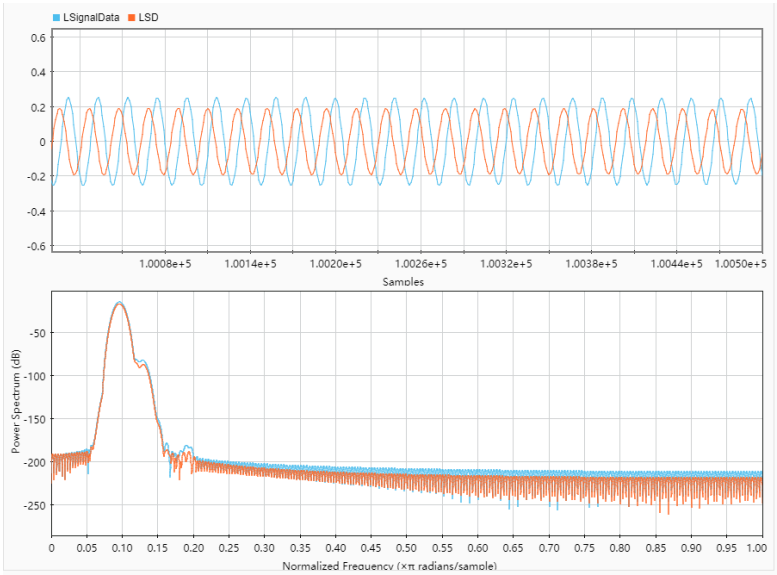


图 4-6: 异常点移除后的数据

4.3.3 分窗加帧

尽管音频信号是非平稳的，但是在极小的时间窗口内，例如几十毫秒的范围下，音频信号依然是比较平稳的。通过这个想法，可以把音频信号看做是一帧一帧的小时间窗口的信号，每一帧的信号都可以看成独立的声音信号。在这个意义下看，也就是在音频信号上加了一个滑动窗口，窗口的移动速度在窗口长度的三分之一左右即可。如图 (4-7)

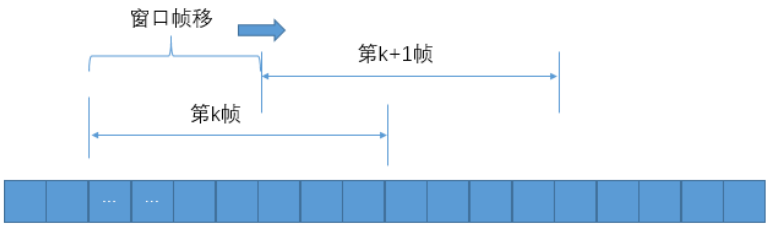


图 4-7: 数据分帧

由于在信号分帧中，基本上不可能做到周期性截断，因此会在时域上进行加窗以调制信号，减少频谱泄露。常见的窗函数有矩形窗，汉宁窗和汉明窗[9]。以 N 表示帧长，这两种窗函数的时域表达式如下所示。

(1) 矩形窗的时域表达式:

$$w(n) = \begin{cases} 1, 0 \le n \le N-1 \\ 0, n = else \end{cases} \tag{4-2}$$

(2) 汉宁窗和汉明窗的时域表达式:

$$w(n) = a_0 - (1 - a_0) \cdot cos(\frac{2\pi n}{N}), 0 \le n \le N \tag{4-3}$$

当 a_0=0.5 时，为汉宁窗，当 $a_0 = 0.54(\frac{25}{46})$ 时，为汉明窗.

对窗函数的性能区别我们不做过多描述，在制定频段的声音下，为了保持平滑，汉明窗的性能更优。

4.3.4 端点检测

尽管我们采集的信号是“连续”的信号，当时仍然有可能部分数据帧并不包含原始信号。因此可以在数据处理中通过端点检测找到有效的发生部分。前文数据采集中，我们对数据采集设定了一个基于能量的硬阈值，虽然短时能量可能出现噪声特征值较大，或者声音混淆的情况，但是在实际实验中，由于室内环境噪声较小，利用短时能量依然可以满足我们的需要。对这方面的优化依可以通过参考文献[10] 中的对数能量特征进行优化。

4.3.5 到达时间差计算

本小节基于前期的信号处理获取到两路同步的声音信号，调用传统的 GCC 算法对两路同步信号进行到达时间差 (TDOA) 计算 (估计)。对于信号时延计算原理在第三章已经说明，因此这里我们不再赘述其原理。

由于接收到的声音信号来自于同一个声源，因此两路信号之间具有较强的相关性而与噪声无关。广义互相关方法恰好可以得到两路信号之间的时延差。前文中我们也有说明，这里计算出的“时延”实际是两个信号相差的采样点数，但是，由于采样率为设定好的 44100Hz, 因此可以通过计算出相差的采样点数，经过转换从而可以得到对应的时延，也就是实际的到达时间差 (TDOA)，后面我们用相差的采样点数代指到达时间差 (TDOA)，转换公式无需赘述。可以明确的是，当 TDOA < 0 时，声音先到达第二个麦克风，TDOA > 0 时，声音先到达第一个麦克风，TDOA = 0 时，声音同时到达。降至二维空间下，详情

如图 (4-8) 所示。

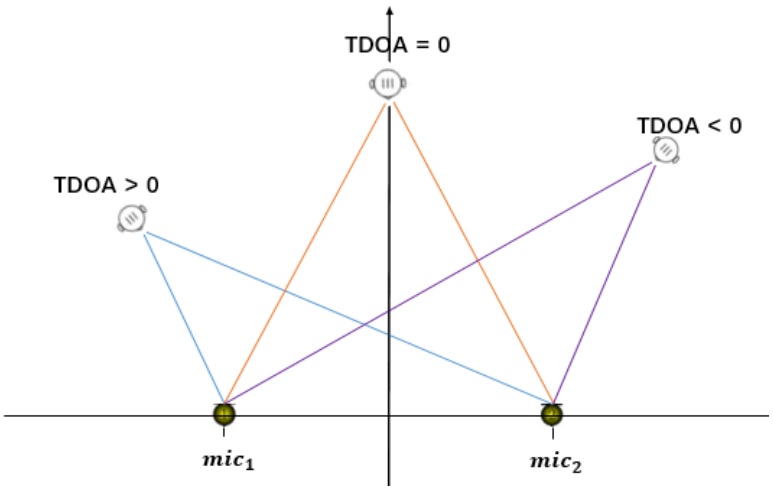


图 4-8: TDOA > 0, TDOA < 0, TDOA = 0

4.3.6 到达时间差处理

利用智能手机作为传感器，计算指定声源到达同一个手机麦克风的到达时间差 (TDOA)，可以得到大量的数据并且计算出很多个数据段的到达时间差 (TDOA)。因此鉴于数据量多的优势，那么我们就可以利用此优势进行数据的筛选。即便我们的某些到达时间差 (TDOA) 测量并不是足够“精确”，但是依然可以辅助进行方向探测。

回到最初的系统设置，我们设置的采样率 44100Hz，编码为 16bit，我们暂时不考虑高编码编码的精确度，但显然的是，由于使用的 bit 数较大，因此数据量化的范围也更大，因此数据也是更精确的。对于 44100Hz 的采样率，麦克风每手机一秒钟，就可以采样到 44100Hz 的采样点。实验中发现，如果将太多量的数据一次性进行计算，由于我们没有加窗等操作，加上声源功率较小，信号衰减等，大部分测量结果都是不准确的。然而，另一方面，即便数据过多造成了结果不够准确，但是对数据分段，对该一秒钟的部分数据进行 tdoa 计算，在原理上也是对整段数据的 tdoa 的估计，因为延迟是“全局”的延迟，通过实验我们也发现这一个问题。因此，鉴于这种思想，我们可以将信号进行分段，当然，由于在后续实验中设备会一直处理数据，因此我们并不是对 1s 数据分段，而是每 k 个连续数据进行分段。从而，对两个麦克风相同次序的数据段进行到达时间差 (TDOA) 计算，依然可以得到全局的到达时间差 (TDOA)，如图 (4-9)。

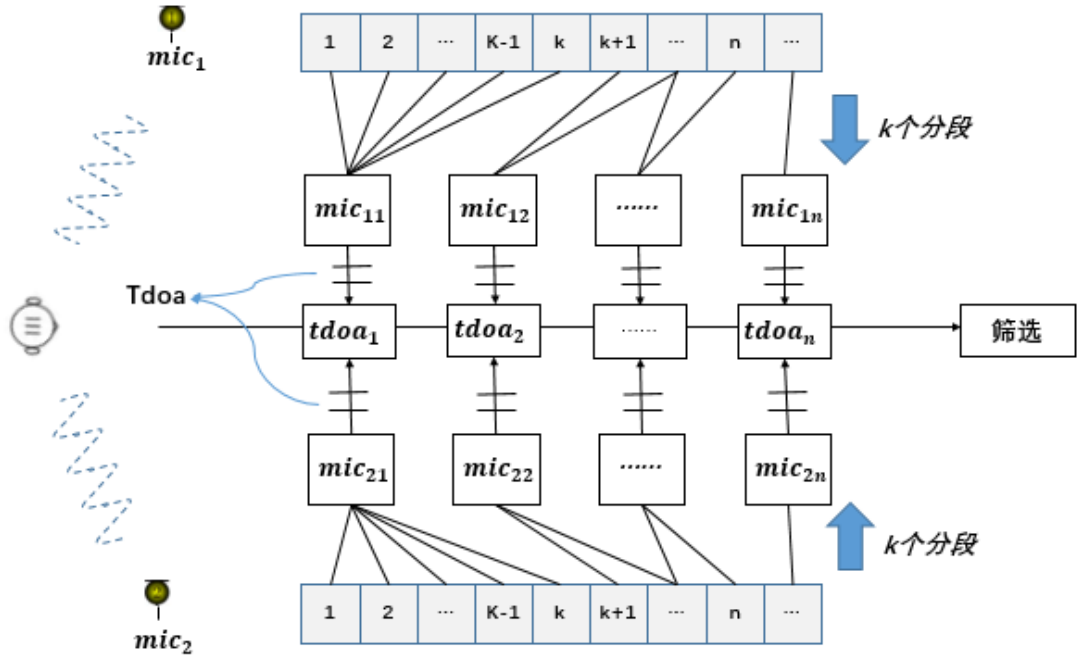


图 4-9: 数据分段示意

数据分段之后，并进行对应的数据段的到达时间差 (TDOA) 计算，从而可以产生一系列的到达时间差 (TDOA) 值。在实际计算中，我们设定 k = 2048，很明显的是，对 k 的设定是一个 trade-off 的问题，若 k 值太小，数据点太少，基本无法得到合理的到达时间差 (TDOA) 值，如果 k 值较大，精度会有影响，同时，也就没有分段的意义了，实验中 k 暂取用 2048。另一方面需要明确的时，实验中手机麦克风距离约为 16cm, 因此到达时间差 (TDOA) 绝对值的最大值 $|\text{TDOA}_{\max}|$ 应该满足下式：

$$|\text{TDOA}_{\max}| = 16/v * 44100 \leq 16/33700 * 44100 \approx 20.9 < 21 \qquad (4\text{-}4)$$

因此 Tdoa 的范围应该在-21 至 21 之间，这就是到达时间差 (TDOA) 合理值的区间。另一方面，虽然由于离散采样的问题，到达时间差 (TDOA) 的值最后应该取整数，但是我们也可以肯定的是，即便有 0.5 个采样点的误差，$\delta distance = 0.5/44100 * v \leq 0.5/44100 * 34000 \approx 0.38cm$, 对于 16cm 的麦克风距离来说，这个值微不足道。

前面中我们提及到，部分信号计算的 tdoa 应该符合或者接近于全局的到达时间差 (TDOA) 值，基于这种思想，我们使用以下的筛选处理方案 (以 TDOA 代指到达时间差):

(1) 每 m s 中，准确来说，应该是每 $\lfloor 44100 * m/2048 \rfloor * 2048$ 个数据，进

行一个到达时间差 (TDOA) 计算，在这一次 TDOA 的计算中，按照前文叙述的方法，每 k(k=2048) 个数据计算一次 TDOA 的计算，因此可以产生个 $\lfloor 44100 * m/2048 \rfloor$ 个 TDOA 的值。

(2) 对于这些 TDOA 的值，如果有超过一半的值相同并且介于合理值区间，那么我们就可以认定该 TDOA 有效，并且其符号包括数值都可以用于后续的处理算法中。

(3) 如果无法满足 (2) 的条件，但是大部分值甚至所有值都是大于 0 或者小于 0，或者等于 0，只有极少的符号与整体不一致，这里我们定义极少为占比不超过 1/7, 那么仍然可以认为该组 TDOA 有效，不过此时我们只使用 TDOA 的符号，也就是大部分值的的符号，用于后续的方位判断。对于特殊情况，我们考察这些数值的众数，如果只有一个众数，如果众数在“大部分值”占比超过一半，那么我们使用该众数作为 TDOA 的数值; 如果存在多个众数，并且数值差值不超过 2，并且所有众数在“大部分值”中的占比超过一半，我们可以使用这些众数的均值作为 TDOA 的数值使用，否则，在 (3) 中，我们只使用 TDOA 的符号。

(4) 如果 (2)(3) 的处理条件都无法满足的话，我们认为这一组的 TDOA 无效。

但是，当 TDOA 不为 0 的时候，信号能量会有较为明显的对比，因此对于符号的判断基本都可以保证，因此大部分测量结果至少可以返回一个合理的 TDOA 符号，因此大致方向可以确定。并且在离线处理中，取 m=1，就可以满足我们的实验需求。

4.4 离线处理下的定位方法

图 (4-8) 中，如果我们以 mic_1 为左，以 mic_2 为右，根据到达时间差 (TDOA) 的符号 (0,-1,1) 实际可以确定对于声源与中心关系，这里我们只简单定义为左，前，右三个方向。后续说明方位的时候，我们所指方向就是这里说明的左，前，后。

由到达时间差 (TDOA) 我们很容易确定一个双曲线方程，对于双曲线的渐近线方向，就是对声源的很好一个指示，至少在 2D 的范围中是很好的一个标准。这里我们根据到达时间差 (TDOA) 的值和麦克风距离，就可以计算出渐近线的角度 (方向)。后续说明角度的时候，我们所指角度就是这里说明的角度。

在室内环境 (图 4-10) 下，我们需要在整个空间坐标系下确定每次设备的所

处的位置，也就是说，已经确定了图4-10中 Position_i 的位置，随后我们在每个位置下录取一段音频，并进行到达时间差 (TDOA) 的计算，由于设备在三维空间中可以有很多的位置，因此，我们可以得到一系列的 Position_i，以及对应的 $X_i, Y_i, Z_i, TDOA_i$。我们利用已经比较成熟的前文所描述的 Chan 算法即可计算出后续的结果。

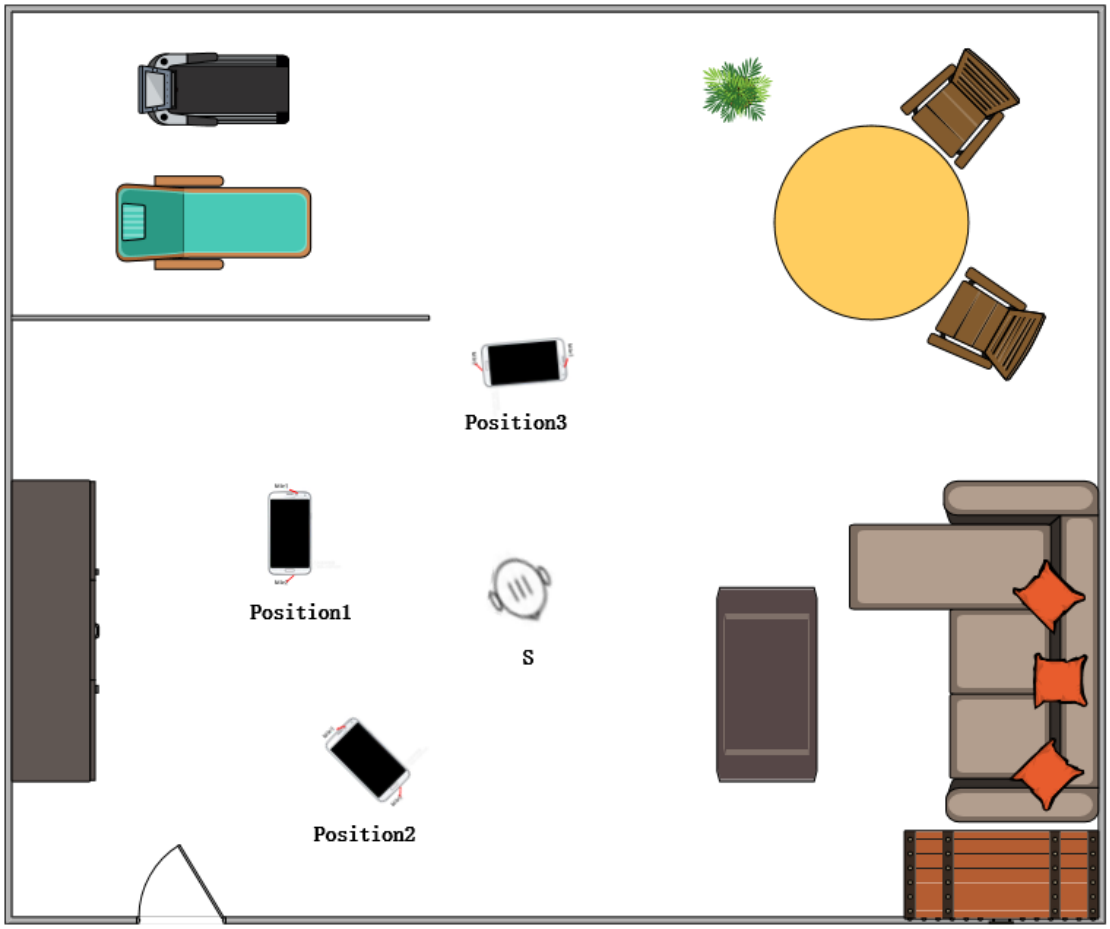


图 4-10: 室内环境

4.5 离线计算到在线指引的迁移

4.5.1 概念阐述

对数据的离线处理和定位计算尽管可以获得更为精确的结果，但是在实际操作中过于复杂，并且在实际寻找的过程中，我们也并不是“视而不见”，简单的提示信息或许就可以帮助我们的进行探测，因此我们降低了实际需求，从精确到 3D 指引降至 2.5D 的探测范围。因此在离线定位算法的处理的基础上，我们提出了数据在线的处理流程。

这里我们需要更新之前的处理流程，因为在数据在线处理流程中，麦克风处于一直的录音状态，因此有效声音提取部分也就不是那么重要，在前面的讲解中，我们指出无效声音段并不是声音信号的主体，因此可以直接抛弃一些数据。尽管数据的在线处理流程也可以看做多个离线处理的组成，但是在实际操作中，我们还是不希望在寻找过程中，在一个位置上长期停留，后续，我们引入一些机制进行一些状态的判断。

首先，我们需要提出一种降维的思想。如图4-11, 记智能手机屏幕所在平面为 P_1, 声源实际位置 S 与两个麦克风 mic_1,mic_2 所在平面为 P_2，记 P_1 和 P_2 构成二面角为 α, 在 P_2 中，声源位置 S 与智能手机中心的连线 l_1 与麦克风所在直线 l_2 夹角为 β, 声源在平面 P_1 的投影 S' 与智能手机中心的连线 l_3 与麦克风所在直线 l_2 夹角为 γ. 显然很容易得到 $\tan\gamma = \cos\alpha \cdot \tan\beta$. 图中 l_4 为平面 P_1 上与麦克风所在直线 l_2 夹角为 β 的直线。尽管 α 的值我们无法获知，但是计算出一个粗略的 β，依然有一定的指导意义，因为在平面 P_1 上，声源投影的位置 S' 的方向角 $\gamma = \arctan(\cos\alpha \cdot \tan\beta)$。

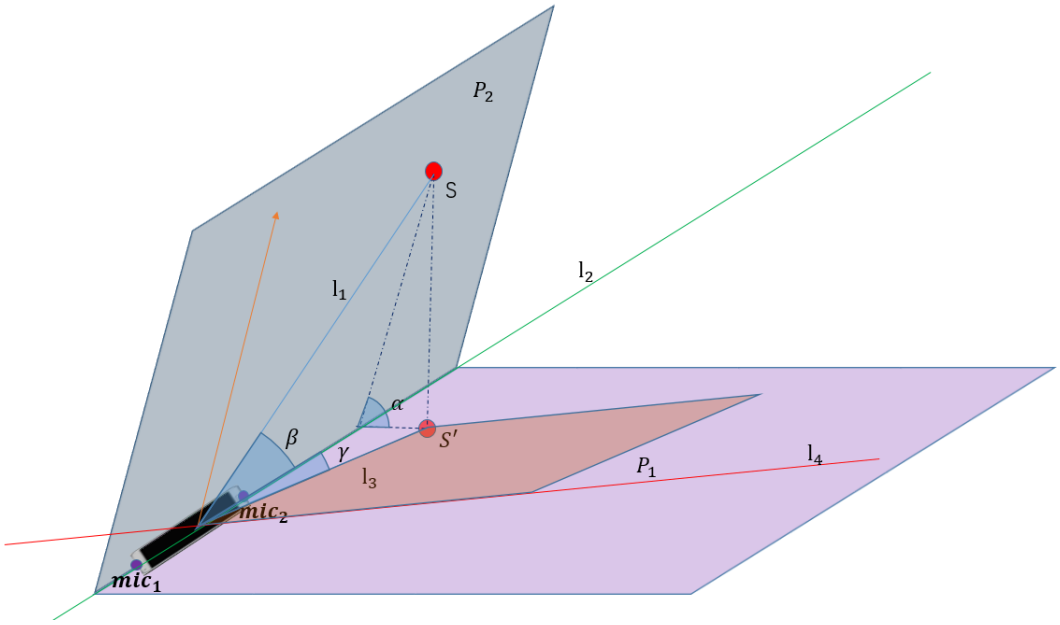


图 4-11: 3D 降维指引

如果在平面 P_1 上仍使用 β 作为指引的方向，那么，这里就会产生一个角度上偏离误差 $\Delta\theta = \beta - \arctan(\cos\alpha \cdot \tan\beta)$，显然当 α 越大时，误差 $\Delta\theta$ 越大 (这里 α 不超过 90 度)。显然在实际运行过程中，我们并不需要考虑这种情况。这是因为实际过程中，智能手机并非是固定不动的，因此，稍微的转动就可以使得二面角发生较大的变化。如图4-12所示，当手机位于 Loc_1 时, 二面角 α 达到 90

度，但是只要稍微转动至 Loc_2 处，就可以获得一个新的 α，甚至我们可以把手机移动到 Loc_3 处。当手机的转动已经不能造成二面角 α 较大的变化时，我们也就来到了声源投影位置的附近区域，这样，声源的位置我们也就大致获得了。

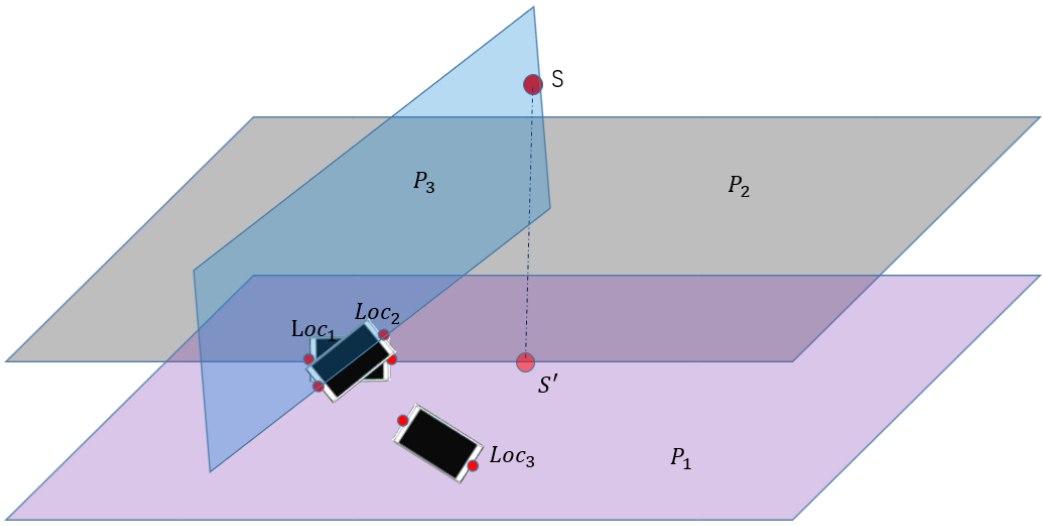


图 4-12: 手机状态改变

4.5.2 在线处理算法模型

在在线指引算法的模型下，我们需要引入一些机制，因为在在线指引的模型下，我们需要一个低要求的方向指引，而不再是准确的三维坐标。而且，在寻迹中，智能手机是动态移动的，而我们在计算到达时间差 (TDOA) 的过程中，必然是要保证麦克风位置时不变的。因此对于状态的判断就显得尤为重要。其次为了能够给予较好的方向指引，我们混合了地理方向等判断。

前文中我们提到了麦克风采样率为 44100Hz，因此在到达时间差和方向计算下，我们依然用滑动窗口的方式，窗口长度为 5s，而窗口移动的速度则设置为 1s，因此，第从 5s 开始后，系统将会每一秒反馈一个计算出的结果进行指引。

4.5.3 状态判别

在手机状态判别问题上，主要包括静止状态判定，声源位置方位判定。

一方面，在静止状态判定问题上，我们自然可以采用时域下某个指标积分后的结果进行衡量，但是由于 Android 自带的传感器必然存在温差，零漂的问题，因此，我们采用陀螺仪瞬时值进行衡量。

另一方面，在声源位置方位判断上，只是为了粗略估计声源是否发生了位置移动。在前文中，我们提及了声源相当于手机中心的“左右”位置关系，因此在在线指引的模型下，我们通过对“左右”关系进行检测，如图 (4-13) 所示，当我们等待过程中，如果系统反馈的声源方向在“左”(T_0')“右”(T_0) 关系上不恒定，那么我们将认为声源的位置在改变。尽管在后续实验中并不会出现这种情况，但是这是对实际应用下系统性能的优化。

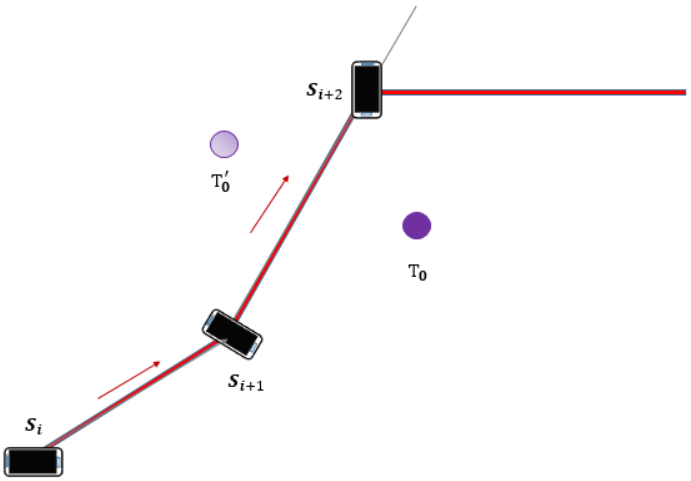


图 4-13: 状态判别

4.5.4 陀螺仪数据使用

首先我们需要确定手机处于“静止”状态，这里，尽管可以围绕麦克风所在直线旋转，但是在在线指引的模型下，我们不需要有这种操作，因此我们把这种状态也当做“非静止”的状态进行处理。以下给出一组围绕 X 轴依次旋转 90,180,270,360 度的采集到的数据，如图 (4-14)，分别表示三轴采集到的陀螺仪数据，横坐标为采样点，纵坐标陀螺仪数据，单位为弧度。从中可以看出，在设备绕 x 轴旋转的过程中，y,z 两轴同样会有数据的波动，但是，即便存在数据的波动，依然维持在 0 的附近，而发生运动的过程时，陀螺仪的数据就会有很大的波动，因此为了实际的应用场景，我们对三轴设定一个硬阈值，当陀螺仪数据 (的绝对值) 超过阈值时，则判定为设备发生了运动。

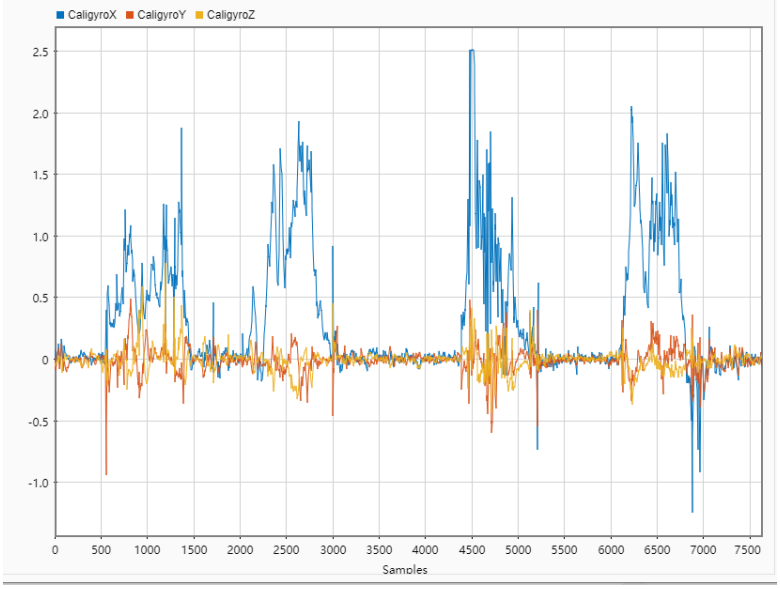


图 4-14: 陀螺仪三轴数据

4.5.5 加速计和磁力计的使用

随后，为了辅助与实际方向结合，因此借用 Android 的加速计和磁力计，其实现细节不做过多描述。当我们在 Android 设备下注册了加速计和磁力计两个传感器后，就可以通过对 SensorManager 调用 getRotationMatrix 以及 getOrientation，即可获得屏幕朝向与磁极的夹角，从而获取当前的方向，因此在过程中，并没有对加速计和磁力计数据的直接使用。

第五章　模型系统实现与实验分析

本系统是对前文所述的基于双麦克风手机的声源定位方案的实现，系统包括在线指引功能，离线计算等。离线计算的呈现方式是采取手机录音和 matlab 计算相结合的方式，在线指引的方式是制作成一个 Android 系统下的智能手机应用。此外，在声源方面，也是使用了一个自制的 Android 系统下的智能手机应用，该应用可以发出指定的声波，用于区分声源。

5.1　开发环境与平台

1. 硬件条件:
   1.1. 声源设备使用的是: 红米
   1.2. 录音设备使用的是: 华为 Mate20
2. 系统平台:
   2.1. 录音软件，声源发生软件都运行在 Android 系统下,
   2.2. 离线计算相关程序运行在 Windows 环境中，以 Maltab 程序的方式呈现
3. 开发工具:
   3.1. 手机应用开发平台
      3.1.1. Android Studio, 版本号:2.3.3
      3.1.2. JRE: 1.8.0_112
      3.1.3. JVM: OpenJDK 64-Bit Server VM by JetBrains s.r.o
   3.2. Matlab 版本:9.4.0.813654 (R2018a)

5.2　系统设计

在系统实现上有两种版本，正如前文描述的那样，一个版本是通过在指定位置下进行录音并保存音频为 wav 文件，然后通过将录音位置和波形文件导入的 matlab 中进行线下的计算，以定位声源的位置；另一个版本则是在手机屏幕上实时进行录音和方向反馈，通过不断的方向指引帮助我们去寻找声源。由于第一种版本除了录音阶段在智能手机上运行，其他阶段在都在 matlab 上运行，

而且两种版本实现方式有一定的重合部分，因此会在一起进行介绍，尤以第二个版本为重。

5.2.1 多模块设计

在系统运行中，各个模块功能的并行运行是突出的。因此对在线指引系统中，系统涵盖的模块可以分为: 传感器监听模块，录音模块，数据处理模块，显示模块等。

传感器监听模块:Android 的开发者已经对传感器提供了很好的接口，系统可以通过实例化一个 SensorManager 对象对传感器进行监听和数据读取。在系统中注册了加速计，磁力计，陀螺仪三个传感器，并且实时监听器数据。系统运行时，从 SensorEvent 对象中读取传感器数据，在重写 onSensorChanged 方法中，根据数据进行相应的处理。

录音模块: 需要说明的是，Android 6.0 以后为了用户隐私，将一些权限的申请放在了应用运行的时候去申请，所以申请麦克风权限也需要进行动态申请。当获取麦克风权限后进行录音，该模块会将录取的数据进行声道的分离，并将数据拷贝用于后续到达时间差的计算。

数据处理模块: 该模块主要用于对录音数据进行到达时间差的计算，并且反馈指引所需的角度和状态判别。通过对指定长度的录音数据进行截取，先后经过滤波处理等清洗操作，最后调用自定义的 Correlation 进行计算。在滤波操作中需要借助 dsp-collection 这一个 jar 包的帮助。最后计算的结果会传入到显示模块。

显示模块: 显示模块通过传入的结果进行方向的绘制，主要依赖于 graphics 库进行界面的绘制。该模块的显示界面包括了方向指示，同时也会和地理方向相结合，方便用户寻找方向，相关的数据会反馈在界面上。

此外，对于发声的设备，主要就是通过捕获用户的输入，从而产生相应的声波，由于实现难度并不是很大，所以不做过多的介绍。在前面所描述的系统核心和数据处理流程简图如下图 (5-1) 所示。

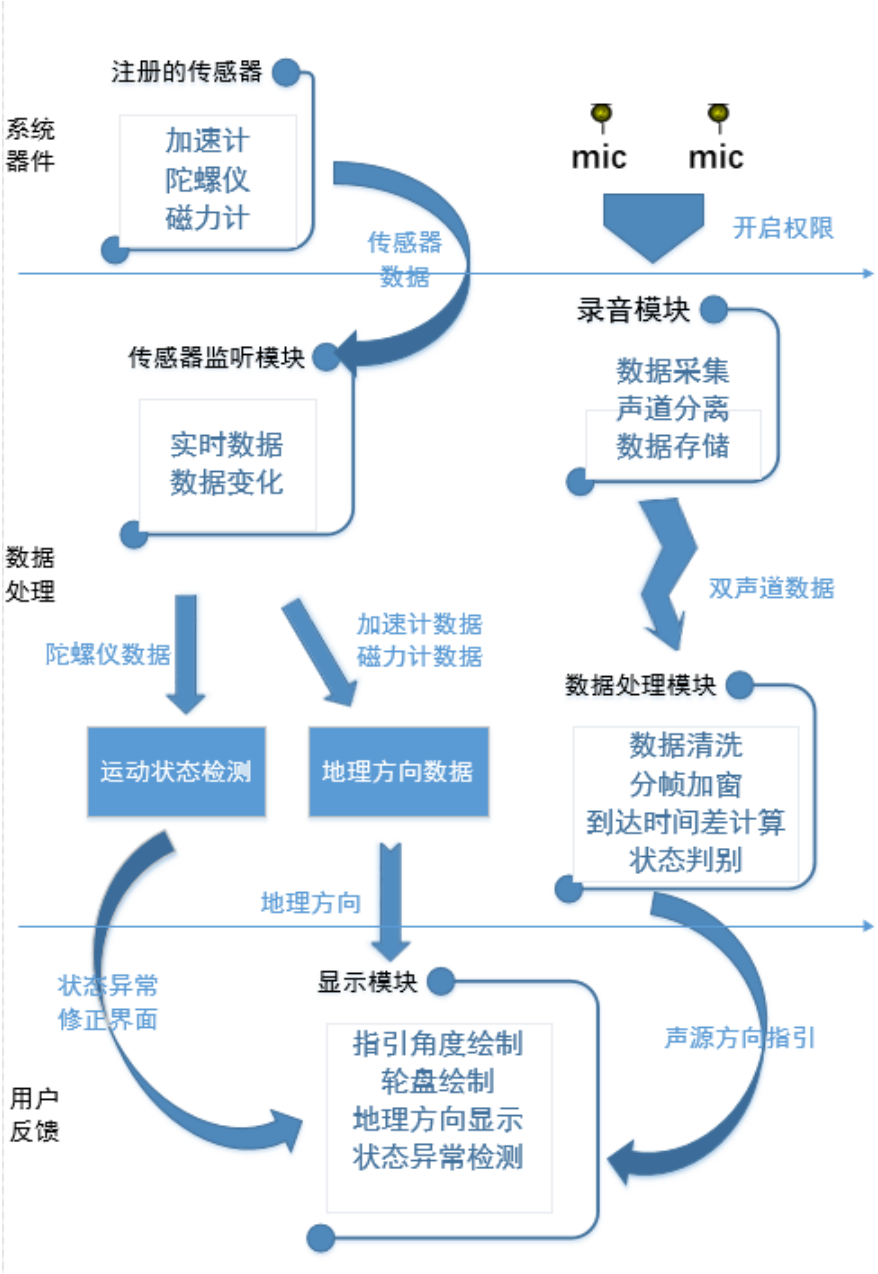


图 5-1: 系统流程和数据流简图

5.2.2 界面与运行方式

下面给出系统给出的实时指引的应用界面，具体如下图 (5-2, 5-3) 所示。

简要介绍应用功能，在图 (5-2) 中，通过选择波形和对应的频率数据，点击 PLAY 键将可以单声道播放对应的音频, 点击 PAUSE 即可暂停播放。

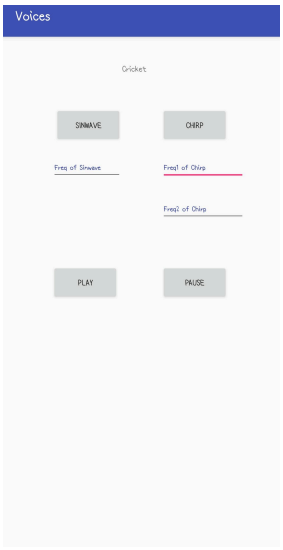
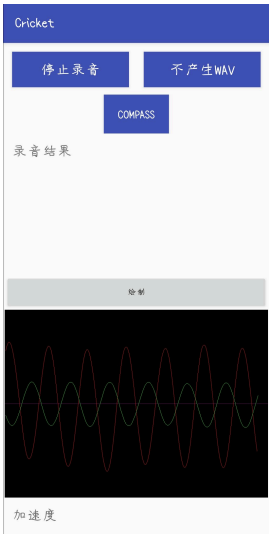


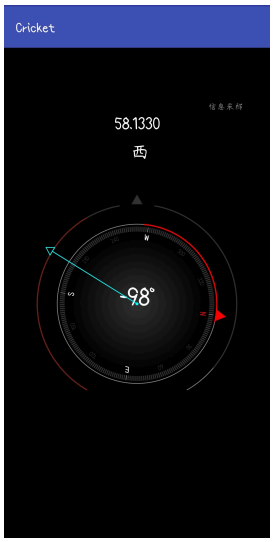
图 5-2: 发声 app 界面

在图 (5-3(a)) 中，通过选择录音，并且选择是否产生 wav 文件进行录音，期间可以绘制前一秒中部分的数据波形，在此界面下通过选择“产生 wav 文件”生成相对应的录音文件并保存在手机存储中，通过多个位置的操作，从而可以获得一系列的 wav 文件，这些文件将应用于后期的离线计算。

在在线指引的模型下，如图 (5-3(b))，系统会默认打开麦克风并开始采集数据，当手机状态“静止时”才会收集有效数据，当产生一定量的有效数据后，系统后台将会计算出目前声源相对应手机中心的方向，也就是图中的青绿色线，并且显示与手机 y 轴的夹角与当前的方向 (如图中”西” 所示)，轮盘则是对应的地理方向，从而方便我们寻迹。



((a)) 离线采集数据界面



((b)) 在线指引界面

图 5-3: (a) 离线采集数据界面，(b) 在线指引界面

5.3 系统功能实现

5.3.1 录音模块功能

数据采集阶段核心代码如算法5.1所示. 手机调用麦克风不断检测周围环境的声音信号，当信号端的平均能量高于阈值时，系统判断声源发声，并且将两路的声音信号进行分离，用于后续的处理。

算法 5.1 dual-microphone signals acquisition(双麦克风信号采集)

```
1:  mAudioRecord.startRecording()
2:  //系统检测周围环境中的声音信号
3:  while isRecording do
4:    //读取两路声音信号
5:    read = mAudioRecord.read(mBuffer, 0, BUFFER_SIZE)
6:    if read <= 0 then
7:      return false
8:    else
9:      //根据设定的阈值判断是否有声源发声
10:     sum = 0
11:     for i = 0 to read do
12:       sum += Math.abs(mBbuffer[i])
13:     end for
14:     //达到设定的阈值，认为有声源发声
15:     if sum / read > thresholdValue then
16:       //两路信号分离
17:       index = 0
18:       for i = 0 to mBuffer.length/2 do
19:         if i % 2 == 0 then
20:           System.arraycopy(mBuffer, i * 2, mBuffer1, 0, 2)
21:           LY[index] = ((mBuffer1[0]&0x000000FF) | (((int)mBuffer1[1])«8))/
              32768.0
22:           LYList.add(LY[index])]
23:           index = i / 2
24:         else
```

```
25:              System.arraycopy(mBuffer, i * 2, mBuffer2, 0, 2)
26:              RY[index] = ((mBuffer2[0]&0x000000FF) | (((int)mBuffer2[1])«8))/
                 32768.0
27:              RYList.add(RY[index])]
28:            end if
29:          end for
30:        end if
31:        //other code for further data-processing
32:     end if
33:  end while
```

5.3.2 数据处理模块功能

通过麦克风采集获取的数据，将要经过滤波等清洗后并分帧进行计算。其处理流程如伪代码5.2所示。

算法 5.2 data-processing(数据处理功能)

```
1: //数据清洗
   1.1: //filter(dsp) 滤波
      LSignalData = filter(bpFilt,LY)
      RSignalData = filter(bpFilt,RY)
   1.2: //RemoveOutliers 异常点移除
      LSD = RemoveOutliers(LSignalData,m)
      RSD = RemoveOutliers(RSignalData,m);
2: //分帧加窗
   2.1: LY_enframe = my_enframe(LSD, frameLen, overLoap, 'rect')
   2.2: RY_enframe = my_enframe(RSD, frameLen, overLoap, 'rect')
3: //端点检测
   3.1: [VSDst,VSDed] = myVSD(LSD,RSD,Unit,m)
   3.2: VLSD = LSD(VSDst:VSDed)
   3.3: VRSD = RSD(VSDst:VSDed)
4: //到达时间差计算
   4.1: //状态检测
   4.2: //调用自定义的 Tdoa 函数对两路信号进行计算
```

```
    4.3: [TDOA,Sign] = myTdoa(VLSD,VRSD,N,"True")
  5: //方向角度计算与显示
```

5.3.3 传感器监听模块功能

通过注册和监听传感器，并实时获取传感器数据进行处理。其处理如伪代码 (5.3) 所示。

算法 5.3 sensor-listering(传感器监听)

```
1: //传感器注册
2: //传感器管理器对象
3: mSensorManager = (SensorManager)getSystemService(SENSOR_SERVICE)
4: //传感器对象
5: mGyroscope = mSensorManager.getDefaultSensor(GYROSCOPE...)
6: mAccelerometer = mSensorManager.getDefaultSensor(ACCELEROMETER...)
7: mMagneticField = mSensorManager.getDefaultSensor(MAGNETIC_FIELD...)
8: //传感器监听器注册
9: mSensorListener = new SensorEventListener(){...}
10: //监听器方法重写
11:    //陀螺仪数据
12:    gyroscopeValues = event.values
13:       //设备动态监测
14:       IsMove(gyroscopeValues)
15:    //加速计数据
16:    accelerometerValues = event.values
17:    //磁力计数据
18:    magneticFieldValues = event.values
19: //获取方向
20:    getRotationMatrix(rotation, null, accelerometerValues,magneticFieldValues)
21:    getOrientation(rotation, orientation)
22:    //获取方向，并更新界面
23:    chaosCompassView.changeVal(orientation[0],degree)
24: //传感器监听
25: mSensorManager.registerListener(mSensorListener, mGyroscope,...)
```

```
26: mSensorManager.registerListener(mSensorListener, mAccelerometer,...)
27: mSensorManager.registerListener(mSensorListener, mMagneticField,...)
```

5.3.4 显示模块

显示模块通过在界面已经展示，由于其实现细节主要就是对 graphics 库应用，与本文主要内容无关，因此在此不做描述。

5.4 系统性能测试和分析

本文工作的过程伴随着大量的数据分析和实验验证，下面对基于完整的双麦定位实验给出数据展示和分析。

第六章　总结与展望

6.1　工作总结

6.2　前景展望

参考文献

[1] HONGZI Z, YUXIAO Z, ZIFAN L, et al. HyperEar: Indoor Remote Object Finding with a Single Phone[J]. IEEE 39th International Conference on Distributed Computing SysTems(ICDCS), 2019.

[2] Wikipedia contributors. Global Positioning System[EB/OL]. Wikipedia, The Free Encyclopedia, 2020 (2020/4/2) [2020/5/20]. https://en.wikipedia.org/wiki/Global_Positioning_System.

[3] KE S, TING Z, WEI W, et al. VSkin: Sensing Touch Gestures on Surfaces of Mobile Devices Using Acoustic Signals[J]. ACM MobiCom, 2018.

[4] AARABI P. The Fusion of Distributed Microphone Arrays for Sound Localization[J]. EURASIP Journal on Applied Signal Processing, 2003, 2003(4): 338–447.

[5] Google Inc. Android Developers Sensors Overview[EB/OL]. 2020 [2020/5/20]. https://developer.android.com/guide/topics/sensors/sensors_overview.html.

[6] 李招华，汪毓铎，邵青. 基于 Chan 的 TDOA 三维定位算法 [J]. 现代电信科技, 2014, 44(11): 36–40+4.

[7] Wikipedia contributors. Normalized frequency[EB/OL]. Wikipedia, The Free Encyclopedia, 2020 (2017/7/16) [2020/5/20]. https://en.wikipedia.org/wiki/Normalized_frequency_(unit).

[8] Wikipedia contributors. Moving average[EB/OL]. Wikipedia, The Free Encyclopedia, 2020 (2020/2/19) [2020/5/20]. https://en.wikipedia.org/wiki/Moving_average.

[9] Wikipedia contributors. Window function[EB/OL]. Wikipedia, The Free Encyclopedia, 2020 (2020/4/18) [2020/5/20]. https://en.wikipedia.org/wiki/Window_function.

[10] 肖述才，王作英. 端点检测中的一种新的对数能量特征 [J]. 电声技术, , 2004(6) : 37 – 41.

致　　谢

非常感谢在 Dislab 实验室度过的两年时光，让我在实验室里拓宽眼界的同时，也不断纠正了自己的学习方法和学习态度。我的毕业论文撰写和系统实现中，也或多或少的遇到许多困难，也走过很多弯路。我很感谢我的导师谢磊副教授，从我毕设选题阶段，到我研究工作的开展，再到最后的论文撰写中，谢老师都给予了我细心的指导和点拨，给我纠正了很多错误的方向，避免我走了很多弯路。在实验室的两年，谢磊老师也给我的研究工作不断指引方向，在此，我向谢磊老师致以最真挚的谢意。

同样，我也要感谢在 Dislab 实验室的同一课题组的各位师兄师姐，鲁欣然师兄在我的成长道路上指引了很多，从细节方面给予我帮助，王楚豫师兄同样，给了我研究工作上的很多帮助，让我的研究工作变的富有创新，也是他们给我提出的宝贵建议，让我可以不断改进自己。

感谢大学四年来四种相伴的同学和朋友，因为你们的存在，我的大学才会在忙碌中透露出精彩。感谢每一位悉心教导我的老师，感谢钱柱中教授的指引，让我能够去香港游学增长知识和阅历，让我在问题求解的课程中感受到计算机和算法的魅力。