北京郵電大學

本 科 毕 业 设 计（论文）

题目: 基于 Android 平台的可见光测距定位软件的研究与实现

姓　　名　朱明晖

学　　院　软件学院

专　　业　软件工程

班　　级　2013211502

学　　号　2013212049

班内序号　01

指导教师　赵方

2017 年 6 月

北 京 邮 电 大 学

本科毕业设计（论文）诚信声明

本人声明所呈交的毕业设计（论文），题目《基于 Android 平台的可见光测距定位软件的研究与实现》是本人在指导教师的指导下，独立进行研究工作所取得的成果。尽我所知，除了文中特别加以标注和致谢中所罗列的内容以外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得北京邮电大学或其他教育机构的学位或证书而使用过的材料。

申请学位论文与资料若有不实之处，本人承担一切相关责任。

本人签名：_____ 日期：_____

基于 Android 平台的可见光测距定位软件的研究与实现

摘　要

如今，随着科技的飞速发展，智能生活已经不再是一个遥远的概念。各种智能设备和服务层出不穷，为人们的生活提供便利，帮助人们解决难题。基于位置的服务是智能服务的重要部分，而智能手机是人们使用最广泛的智能设备，因此通过智能手机获取用户的位置信息是非常重要的研究方向。

在户外环境下，GPS 的定位技术已经十分成熟，足以应付大多数应用问题。但是在室内 GPS 信号较弱的环境下，精准定位问题仍然需要更好的技术来解决。现有的其他室内定位技术都存在着一些弱势，例如基于 Wi-Fi 的定位技术精度较低，基于红外线的定位技术成本较高。然而基于可见光的定位技术的定位精度可以达到 0.1m，易于部署，且成本低廉。

本文对可见光的测距定位进行研究，选用 LED 灯作为光源，并选择 Android 平台进行相关软件的开发和实现。本文首先比较了现有的各种室内定位技术的优缺点，介绍了相关硬件、模型以及技术的概念和特点。然后按照软件工程的流程，根据这些特点分析本定位软件的需求，对本软件进行设计和实现，最后对软件进行测试和评估。

本文设计的可见光测距定位软件主要分为两个定位模式：多灯测距定位、计步器与单灯校准融合定位。其中多灯测距定位利用光信道模型以及三边定位算法来进行定位；计步器与单灯校准融合定位则是利用计步器和航向角传感器得到步数和方向，再设置一定的步长，进而推算用户的运动轨迹和位置，再利用对光信号的检测和信标的坐标信息来进行校准。

本软件功能已全部实现，并进行了测试和评估，结果表明其是一个功能完善、精准度较高、稳定性强的软件。

关键字　室内定位　可见光　LED　三边定位　Android

Research and Implementation of the Software for Visible Light Trilateration based on Android Platform

ABSTRACT

Nowadays, thanks to the rapid development of technology, intelligent life is no longer only a remote concept. Various smart devices and services emerge one after another, providing conviniance and helping people out. The service based on location is one of the most important part of intelligent services, and smart phone is nearly most popular among intelligent devices, thus making getting users' location information through smart phones an extremely important research direction.

GPS positioning technology is mature enough for solving problems under outdoor circumstances. However, when it comes to indoor environment where GPS signals are weak, precise localization still need better alternative technology. Several existing indoor positioning technologies have been proposed, yet they all have their weaknesses. For example, localization based on Wi-Fi has a low accuracy, and positioning based on infrared has a high cost. However, the accuracy of localization based on visible light can reach 0.1m, and the system is easy to deploy resulting in low cost.

This paper will research on visible light trilateration, choosing LED as light source and Android platform to implement software. This paper first compares advantages and disadvantages of exsiting indoor localization technologies and introduce concepts and features of relevant hardwares, models and technologies.. And then, according to standard process of software engineering, describes requirement, design, implementation, and finally test and evaluation of this software in turn.

The software for visible light trilateration in this paper has two localization modes-trilateration mode and pedometer caliberated with single light mode. To be specific, the first mode use visible light channel model and trilateration algorithem to localize. Yet the second mode calculates users' coordinates with step number taken by the pedometer, orientation recorded by azimuth angle sonsor, and set step length and then caliberates coordinates with visible light information.

All functions of this software have been implemented and test and evaluation of system have been completed. The result shows that it is a good software with comprehensive functions, high accuracy and high stablility.

KEY WORDS indoor positioning visible light LED trilateration Android

目 录

第一章 绪论

1.1 背景介绍

如今，人类已经进入智能生活时代。许许多多的智能服务如雨后春笋般涌现，而基于位置的服务是智能服务的重中之重，吸引了许多研究者的目光。

1.1.1 室内定位技术介绍

在室外环境下，Global Positioning System（GPS）定位技术已经十分成熟，并且应用范围广泛。全球覆盖率达到了 98%，综合定位精度可达厘米级甚至毫米级，民用领域开放的精度也可达到 10m 之内[1]。只要用户的手机可以与 4 颗以上的卫星相连并进行通讯，就可以精确地测量出用户的位置。但是在室内环境下，GPS 信号会被墙壁遮挡，导致定位精度下降，无法满足室内精准定位的要求[2]。

如今可以应用于室内场景的定位技术有很多，例如基于可见光[3]、Wi-Fi[4]、红外线（IR：infrared ray）[5]、超声波[6]。其中基于 IR 的室内定位技术是最早出现的，原理也相对简单，但其弱点是需要部署很多的传感器，因此成本较高[5]。基于 Wi-Fi 的室内定位技术是目前应用比较广泛的技术，它有着较高的覆盖率，并且不需要视距传播（LoS：line-of-sight）[错误!未找到引用源。]。但是由于多路径效应和信标的重叠，位置误差会达到 2m-5m[4]，达不到室内定位中一些应用（如商场店铺地图）对精度的要求。同时，基于 Wi-Fi、红外线、超声波的室内定位技术都对环境要求非常高，一旦增设或移除信标，就会导致定位结果不准确。

然而基于可见光的室内定位技术可以克服以上技术的缺点，定位精度可以达到分米级甚至厘米级，原理简单、易于实现，且易于部署、成本低廉。这些优点使得基于可见光的室内定位技术吸引了很多研究者的目光和兴趣，本文也将对这一技术进行研究和实现。

1.1.2 LED 介绍

发光二极管简称 LED（Light Emitting Diode），是半导体二极管的一种，可以把电能转化为光能[错误!未找到引用源。]。与白炽灯泡相比，发光二极管有以下优点：

- 节能，工作电压很低（大约是紧凑型荧光灯的二分之一）；

- 发光效率稳定性很强（70000 小时工作后下降约 10%[9]）；可靠性高，寿命长；
- 通过调制通过的电流强弱可以调制发光的强弱；
- 不含汞，环保[错误!未找到引用源。]。

以上特点使得 LED 灯很有可能在不远的将来大范围替代白炽灯泡，进入千家万户，引领照明界的革命。

另外，LED 的另一大特点就是瞬时开关，波形为方波，如图 1-1 所示，其中 $\frac{\tau}{T}$ 为占空比。

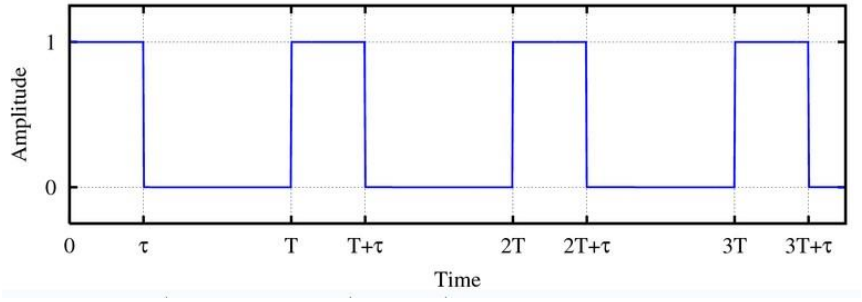


图 1-1 LED 灯波形图[10]

1.1.3 光信道模型介绍

为实现高精度三边定位，需要精确估计光源与接收设备之间的距离，为此需要构建光照强度与距离之间的模型，如图 1-2 所示。

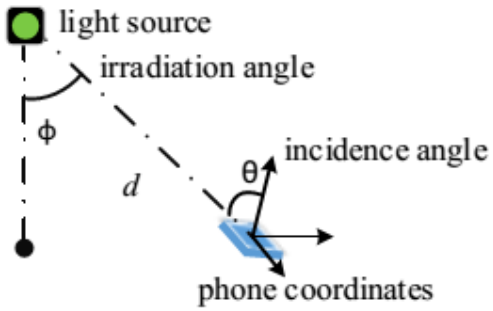


图 1-2 可见光发射接收示意图[3]

光信道模型满足朗伯模型，如公式（1-1）[3]：

$$\mathrm{Pr} = \mathrm{C} \cdot \sin\left(\frac{\tau}{T}\pi\right) \cdot \frac{\cos\theta \cos\varphi}{\mathrm{d}^2} \qquad (1\text{-}1)$$

其中 Pr 是光照强度，φ 为出射角，θ 为入射角（即光线与手机坐标系 z 轴之间的夹角），当 φ、θ 大于±60°时，误差将会变大，d 为光源与传感器之间的距

离，C 和占空比τ/T对于每一个灯来说都为常数，因此对于每个灯来说也满足公式（1-2）：

$$\mathrm{Pr} = \mathrm{C} \cdot \frac{\cos\theta \cos\varphi}{\mathrm{d}^2} \qquad (1\text{-}2)$$

本文对于该模型进行了验证，均符合公式（1-2），光照强度与入射角的关系如图 1-3，光照强度与出射角的关系如图 1-4，光照强度与距离之间的关系如图 1-5。

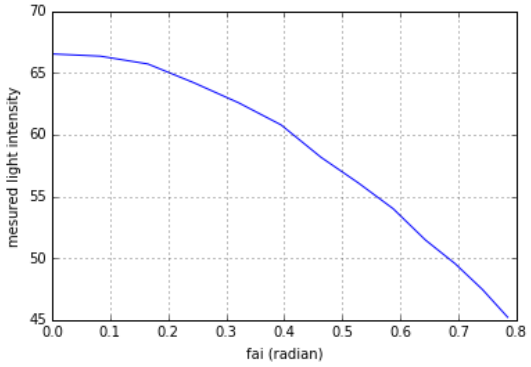


图 1-3 光照强度与入射角关系

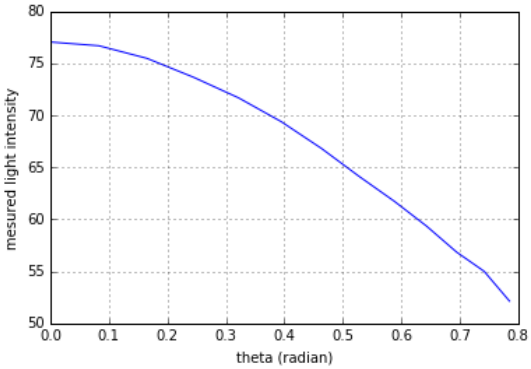


图 1-4 光照强度与出射角关系

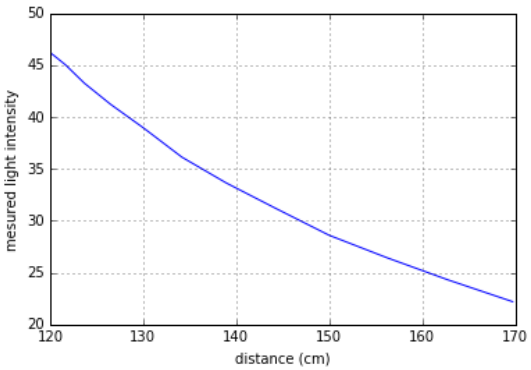


图 1-5 光照强度与距离关系

1.2 研究内容

随着智能服务对于精准位置计算的需求越来越高，室内精准定位吸引了很多研究的兴趣。

针对这种趋势，本文对其中一种定位技术——基于可见光的室内定位技术进行研究并实现相关软件。

为此首先需要对 LED、光传感器等硬件进行一定的了解，同时需要阅读大量可见光定位相关论文，比较不同定位算法的优缺点，并将基于测距的定位算法确定为本软件的实现算法。对于多灯定位来说，需要对朗伯模型、三边定位算法以及手机姿态角的计算方法进行研究。为了将不同光源区分开来，还需要学习快速傅里叶变换相关知识。另外本文还提出了一种使用计步器与航向角传感器来推算运动轨迹及坐标，并利用单灯来进行校准的定位模式。这其中需要了解计步器的实现原理。

在研究透彻理论实现方法之后，还要对具体实现方法进行研究，包括对 Android、Java、Python 开发的复习。具体来说，主要内容如下：

- 在采集可见光数据的部分要对 Android 的录音 API 以及数据存储方式进行研究；
- 在处理可见光数据的部分要对最小二乘法进行相关学习，以得到优化结果；
- 在手机姿态计算的部分主要需要对获取方位角传感器数据、对其的平滑操作以及旋转矩阵的乘法进行研究；
- 在多灯定位部分要探究多元高次方程组的求解方法以及定位结果展示的实现方法，另外 Android 端和 PC 上的 Python 端的通信的实现方法也需要复习；
- 计步器推算位置与单灯校准融合的定位模式主要需要对计步器的实现方法进行研究。

1.3 论文结构

本论文是本人毕业设计基于 Android 平台的可见光测距定位软件的研究和实现工作成果的总结，共由七章组成，论文的组织结构如下：

第一章，绪论，概述了本研究的背景，介绍了现有的几种定位技术并比较了它们的优缺点，阐述了 LED 和光信道模型的相关概念以及本论文的主要研究内容。

第二章，相关技术介绍，概述了项目所需要用到的具体技术，包括信号处理技术、数据处理技术、软件系统开发和硬件系统搭建的相关技术。

第三章，系统需求分析，主要描述了本软件的功能性需求和非功能性需求。

第四章，系统设计，主要描述了系统的体系结构，各个功能模块的设计。整个系统主要包括 6 个功能模块：可见光数据采集模块、可见光数据处理模块、手机姿态角计算模块、多灯定位模块、计步器定位模块、单灯校准模块。

第五章，系统功能实现和结果演示，主要描述了开发工具的选择、数据的存储模式、功能模块的具体实现以及程序运行效果演示。

第六章，系统测试和验证，主要描述了测试环境、测试用例以及定位结果的准确性。

第七章，总结与展望，对本文的主要工作进行总结，并对未来的工作和研究方向进行展望。

第二章　相关技术介绍

本章主要概述了项目所需要用到的具体技术，包括信号处理技术、数据处理技术、软件系统开发和硬件系统的相关技术。

2.1 信号处理技术

在多灯定位模式下，需要将三个灯区分开来，分别得到它们的光照强度。单灯校准模式下，需要区分检测到的光源，并以对应光源的坐标进行校准。要做到这些，就需要使用快速傅里叶变换（FFT：Fast Fourier Transformation）[错误!未找到引用源.]来将时域数据转换为频域数据。

本节主要介绍 FFT 的物理意义。一个模拟信号，经过 ADC（Analog-to-Digital Converter）[错误!未找到引用源.]采样变为数字信号，采样频率需要尽可能比信号频率大，这样一个信号周期的采样点才更多，可以更完整、更好地反映信号的信息。这些时域数据就可以用来做 FFT。选取其中 N 个采样点进行变换，N 称为窗口大小，变换之后得到 N 个 FFT 结果，通常 N 取 2 的整数次方。

假设采样频率为 Fs，信号频率为 F，采样点数为 N。那么 FFT 之后得到的结果就是 N 个复数结果，每一个结果对应一个频率点，如图 2-1。每个点的模值代表着该频率值下的幅度特性。假设原始信号的峰值为 A，那么 FFT 结果的模值就是 A*N/2。除去第一个点是直流分量，它的模值就是直流分量的 N 倍。每个点的相位也对应该频率信号的相位。第一个点表示直流分量，即 0Hz，第 N+1 个点（实际上不存在）则代表采样频率 Fs，这中间有 N-1 个点，将频域平均分成 N 等份，每个点的频率依次增加。因此频域中某点 n 所表示的频率为：$\frac{(n-1)*Fs}{N}$。频率分辨率为 Fs/N [错误!未找到引用源.]。

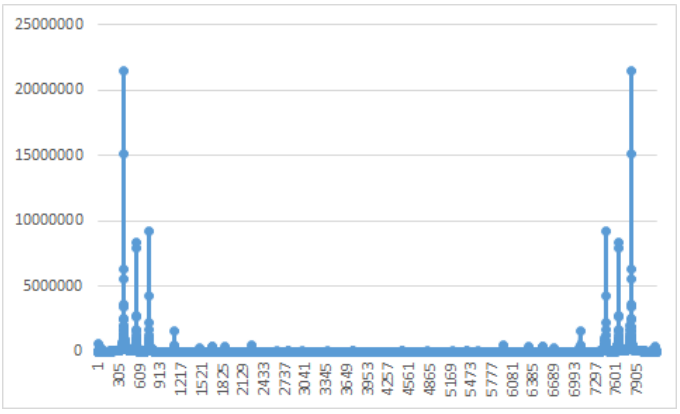


图 2-1 FFT 结果图

本文选用闪烁频率为 2K、3K、4K 的 LED 灯，采样频率设置为 44.1K，窗口大小设置为 8K，则频率分辨率为$\frac{Fs}{N}$，约为 8.82Hz，2K 灯对应频点n = $\frac{Fn(2K)*N(8K)}{Fs(44.1K)}+1$，约为 373。实际数据与理论结果会有一定偏差，实际测得结果如图 2-2。

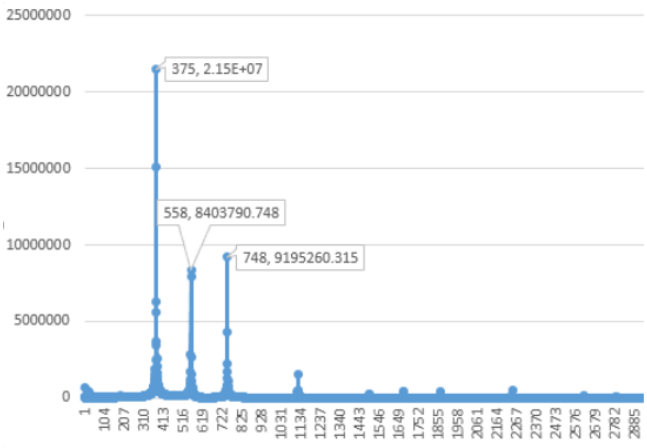


图 2-2 三灯同时开启时测得 FFT 数据

2.2 数据处理技术

2.2.1 平滑处理

对于实际数据接近于平稳不变的情况，可以应用平滑处理，以消除偶然因素的影响[错误!未找到引用源。]。在本软件中，由于需要调用安卓传感器，其数值是处于不断变化过程中的，如果仅使用一次的数值，则无法消除偶然因素的影响，因此要使用平滑处理来规避这种影响。

本文中使用的平滑算法是简单平均法，将一段较短时间内的各数据求和并除以数据数量，求得算术平均数既为预测值。由于本文中需要采集的数据变化较小且无明显趋势，可采用此法进行短期预测[错误!未找到引用源。]。

2.2.2 最小二乘优化

最小二乘法是一种数学优化技术，它通过最小化误差的平方和寻找数据的最佳函数匹配。利用最小二乘法可以简便地求得未知的数据，并使得这些求得的数据与实际数据之间误差的平方和为最小[错误!未找到引用源。]。

本文在对各个灯的常数参数进行拟合以及三边定位求解多元高次方程组的部分均用到了最小二乘优化技术进行优化。

2.3 软件系统开发

2.3.1 Android 操作系统

Android 是一种基于 Linux 的自由及开放源代码的操作系统，主要使用于移动设备，如智能手机和平板电脑，由 Google 公司和开放手机联盟领导及开发[错误! 未找到引用源。]。

Android 平台的主要优势有：

1. 开放性：

Android 开发平台是开源的，越来越多的开发者因为其开放性而加入 Android 开发当中，因此有大量开源的代码库、免费的开源软件，帮助 Android 平台快速地走向成熟。

2. 运营商对网络的束缚减少：

以前的手机应用在功能和网络方面会很大程度地受到运营商的限制，而 Android 在终端天生就有网络特色，同时随着网络的不断发展，会带来更多更好的用户体验。

3. 丰富的硬件选择：

Android 平台允许任何移动终端厂商加入到阵营当中，因此很多厂商为了达到更加吸引用户的目的，会对 Android 系统基础上的硬件加以改造，推出具有不同功能特色的产品，针对不同用户群体，丰富用户体验。

4. 软件开发中的不受限制：

第三方开发商可以十分自由的开发软件，大量的开源代码库也使得开发变得简单快速，软件的功能和设计也是不断地更新和增强。

5. 无缝结合的 Google 应用：

Android 平台手机可以无缝结合 Google 公司推出的服务，例如地图、邮件、搜索等[17]。

2.3.2 Python

Python 是一种面向对象的解释型计算机程序设计语言，它常被昵称为胶水语言，能够把用其他语言制作的各种模块（尤其是 C/C++）很轻松地联结在一起。

Python 语言的主要优势有：

1. 简单易学：

Python 语言十分简单。因此开发者能够专注于需要解决问题本身而不是去理解一门复杂的编程语言。它还极其容易上手，因为有简单且详细的说明文档。

2. 速度快：

Python 的底层以及很多标准库和第三方库都是由 C 语言实现的，因此运行速度很快。

3. 免费、开源：

Python 是开源软件之一。使用者可以在社区中自由地发布软件，阅读其他软件的源码，对其做改进，或者将一部分代码用于自己开发的软件当中。

4. 高层语言：

在使用 Python 语言进行编程时，开发者无需考虑底层细节，例如内存管理等问题。

5. 可移植性：

由于 Python 的开源，Python 也已经拥有了强度大的可移植性。可移植的平台包括 Linux、Windows、FreeBSD、Macintosh、Solaris、OS/2、Amiga、AROS、AS/400、BeOS、OS/390、z/OS、Palm OS、QNX、VMS、Psion、Acom RISC OS、VxWorks、PlayStation、Sharp Zaurus、Windows CE、PocketPC、Symbian 以及 Google 基于 linux 开发的 android 平台[错误!未找到引用源。]。

6. 可扩展性：

在 Python 程序中可以使用其他语言（如 C 或 C++）的代码，这样可以使得一段关键代码运行速度提高或者算法不公开。

7. 可嵌入性：

也可以把 Python 嵌入其他语言的程序，为程序提供脚本功能。

8. 丰富的库：

Python 标准库很庞大，它可以帮助处理各种工作，包括正则表达式、文档生成、单元测试、线程、数据库、网页浏览器、CGI、FTP、电子邮件、XML、XML-RPC、HTML、WAV 文件、密码系统、GUI（图形用户界面）、Tk 和其他与系统有关的操作[错误!未找到引用源。]。除此之外，还有许多优秀的第三方库，如 Pyplot、SciPy 和 Pandas 等等。

9. 规范的代码：

Python 的强制缩进使得代码具有较好可读性，并且 Python 语言程序无需编译成二进制代码。

2.4 硬件系统

2.4.1 光敏电阻模块

光敏电阻是用硫化隔或硒化隔等半导体材料制成的特殊电阻器，如图 2-3，其工作原理是内光电效应。光照愈强，阻值就愈低，随着光照强度的升高，电阻值迅速降低，亮电阻值可小至 1KΩ 以下。光敏电阻对光线十分敏感，其在无光照时，呈高阻状态，暗电阻一般可　达 1.5MΩ [错误!未找到引用源。]。

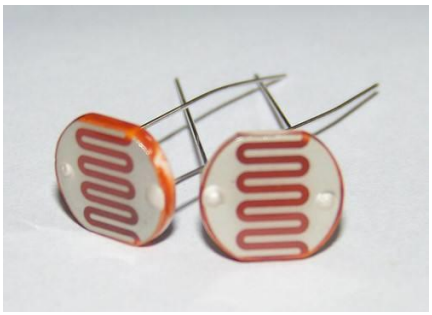


图 2-3 光敏电阻

光敏电阻的光电特性曲线，如图 2-4，以及对于多光源的响应都是非线性的，因此实际上不宜作为精度要求较高的线性光电转换器件[20]，但本文的主要研究内容不是硬件，因此选用了光敏电阻模块。

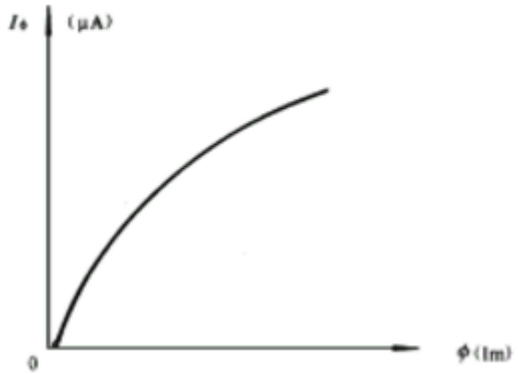


图 2-4 光敏电阻的光电特性曲线（横轴为光通量，纵轴为光电流）

2.4.2 安卓传感器

本文主要使用了安卓的加速度传感器、方向传感器以及光线传感器，本节对这些传感器进行简要介绍。

1. 加速度传感器：

加速度传感器（G-sensor），调用时返回 x、y、z 三轴的加速度数值。该数值包含地心引力的影响，单位是 m/s^2。当手机面朝上平置在水平面上时，x 轴加速度近似为 0，y 轴加速度近似为 0，z 轴加速度近似为 9.81。当手机面朝下平置在水平面上时，z 轴加速度近似为-9.81[21]。手机向左倾斜，x 轴为正值。手机向右倾斜，x 轴为负值。手机向上倾斜，y 轴为负值。手机向下倾斜，y 轴为正值。

目前市面上的加速度传感器种类有很多，手机中常用的加速度传感器有 BOSCH（博世）的 BMA 系列，AMK 的 897X 系列，ST 的 LIS3X 系列等[21]。这些传感器一般提供±2G 至±16G 的加速度测量范围，采用 I2C 或 SPI 接口和 MCU 相连，数据精度小于 16bit[21]。

2. 方向传感器：

方向传感器（O-sensor），调用时返回三轴的角度数据，单位是角度。为了得到精确的角度数据，电子罗盘需要获取 G-sensor 的数据，经过计算生产 O-sensor 数据，否则只能获取水平方向的角度。电子罗盘会有一个后台进程来完成这部分工作，其算法一般是电子罗盘公司的私有产权。

方向传感器返回的三个数据分别为航向角（azimuth）、俯仰角（pitch）和横滚角（roll）。航向角返回水平时磁北极和 Y 轴的夹角，范围为 0°至 360°。0°为北，90°为东，180°为南，270°为西。俯仰角返回 x 轴和水平面的夹角，范围为-180°至 180°。当 z 轴向 y 轴转动时，角度为正值。横滚角返回 y 轴和水平面的夹角，范围为-90°至 90°。当 x 轴向 z 轴移动时，角度为正值。

在获取正确的数据前，电子罗盘通常需要用 8 字校准法进行校准。8 字校准法要求用户使用需要校准的设备在空中做 8 字晃动，原则上尽量多的让设备法线方向指向空间的所有 8 个象限[21]。

手机中使用的电子罗盘芯片有 AKM 公司的 897X 系列，ST 公司的 LSM 系列以及雅马哈公司等等[21]。

3. 光线传感器：

光线感应传感器检测实时的光照强度，单位是 lux，其物理意义是单位面积上的光通量。

2.5 本章小结

本章概述了在本文中主要用到的相关技术的知识。包括信号处理技术、数据处理技术、软件系统开发和硬件系统的相关技术。为后文进行理论知识的铺垫。

第三章　可见光测距定位软件的需求分析

本章主要描述了基于 Android 平台的可见光测距定位软件的语境分析、功能性需求和非功能性需求。

3.1 功能性需求

3.1.1 用例图

可见光数据采集功能包括录音、读取数据、快速傅里叶变换、取出数据并平滑数据、传输数据、接收结果，其用例图如图 3-1。

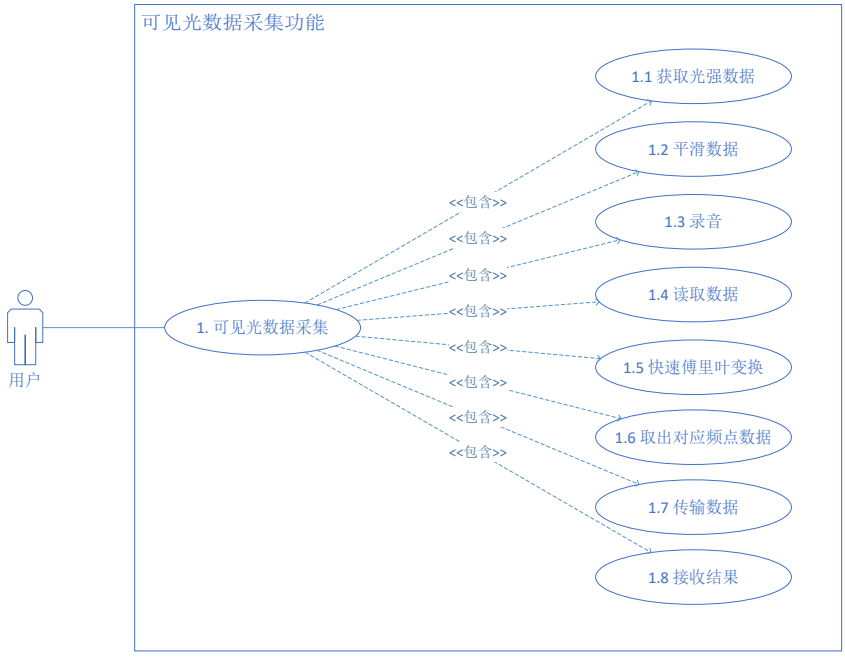


图 3-1 可见光数据采集功能用例图

可见光数据处理功能包括接收数据、整理数据、参数拟合。其用例图如图3-2。

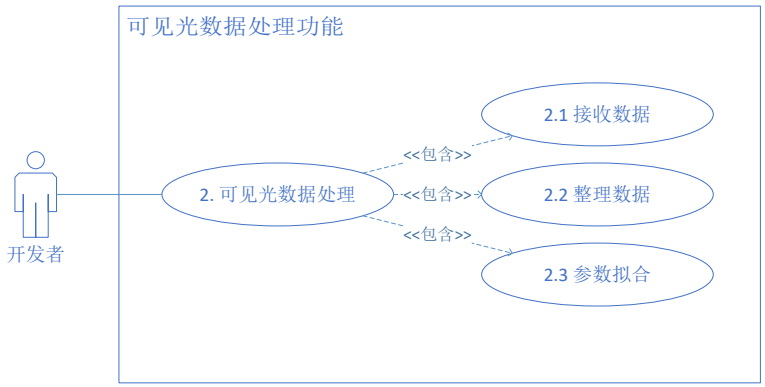


图 3-2 可见光数据处理功能用例图

手机姿态计算功能包括获取方向数据、平滑数据、计算手机法向量、传输数据。其用例图如图3-3。

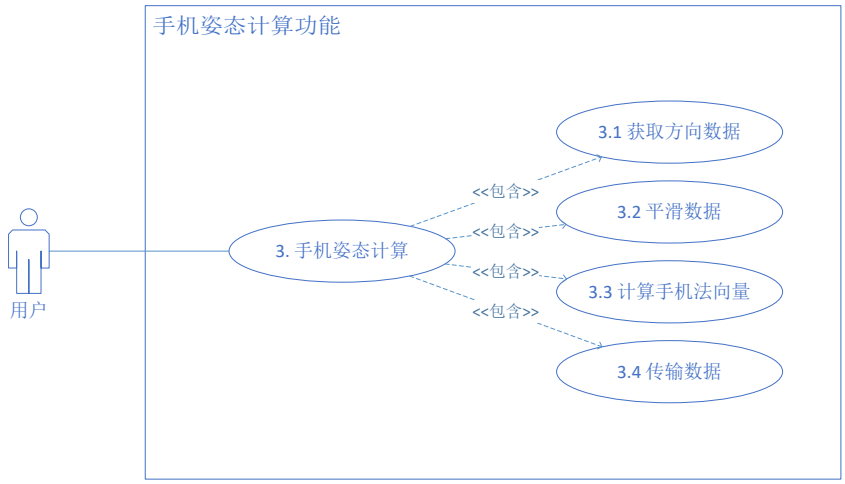


图 3-3 手机姿态计算功能用例图

多灯定位功能包括接收数据、三边定位、绘制结果图、返回结果。其用例图如图3-4。

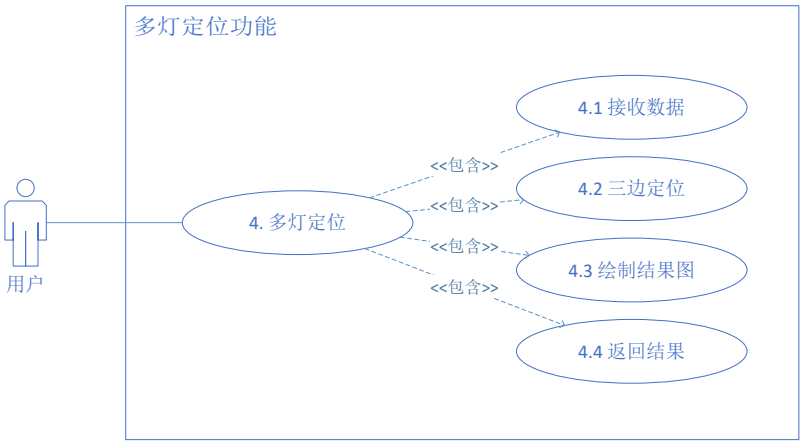


图 3-4 多灯定位功能用例图

计步器定位功能包括获取航向角数据、平滑数据、计步、设置步长、推算坐标。其用例图如图 3-5。

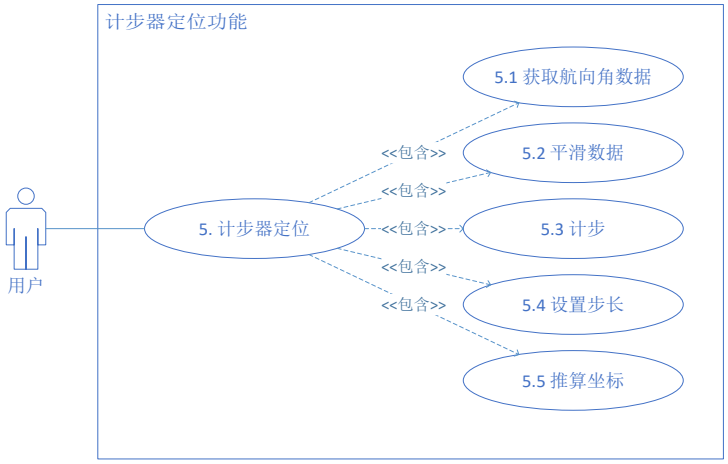


图 3-5 计步器定位功能用例图

单灯校准功能包括持续监测对应频率幅度值、位置校准。其用例图如图 3-6。

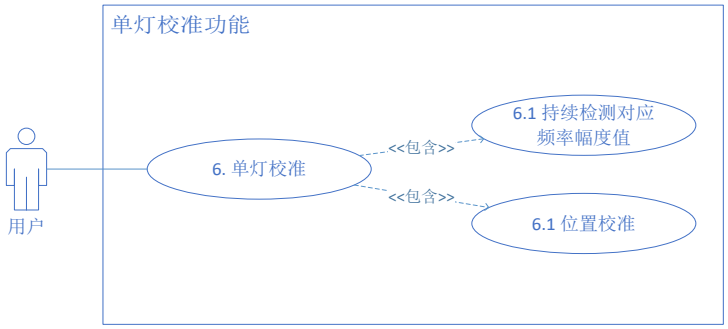


图 3-6 单灯校准功能用例图

系统的整体用例图如图 3-7。

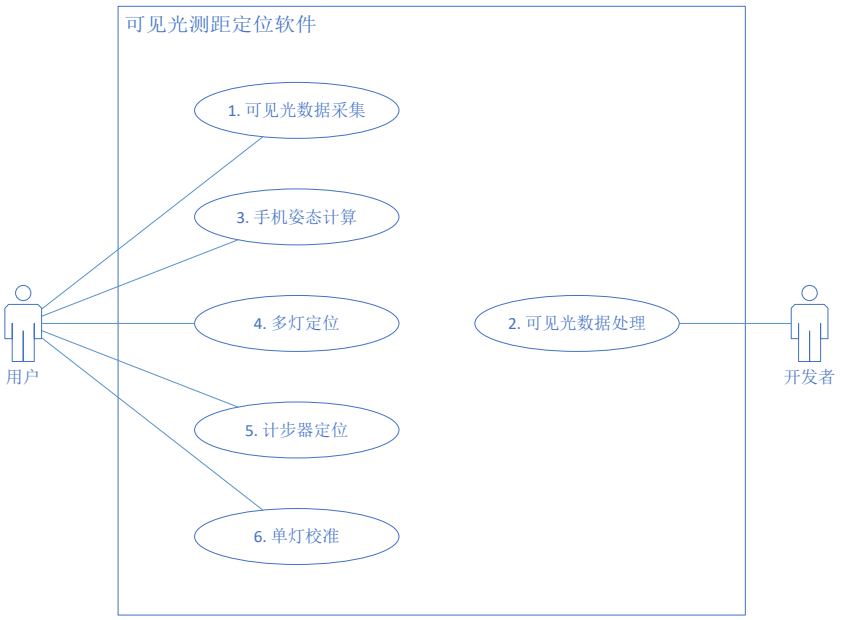


图 3-7 可见光测距定位软件的系统用例图

3.1.2 功能描述

本小节对软件所需功能进行描述，如表 3-6。

表 3-6 可见光测距定位软件的功能描述

功能	概述	前置条件	输入信息	输出信息	异常处理
1.1 获取光强数据	调用光线传感器 API 获取光强数据并记录	多灯测距定位模式下使用光强进行定位并点击“记录光强”按钮	无	光强数据	无
1.2 平滑数据	记录各次光强数据，并使用简单平均法进行平滑	获取光强数据结束	方向数据	平滑后的方向数据	无
1.3 录音	调用录音 API 将光信号转化为声音信号并录制在 pcm 文件中	多灯测距定位模式下使用幅度进行定位并点击“记录幅度”按钮或选择计步器与单灯校准融合定位模式之后	光信号	pcm 文件	App 录音及文件存储权限未打开
1.4 读取数据	从录制好的 pcm 文件中将数据读取到内存中，存为 double[]形式	录音结束	pcm 文件	时域数据	pcm 文件损坏
1.5 快速傅里叶变换	取出窗口大小为 8K 的时域数据，交给 FFT 包做快速傅里叶变换，并对其结果求模值	读取数据结束	时域数据	频域数据	无
1.6 取出对应频点数据	选取 LED 灯频率对应的频点的数据，并做平滑操作	快速傅里叶变换结束	频域数据	幅度数据	无

续上表

1.7 传输数据	将光强数据或FFT 之后的对应结果传输至可见光处理功能或多灯定位功能	点击“传输数据”按钮	光强数据或幅度数据	无	socket 连接失败
1.8 接收结果	等待并接收服务器传来的最终定位结果	多灯测距定位模式下传输数据结束	无	定位结果	socket 连接失败
2.1 接收数据	接收从可见光数据采集功能传来的数据	打开服务器并与客户端连接结束	无	光强数据或幅度数据	socket 连接失败
2.2 整理数据	将接收到的数据加入到对应数组当中	接收数据结束	光强数据或幅度数据	无	无
2.3 参数拟合	使用最小二乘法对相应参数进行拟合	整理数据结束	数组	参数	无
3.1 获取方向数据	调用方向传感器API 获取航向角、俯仰角、横滚角三个方向数据	选择多灯测距定位模式之后	无	方向数据	无
3.2 平滑数据	记录各次方向数据，并使用简单平均法进行平滑	获取方向数据结束	方向数据	平滑后的方向数据	无
3.3 计算手机法向量	将初始法向量(0, 0, 1)乘以旋转矩阵得到手机的法向量	平滑数据结束	平滑后的方向数据	手机法向量	无
3.4 传输数据	将当前屏幕法向量传输至多边定位功能	计算手机法向量结束	手机法向量		无

续上表

4.1 接收数据	接收从可见光数据采集功能以及手机姿态计算功能传来的数据	打开服务器并与客户端连接结束	无	光强数据或幅度数据、手机法向量	socket 连接失败
4.2 三边定位	求解三边定位方程组	接收数据结束	光强数据或幅度数据	定位结果	无
4.3 绘制结果图	将定位结果和灯的位置绘制在一张图当中进行展示	三边定位结束	定位结果	结果展示图	无
4.4 返回结果	将定位结果发送至 app	三边定位结束	定位结果	无	socket 连接失败
5.1 获取航向角数据	调用方向传感器 API，获取航向角数据	进入计步器与单灯校准融合定位模式	无	航向角数据	无
5.2 平滑数据	记录各次方向数据，并使用简单平均法进行平滑	进入计步器与单灯校准融合定位模式	方向数据	平滑后的航向角数据	无
5.3 计步	使用计步器模块来计步	进入计步器与单灯校准融合定位模式	无	步数	无
5.4 设置步长	用户设置自己的大概步长	选择计步器与单灯校准融合定位模式	步长	无	偏离一般值过多
5.5 推算坐标	根据航向角、步数以及步长来推算坐标，初始值设为 (0, 0)	航向角、步数、步长数据采集完毕	航向角、步数、步长	坐标	无

续上表

6.1 持续检测对应频率幅度值	持续调用功能1.1-1.4，获取对应频率的幅度值	进入计步器与单灯校准融合定位模式	无	对应频率幅度值	App 录音及文件存储权限未打开
6.2 位置校准	当发现对应频率的幅度值大于设定阈值时，将手机的坐标设置为灯的坐标	对应频率的幅度值大于设定阈值	对应频率幅度值	坐标	无

3.2 非功能性需求

3.2.1 可靠性

应用在没有发生非人为（自然灾害，断电等不可抗拒的事件）操作失误时，不得出现系统崩溃不可用的情况。也就是，不要出现手机软件常有的闪屏、闪退问题。

3.2.2 可移植性

本应用的移动终端版本能够在安卓 4.4 及以上版本系统上正常运行。

3.3 本章小结

本章介绍了基于 Android 平台的可见光测距定位软件的需求，包括功能性需求和非功能性需求。功能性需求使用功能结构图、用例图以及功能描述的方式进行介绍。非功能性需求主要介绍了对于可靠性和可移植性的需求。

第四章　可见光测距定位软件的系统设计

本章主要描述了基于 Android 平台的可见光测距定位软件的系统体系结构、系统组织设计、系统功能设计以及系统性能设计。

4.1 系统体系结构

基于 Android 平台的可见光测距定位软件主要分为两个定位模式，一个是多灯测距定位模式，该模式主要用到的模块有可见光数据采集模块、可见光数据处理模块、手机姿态计算模块、多灯定位模块；另一个是计步器与单灯校准融合定位模式，该模式主要用到的模块有可见光数据采集模块、计步器定位模块、单灯校准模块。其体系结构如图 4-1 所示。

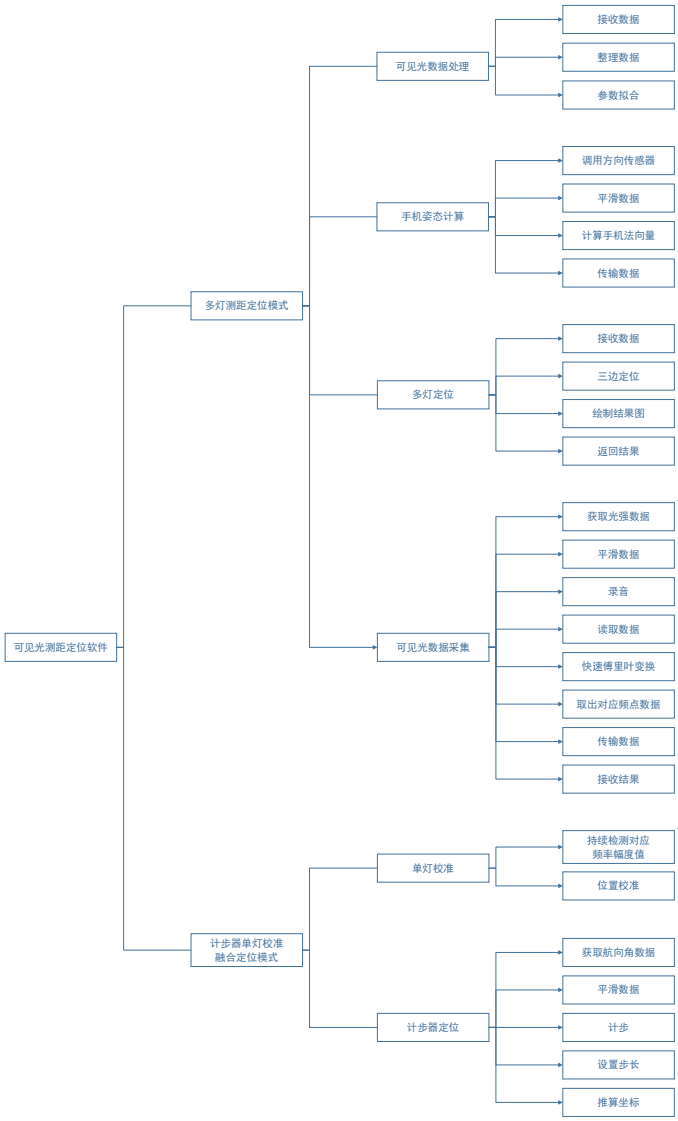


图 4-1 可见光测距定位软件的系统体系结构

可见光数据采集主要负责采集可见光数据。首先将光信号转化为声音信号并存储为 pcm 文件，然后从 pcm 文件中将数据读取到内存中，之后再对这些数据做快速傅里叶变换并取出数据后做平滑操作。在多灯测距定位模式中，如果使用光强数据进行定位，则取出对应频点的最大值，判断出是哪个光源，然后调用光线传感器获取光强数据并对其进行平滑操作之后记录；如果使用幅度数据进行定位则将最大值对应数据记录下来。之后将光强数据或幅度数据传输至多灯定位模块，然后等待定位结果。

可见光数据处理主要负责对采集到的可见光进行处理，首先接收从可见光数据采集模块传来的光强数据或幅度数据，将其装进对应数组中，再使用最小二乘法对数据进行拟合。

手机姿态计算主要负责计算当前手机屏幕的法向量以便三边定位，首先调用方向传感器获取航向角、俯仰角以及横滚角的数据，并对其做平滑操作，再乘以旋转矩阵以得到手机屏幕的法向量，最后将法向量传输至多灯定位模块。

多灯定位主要负责计算位置坐标，在接收到从可见光数据采集模块传来的光强数据或幅度数据以及从手机姿态计算模块传来的法向量后，将朗伯模型中关于角度以及距离的项全部用手机坐标（x,y,z）以及灯的坐标（x_i,y_i,z_i）表示，并对其解三边定位方程组，然后将定位结果以图的形式绘制在电脑屏幕上，再将结果返回至可见光数据采集模块。

计步器定位主要是根据航向角、步数以及步长三个数据来推算用户的运动轨迹以及位置信息。主要步骤有调用方向传感器以获取航向角数据，并对其进行平滑，之后使用计步器模块来进行计步，再设置一定的步长，就可以推算坐标了。

单灯校准主要负责对用户的位置进行校准，因为计步器定位误差会偏大，因此利用 LED 灯来校准。持续对对应频率的幅度值进行检测，一旦其值大于设定好的阈值，则将用户的坐标校准为灯的坐标。

4.2 系统组织设计

基于 Android 平台的可见光测距定位软件主要分为 6 大模块：可见光数据采集模块、可见光数据处理模块、手机姿态计算模块、多灯定位模块、计步器定位模块以及单灯校准模块。其系统组织表如表 4-1，系统组织图如图 4-2。

表 4-1 可见光测距定位软件的系统组织表

模块编号	中文名称	业务职能	安装地点	开发语言	备注
1	可见光数据采集模块	采集可见光数据及数据传输	Android 测试机	Java	在开始之前确保插入传感器模块并打开录音和数据存储权限。
2	可见光数据处理模块	拟合相关参数	部署 PC 服务器上	Python	开发者使用该模块获取相关参数
3	手机姿态计算模块	计算当前手机屏幕的法向量	Android 测试机	Java	获取方向数据之前需进行 8 字校准法
4	多灯定位模块	计算用户位置坐标	部署 PC 服务器上	Python	确保服务器与客户端连接成功
5	计步器定位模块	根据航向角、步数以及步长三个数据来推算用户的运动轨迹以及位置信息	Android 测试机	Java	获取方向数据之前需进行 8 字校准法，且步长设置不可偏离一般值太大
6	单灯校准模块	利用 LED 灯的位置信息对用户的位置进行校准	Android 测试机	Java	阈值需计算

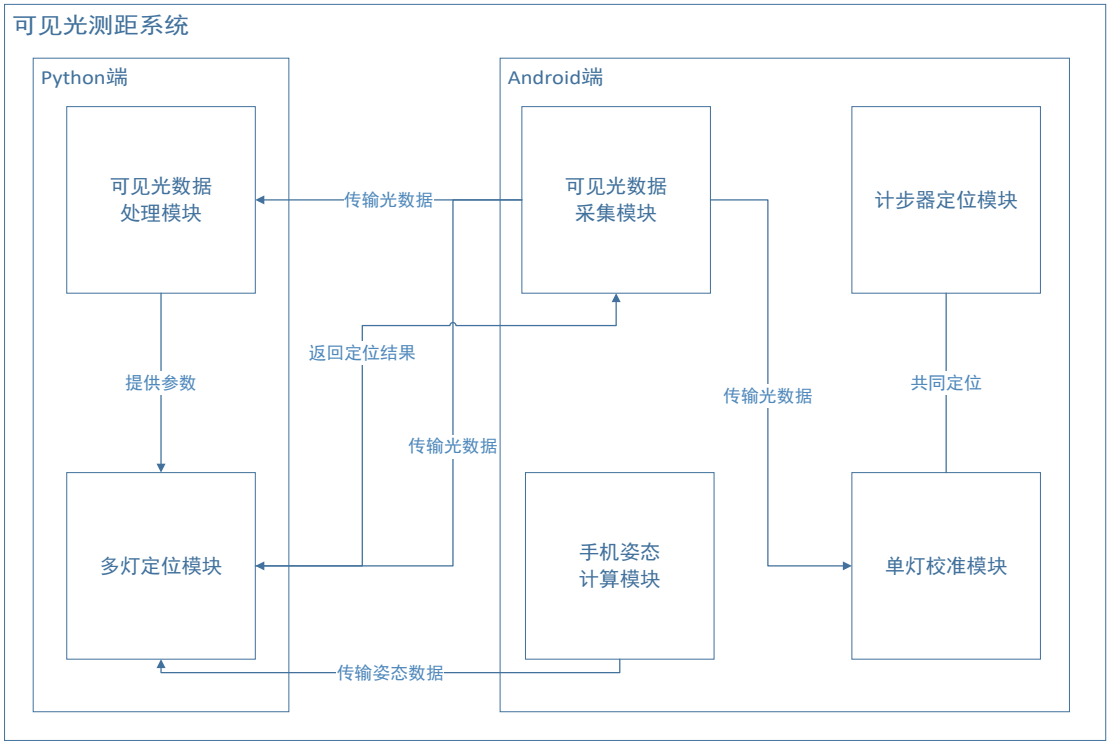


图 4-2 可见光测距定位软件的系统组织图

4.3 系统功能设计

4.3.1 功能模块划分

基于 Android 平台的可见光测距定位软件主要有 6 大功能模块，分别为：可见光数据采集功能模块、可见光数据处理功能模块、手机姿态计算功能模块、多灯定位功能模块、计步器定位功能以及单灯校准功能模块。本软件的功能结构图如图 4-3。

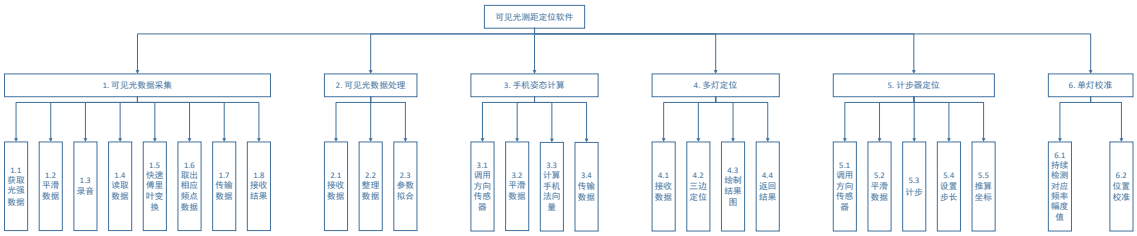


图 4-3 可见光测距定位软件的功能结构图

4.3.2 可见光数据采集模块

可见光数据采集模块主要负责采集可见光数据及数据传输。其工作流程如下：打开 App 后，首先选择定位模式，如果选择了多灯测距定位模式，则要继续选

择定位所用数据。如果选择了使用光强进行定位，则会进行 FFT 操作，主要包括录音、读取数据、快速傅里叶变换、取出对应频点数据、平滑数据，在之后需要对光源进行分辨，然后调用光线传感器 API 获取当前光强数据，在平滑操作后记录数据；如果选择了使用幅度进行定位，则同样重复 FFT 操作，之后分辨光源并记录幅度数据。当三个灯的数据全部采集完之后，就可以将光强或幅度数据发送给多灯定位模块或者可见光数据处理模块，并等待从多灯定位模块传回的定位结果；如果选择了计步器与单灯校准融合定位模式，则同样进行 FFT 操作，之后将幅度数据传至单灯校准模块。其流程图如图 4-4。

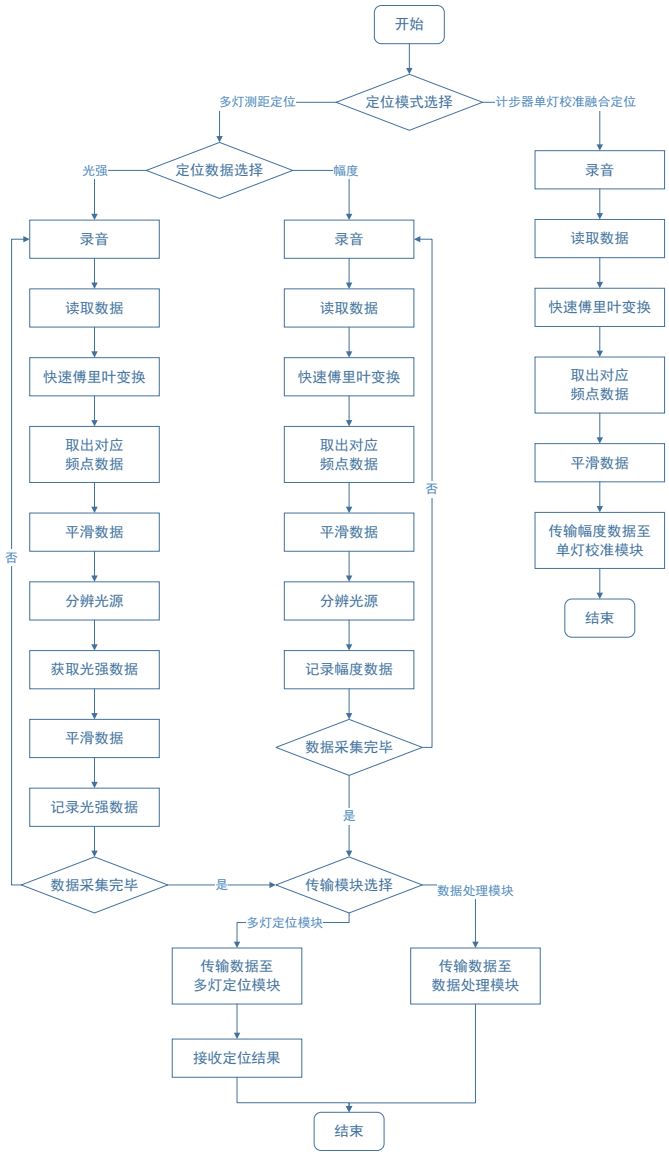


图 4-4 可见光数据采集模块流程图

本文选用三个 LED 灯作为光源，频率分别调制成 2KHz、3KHz、4KHz。

录音包括调用录音 API 进行录音以及将 byte 数据流写入.pcm 文件，录制音频的采样频率设置为 44.1K，采样设置为 1s，即共录制 44.1K 个采样点。这样每

一个波形可以采集到至少 10 个点，可以比较完整的反映波形。同时为了提升速度，使用双线程模式，在录音的同时写入文件。

在本文第二章中提到过，FFT 的窗口大小选定为 8K，因此将 44.1K 个采样点截取成 5 个 8K 的窗口，并分别进行 FFT。2K 灯的频点约为 375，3K 灯的频点约为 558，4K 灯的频点约为 748，因此取出幅度数据时，直接取数组中的第 375、558、748 个数即可。取出频点数据后，使用简单平均法进行平滑，即将 5 次 FFT 的结果相加再平均。

对于光强数据的平滑操作也选用简单平均法，设置一个数组，记录下 100 次光强数据，求其算数平均值作为这段时间的光强数据。

由于本文使用光敏电阻采集光数据，而光敏电阻对于不同光源的响应非线性导致不同光源之间会互相影响，使其幅度值减小。因此本文尽量避免同时开启光源，而采用分时复用的方法，仅仅使用快速傅里叶变换来辨别光源，具体来说就是在只开一个灯的情况下，如果该灯对应频点的数值在三个频点中最大，则辨别为此灯。

4.3.3 可见光数据处理模块

可见光数据处理模块主要负责拟合相关参数。其工作流程如下：打开 Python 端的服务器，接收来自可见光数据采集模块的数据，将数据添加入对应的数组中。如果是使用光强数据进行定位，则对朗伯模型中的常数 C 进行拟合；如果使用幅度数据进行定位，则不但要对朗伯模型中的常数 C 进行拟合，还要拟合幅度与光强之间的关系。其流程图如图 4-5。

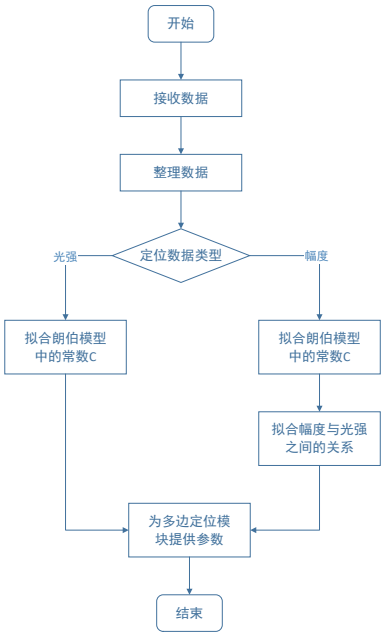


图 4-5 可见光数据处理模块流程图

其中，使用最小二乘拟合方法对参数进行拟合。

朗伯模型如公式 4-1。

$$\mathrm{Pr} = \mathrm{C} \cdot \frac{\cos\theta \cos\varphi}{\mathrm{d}^2} \tag{4-1}$$

为了简化实验，本课题组使手机水平放置在水平面上（手机屏幕法向量为（0, 0, 1））, 使得出射角 φ 与入射角 θ 相等，并记录下光强数据 Pr、传感器坐标（x, y, z）以及灯坐标（xi, yi, zi），因此$\cos\theta = \cos\varphi = \frac{|z-z_i|}{\sqrt{(x-x_i)^2+(y-y_i)^2+(z-z_i)^2}}$，$d^2 = (x-x_i)^2 + (y-y_i)^2 + (z-z_i)^2$，只剩下一个未知常数 C。构造代价函数如式 4-2，其中 i 为实验数据的编号：

$$\mathrm{Cost} = \sum_{\mathrm{j}=1}^{\mathrm{n}} (\mathrm{Pr_j} - \frac{\mathrm{C}}{\mathrm{d_j}^2} \cos\theta_\mathrm{j} \cos\varphi_\mathrm{j})^2 \tag{4-2}$$

光波的幅度与其强度的关系理论上是平方关系，但由于光敏电阻的光电转换是非线性的，且作者不清楚手机录音是否对信号做了增强或者是别的操作。因此本文假设幅度 A 与强度 Pr 满足关系（4-3），并构造代价函数如式（4-4）。

$$\mathrm{Pr} = \mathrm{a} \cdot A^2 + b \cdot A + c \tag{4-3}$$

$$\mathrm{Cost} = \sum_{\mathrm{j}=1}^{\mathrm{n}} [\mathrm{Pr_j} - (\mathrm{aA_j}^2 + \mathrm{bA_j} + \mathrm{c})]^2 \tag{4-4}$$

至于代价函数最小化的问题，本软件设计使用 Python 的 SciPy 包中的 leastsq[22]方法来解决代价函数最小化的问题，其算法是梯度下降[23]。该算法涉及到机器学习的知识，本文没有对该算法进行深入研究，在此不多赘述。

4.3.4 手机姿态计算模块

Android 手机的姿态如图 4-6 所示，其中将手机绕 z 轴转动，航向角会产生变化；绕 x 轴转动，俯仰角会产生变化；绕 y 轴转动，横滚角会产生变化。

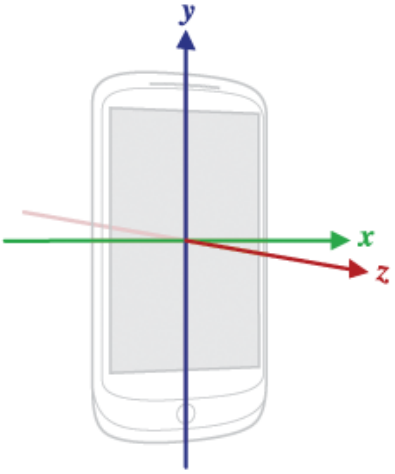


图 4-6 手机姿态[24]

在本软件中选择了多灯测距定位模式之后，调用方向传感器 API，获取航向角、俯仰角、横滚角的数值，并使用简单平均法进行平滑。之后将三个角与旋转矩阵相乘获取手机屏幕法向量，再使用向量夹角公式计算入射角 cos 值，最后将结果传至多边定位模块进行定位。手机姿态计算模块的流程图如图 4-7 所示。

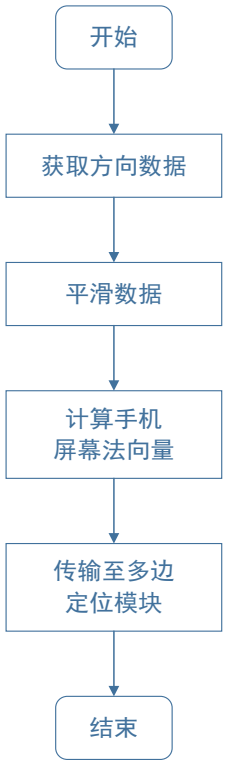


图 4-7 手机姿态计算模块流程图

其中计算手机屏幕法向量的方法是用初始向量$[0 \quad 0 \quad 1]^T$与旋转矩阵分别相乘得到。将绕绕 z 轴的旋转矩阵称为 A，绕 y 轴的旋转矩阵称为 B ，绕 x 轴的旋转矩阵称为 C，γ为航向角，β为横滚角，α为俯仰角。旋转矩阵如下：

A：

$$\begin{bmatrix} \cos\gamma & \sin\gamma & 0 \\ -\sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

B：

$$\begin{bmatrix} \cos\beta & 0 & -\sin\beta \\ 0 & 1 & 0 \\ \sin\beta & 0 & \cos\beta \end{bmatrix}$$

C：

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha \\ 0 & -\sin\alpha & \cos\alpha \end{bmatrix}$$

则手机屏幕法向量计算方法如式（4-5）：

$$\begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} \cos\gamma & \sin\gamma & 0 \\ -\sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\beta & 0 & -\sin\beta \\ 0 & 1 & 0 \\ \sin\beta & 0 & \cos\beta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha \\ 0 & -\sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{4-5}$$

4.3.5 多边定位模块

多边定位模块主要负责计算用户位置坐标，其工作流程如下：打开 Python 端的服务器，等待可见光数据采集模块以及手机姿态计算模块传来的数据。设传感器坐标为(x, y, z)，用 x、y、z 来表示朗伯模型中距离、角度的相关表达式。三个灯构造三个方程组，仅有 x、y、z 三个未知数，解非线性方程组得到定位结果。然后绘制定位结果图，并将结果传回至可见光数据采集模块。其流程图如图 4-8。

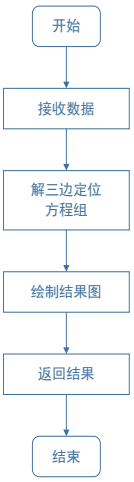


图 4-8 多边定位模块流程图

其中朗伯模型如公式（4-1）。

$$\mathrm{Pr} = \mathrm{C} \cdot \frac{\cos\theta \cos\varphi}{\mathrm{d}^2} \tag{4-1}$$

Pr 为光强，如果可见光数据采集模块传来的是光强数据则直接使用，如果可见光数据采集模块传来的是幅度数据，则使用可见光数据处理模块的拟合结果将幅度数据变为光强数据。C 是常数，由可见光数据处理模块提供。

设传感器坐标为(x, y, z)，三个灯的坐标(x_i,y_i,z_i)，i=1,2,3。

则出射角的 cos 值的表达式如式（4-6）：

$$\cos\varphi_i = \frac{|z - z_i|}{\sqrt{(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2}} \tag{4-6}$$

距离 d 的表达式如式（4-7）：

$$d_i = \sqrt{(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2} \tag{4-7}$$

入射角的 cos 值可由向量夹角公式计算得到，如公式（4-8）：

$$\cos\langle\vec{a},\vec{b}\rangle=\frac{\vec{a}\cdot\vec{b}}{|\vec{a}||\vec{b}|} \tag{4-8}$$

因此入射角的 cos 值得表达式如式（4-9），其中$\vec{n}$为手机平面法向量：

$$\cos\theta_i=\frac{(x_i-x,\ y_i-y,\ z_i-z)}{d_i}\cdot\vec{n} \tag{4-9}$$

即：

$$\cos\theta_i=\frac{(x_i-x,y_i-y,z_i-z)}{\sqrt{(x-x_i)^2+(y-y_i)^2+(z-z_i)^2}}\cdot\vec{n} \tag{4-10}$$

因此：

$$Pr_i=C_i\frac{\cos\theta_i\cos\varphi_i}{{d_i}^2}=C_i\frac{(x_i-x,y_i-y,z_i-z)\cdot\vec{n}\cdot|z-z_i|}{((x-x_i)^2+(y-y_i)^2+(z-z_i)^2)^2} \tag{4-11}$$

三个灯可构造三个关于(x, y, z)的非线性方程组，如式（4-12）：

$$\begin{cases}Pr_1=C_1\dfrac{(x_1-x,y_1-y,z_1-z)\cdot\vec{n}\cdot|z-z_1|}{((x-x_1)^2+(y-y_1)^2+(z-z_1)^2)^2}=F_1(x,y,z)\\ Pr_2=C_2\dfrac{(x_2-x,y_2-y,z_2-z)\cdot\vec{n}\cdot|z-z_2|}{((x-x_2)^2+(y-y_2)^2+(z-z_2)^2)^2}=F_2(x,y,z)\\ Pr_3=C_3\dfrac{(x_3-x,y_3-y,z_3-z)\cdot\vec{n}\cdot|z-z_3|}{((x-x_3)^2+(y-y_3)^2+(z-z_3)^2)^2}=F_3(x,y,z)\end{cases} \tag{4-12}$$

代价函数为：

$$Cost=[Pr_1-F_1(x,y,z)]^2+[Pr_2-F_2(x,y,z)]^2+[Pr_3-F_3(x,y,z)]^2 \tag{4-13}$$

本软件设计使用 Python 的 SciPy 包中的 fslove[25]方法来求解非线性方程组，其算法有 3 种，分别为 trust region dogleg[26]，trust region reflective[27]以及 Levenberg-Marquardt[28]。本文没有对这几种优化算法进行深入研究，在此不多赘述。

4.3.6 计步器定位模块

计步器模块的主要职责是根据航向角、步数以及步长三个数据来推算用户的运动轨迹以及位置信息，其工作流程如下：在 App 上选择计步器与单灯校准融合定位模式后，调用方向传感器 API 获取航向角数据，使用简单平均法进行平滑，以获得方向，再让用户设置步长，同时使用计步器模块来计步，步数每增加 1 步就推算并更新一次坐标。其流程图如图 4-9。

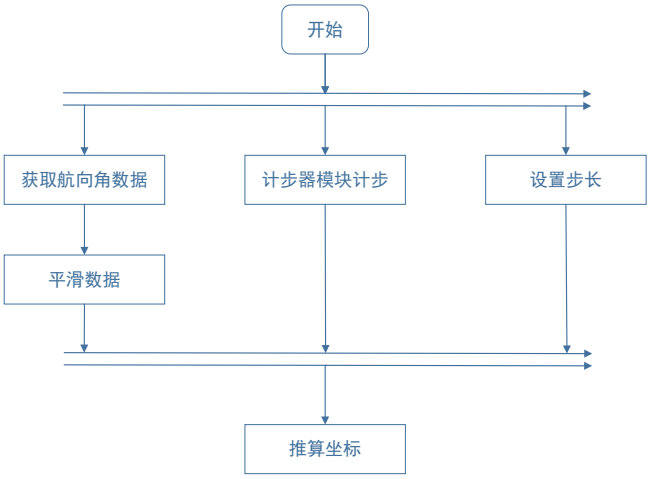


图 4-9 计步器定位模块流程图

航向角为γ，步数为 i，步长为 l。若设初始位置$(x_0, y_0) = (0,0)$，则每增加一步的坐标为：

$$(x_i, y_i) = (x_{i-1} + l * \sin\gamma, y_{i-1} + l * \cos\gamma) \quad (4\text{-}14)$$

4.3.7 单灯校准模块

单灯校准模块主要利用 LED 灯的位置信息对用户的位置进行校准，其流程如下：在 App 上选择计步器与单灯校准融合定位模式后，不断调用可见光数据采集模块以获取幅度数据，当幅度值大于实验测得阈值时，则将传感器的坐标位置校准至对应灯的坐标位置。其流程图如图 4-10。

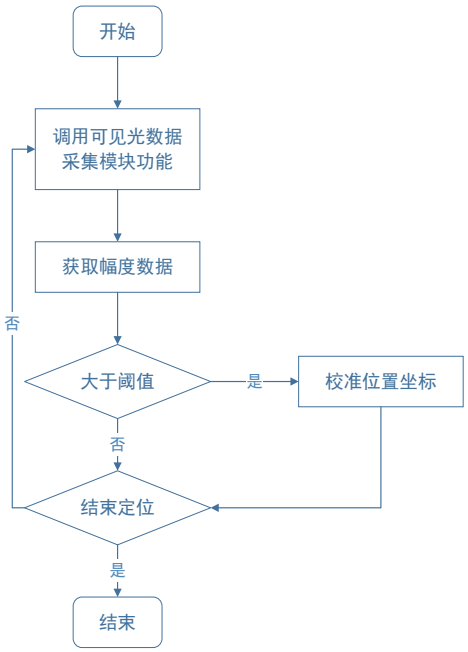


图 4-10 单灯校准模块流程图

4.4 系统性能设计

4.4.1 可靠性

为了保证应用在没有发生非人为（自然灾害，断电等不可抗拒的事件）操作失误时，不出现系统崩溃不可用的情况，从两方面进行设计，一是文件，二是线程。删除使用完毕的 pcm 文件，并在线程运行完毕后释放线程资源。

4.4.2 可移植性

为了使本应用的移动终端版本能够在安卓 4.4 及以上版本系统上正常运行，系统的常量需要根据系统设置，并需要对数值进行校准。

4.5 本章小结

本章节着重描述了可见光测距定位软件的整体体系结构，并详细介绍了各模块的设计。本软件包括 6 大模块：可见光数据采集模块、可见光数据处理模块、手机姿态计算模块、多变定位模块、计步器定位模块、单灯校准模块。本章还对系统的组织及性能的设计做了描述。

第五章　可见光测距定位软件的实现

本章节主要介绍基于 Android 平台的可见光测距定位软件关键部分的实现以及成果展示。

5.1 开发环境

5.1.1 客户端开发环境

1. JAVA 环境：JDK 1.8.0_111。

使用 cmd.exe 查看 java 版本，如图 5-1。

```
C:\Users\Administrator>java -version
java version "1.8.0_111"
Java(TM) SE Runtime Environment (build 1.8.0_111-b14)
Java HotSpot(TM) 64-Bit Server VM (build 25.111-b14, mixed mode)
```

图 5-1 Java 环境

2. Android 环境：Android sdk 5.1.1。

使用 Eclipse 查看安卓版本，如图 5-2。

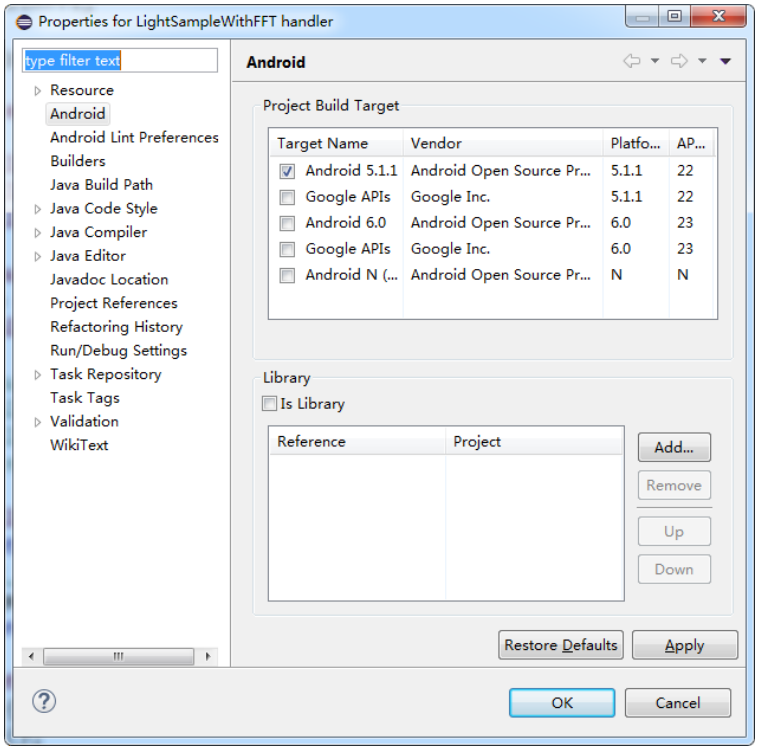


图 5-2 Andorid 环境

3. IDE：Eclipse Mars.1 Release (4.5.1)，开发界面如图 5-3。

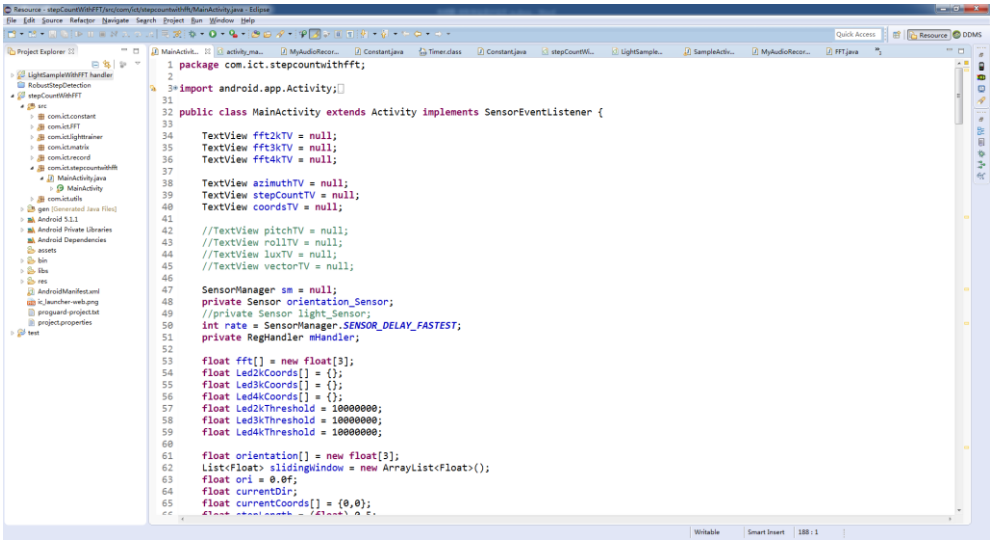


图 5-3 Andorid IDE

5.1.2 服务器开发环境

1. Python 环境：Python 3.5.2 version　Anaconda 4.1.1 for Windows。使用 python.exe 查看 python 版本，如图 5-4。

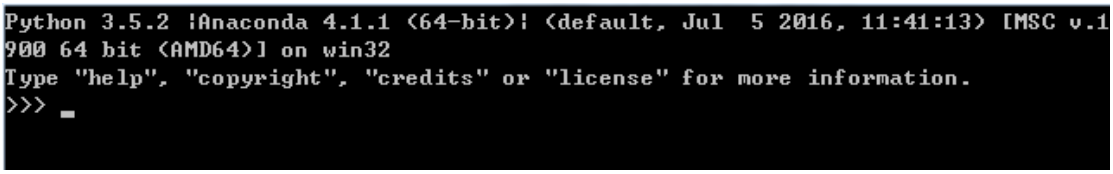


图 5-4 Python 环境

2. IDE：Spyder 2.3.9，开发界面如图 5-5。

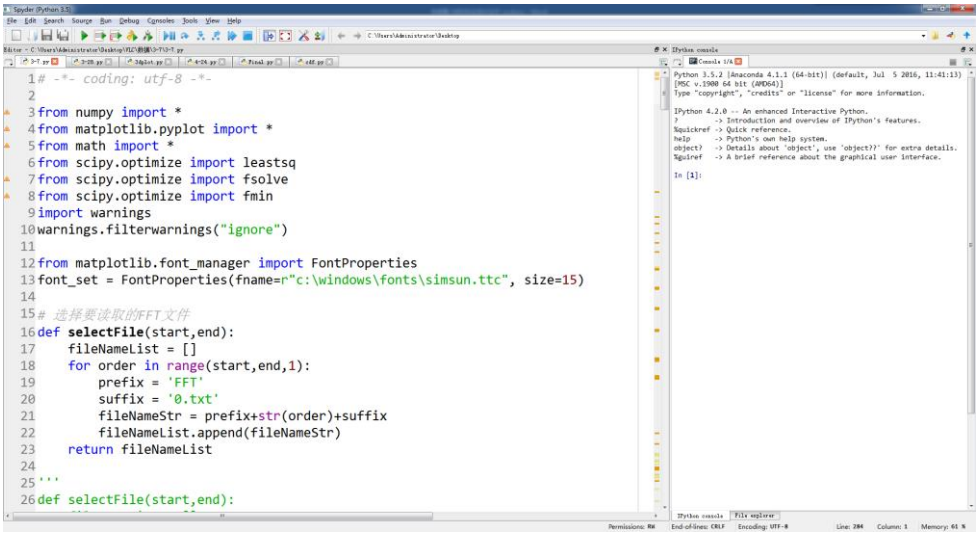


图 5-5 Python IDE

5.2 模块实现

5.2.1 可见光数据采集模块

1. 获取光强数据：

本软件通过调用 Android 光线传感器 API 以获得当前光强数据，具体实现方法如图 5-6。

```
public class SampleActivity extends Activity implements SensorEventListener {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        light_Sensor = sm.getDefaultSensor(Sensor.TYPE_LIGHT);
    }

    @Override
    protected void onResume() {
        super.onResume();
        sm.registerListener(this, light_Sensor, rate);
    }

    @Override
    protected void onStop() {
        // unregister listener
        sm.unregisterListener(this);
        super.onStop();
    }

    @Override
    public void onAccuracyChanged(Sensor sensor, int accuracy) {
        // TODO Auto-generated method stub
    }

    @Override
    public void onSensorChanged(SensorEvent event) {
        if(event.sensor.getType() == Sensor.TYPE_LIGHT) {
            lux = event.values[0];
            luxTV.setText("Light Intensity: "+lux);
        }
    }
}
```

图 5-6 获取光强数据部分实现代码

2. 平滑数据：

构造一个滑动窗口，每当传感器数值改变时就将光强数据记录到窗口中，并对当前窗口中的数据求和，当窗口大小达到 100 时，将 100 个数据之和平均得到平滑的光强数据，实现代码如图 5-7。

```
    @Override
    public void onSensorChanged(SensorEvent event) {
        if(event.sensor.getType() == Sensor.TYPE_LIGHT) {
            lux = event.values[0];
            if(slidingWindow.size() < 100){
                slidingWindow.add(lux);
            } else if(slidingWindow.size()  == 100){
                float sum = 0;
                for(int i = 0; i < slidingWindow.size(); i++){
                    sum += slidingWindow.get(i);
                }
                temp = sum / 100;
                currentLux = temp;
                luxTV.setText("Light Intensity: " + currentLux);
                slidingWindow.clear();
                temp = 0.0f;
            }
        }
    }
```

图 5-7 平滑数据部分实现代码

3. 录音：

录音功能的实现由 MyAudioRecord 这个线程类来完成，其类图如图 5-8。

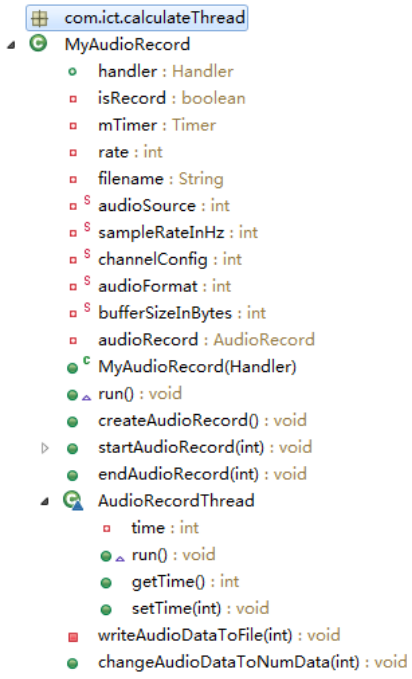


图 5-8 MyAudioRecord 类图

在打开录音线程之后，首先初始化音频录制的相关参数，包括采样时间、采样频率等。之后使用 audioRecord 中的 startRecording 方法进行录音。同时开启新的 AudioRecordThread 线程进行文件写入，其 run 方法执行 writeAudioDataToFile 方法将录制的音频文件写入 pcm 文件。同时设置一个定时器，时长为设置好的采样时间，当定时器到时时执行 endAudioRecord 方法，关闭定时器释放录音资源。

4. 读取数据：

读取数据主要负责从 pcm 文件中读取 byte 流数据到内存中并转化为 double 数组形式，并截取成多个 8K 的窗口以进行快速傅里叶变换。

方法是 MyAudioRecord 类中的 changeAudioDataToNumData 方法，其中调用 Utils 包中的 getArrayfromAndroid 方法，其代码实现如图 5-9。

```
153     /**
154      *
155      * @param file
156      * @return 从PCM文件获取数据流
157      */
158     public static double[] getArrayfromAndroid(String file) {
159         try {
160 //          BufferedInputStream bf = new BufferedInputStream(
161 //                  new FileInputStream(Environment
162 //                          .getExternalStorageDirectory() + "//audio//" + file));
163             BufferedInputStream bf = new BufferedInputStream(
164                     new FileInputStream(  file));
165             double[] data1;
166             try {
167                 byte[] data = new byte[1];
168                 List<Byte> allData = new ArrayList<Byte>();
169                 while (bf.read(data) != -1) {
170                     allData.add(data[0]);
171                 }
172                 data1 = new double[allData.size() / 2];
173                 for (int i = 0; i < allData.size() - 1; i += 2) {
174                     data1[i / 2] = ((allData.get(i + 1) << 8) | allData.get(i) & 0xff);
175                 }
176             } finally {
177                 bf.close();
178             }
179             return data1;
180         } catch (IOException e) {
181             throw new RuntimeException(e);
182         }
183     }
```

图 5-9 getArrayfromAndroid 实现代码

5. 快速傅里叶变换

快速傅里叶变换这部分的功能主要由 LightTrainer 类和 FFT 类来完成，其类图如图 5-10 和 5-11。

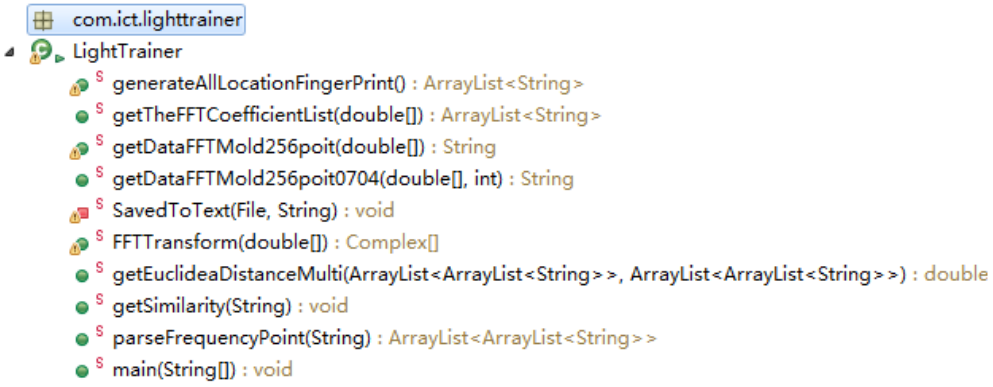


图 5-10 LightTrainer 类图

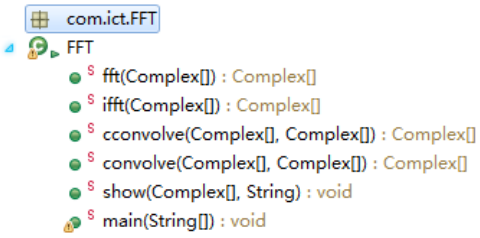


图 5-11 FFT 类图

使用其中的 getDataFFTMold256point0704 方法将截取好的窗口分别进行 FFT，该方法首先调用 FFTTransform 方法，然后该方法再调用 FFT 类中的 fft 方法将时域数据变为频域的复数形式的数据，fft 方法实现如图 5-12，然后求其模值。

```
21    // compute the FFT of x[], assuming its length is a power of 2
22    public static Complex[] fft(Complex[] x) {
23        int N = x.length;
24
25        // base case
26        if (N == 1)
27            return new Complex[] { x[0] };
28
29        // radix 2 Cooley-Tukey FFT
30        if (N % 2 != 0) {
31            throw new RuntimeException("N is not a power of 2");
32        }
33
34        // fft of even terms
35        Complex[] even = new Complex[N / 2];
36        for (int k = 0; k < N / 2; k++) {
37            even[k] = x[2 * k];
38        }
39        Complex[] q = fft(even);
40
41        // fft of odd terms
42        Complex[] odd = even; // reuse the array
43        for (int k = 0; k < N / 2; k++) {
44            odd[k] = x[2 * k + 1];
45        }
46        Complex[] r = fft(odd);
47
48        // combine
49        Complex[] y = new Complex[N];
50        for (int k = 0; k < N / 2; k++) {
51            double kth = -2 * k * Math.PI / N;
52            Complex wk = new Complex(Math.cos(kth), Math.sin(kth));
53            y[k] = q[k].plus(wk.times(r[k]));
54            y[k + N / 2] = q[k].minus(wk.times(r[k]));
55        }
56        return y;
57    }
```

图 5-12 fft 实现代码

6. 取出对应频点数据

将 FFT 并取模值后的数组中的第 375、558、748 个数取出，分别对应 2KHz、3KHz、4KHz 灯的幅度值。

7. 平滑数据

将多个 FFT 窗口所得结果使用之前介绍过的简单平均法进行平滑。

8. 传输数据、接收结果

与 Python 服务器端建立连接并传输数据，然后等待定位结果传回，其实现代码如图 5-13。

```
new Thread(new Runnable() {
    @Override
    public void run() {
        try {
            //获取wifi服务
            WifiManager wifiManager = (WifiManager) getSystemService(Context.WIFI_SERVICE);
            //判断wifi是否开启
            if (!wifiManager.isWifiEnabled()) {
                wifiManager.setWifiEnabled(true);
            }
            WifiInfo wifiInfo = wifiManager.getConnectionInfo();
            int ipAddress = wifiInfo.getIpAddress();
            String ip = intToIp(ipAddress);

            Socket client = new Socket(ip, 8888);
            BufferedWriter out = new BufferedWriter(new OutputStreamWriter(client.getOutputStream()));//发送光强数据
            DataInputStream in = new DataInputStream(client.getInputStream());// 读取返回的数据

            //传输光强
            out.write(lux2k + " " + lux3k + " " + lux4k);
            out.flush();

            //接收坐标
            byte[] buffer = new byte[100];
            in.read(buffer);
            coordinate = new String(buffer,"UTF-8");

            out.close();
            in.close();
            client.close();

            resultTV.setText("calculated result: " + coordinate);
            lux2k = "";
            lux3k = "";
            lux4k = "";

        } catch (Exception e) {
            System.out.println(e);
        }
    }
}).start();
```

图 5-13 Android 客户端数据传输实现代码

5.2.2 可见光数据处理模块

1. 接收数据：

与 Android 客户端建立连接并接收数据，其实现代码如图 5-14。

```
#服务器接收时域数据
hostname = socket.gethostname()
ip = socket.gethostbyname(hostname)
ADDR = (ip,8888)
BUFSIZE = 2048
recvSock = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
recvSock.bind(ADDR)
recvSock.listen(True)
print ("wait...")
conn,addr = recvSock.accept()
print ("send from ",addr)
tmp = conn.recv(BUFSIZE).decode()
lightString = tmp

a = lightString.split(" ")
lux2k = a[0]
lux3k = a[1]
lux4k = a[2]
```

图 5-14 Python 服务器数据传输实现代码

2. 整理数据：

分辨传来的数据是光强数据还是幅度数据，并将其装入对应数组。

3. 参数拟合：

使用 SciPy.optimize.leastsq 方法对参数进行拟合，其使用方法如图 5-15。

```
#待拟合的函数，x是变量，p是参数
def fun1(x,p):
    a = p
    return a*x

#计算真实数据和拟合数据之间的误差，p是待拟合的参数，x和y分别是对应的真实数据
def residuals1(p,x,y):
    return fun1(x,p) - y

def leastSuareFit1(distList,luxList):
    x1 = array(distList)
    y1 = array(luxList)
    p0 = 0
    #调用拟合函数，第一个参数是需要拟合的差值函数，第二个是拟合初始值，第三个是传入函数的其他参数
    r = leastsq(residuals1,p0,args=(x1, y1))
    return r[0][0];
```

图 5-15 leastsq 调用方法

5.2.3 手机姿态计算模块

1. 获取方向数据：

本软件通过调用 Android 方向传感器 API 以获得当前航向角、俯仰角、横滚角数据，具体实现方法与调用光线传感器 API 类似。

2. 平滑数据：

与之前介绍的平滑数据过程实现方法类似。

3. 计算手机法向量：

将初始法向量(0,0,1)与旋转矩阵相乘获得当前手机屏幕的方向量，具体实现方法如图 5-13。需要使用 Matrix 类，其类图如图 5-14。

```
493    private void calculateNormalVector(){
494
495        double initialNormalVector[][] = {{0},{0},{1}};
496
497        double z[][] = {{Math.cos(orientation[0]), Math.sin(orientation[0]), 0},
498                {-Math.sin(orientation[0]), Math.cos(orientation[0]), 0},
499                {0, 0, 1}};
500
501        double x[][] = {{1, 0, 0},
502                {0, Math.cos(orientation[1]), Math.sin(orientation[1])},
503                {0, -Math.sin(orientation[1]), Math.cos(orientation[1])}};
504
505        double y[][] = {{Math.cos(orientation[2]), 0, -Math.sin(orientation[2])},
506                {0, 1, 0}, {Math.sin(orientation[2]), 0, Math.cos(orientation[2])}};
507
508        Matrix rotateWithZMatrix = new Matrix();
509        Matrix rotateWithXMatrix = new Matrix();
510        Matrix rotateWithYMatrix = new Matrix();
511        Matrix temp = new Matrix();
512
513        sensorNormalVector.setArray(initialNormalVector);
514        rotateWithZMatrix.setArray(z);
515        rotateWithXMatrix.setArray(x);
516        rotateWithYMatrix.setArray(y);
517
518        temp = rotateWithZMatrix.multip(rotateWithXMatrix);
519        temp = temp.multip(rotateWithYMatrix);
520        sensorNormalVector = temp.multip(sensorNormalVector);
521
522
523        vectorTV.setText("Normal vector of sensor: " + sensorNormalVector.getArray()[0][0]
524                +", "+sensorNormalVector.getArray()[1][0]+", "+sensorNormalVector.getArray()[2][0]
525
526    }
```

图 5-13 计算屏幕法向量实现代码

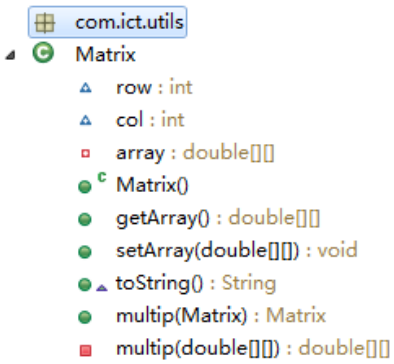


图 5-14 Matrix 类图

4. 传输数据：

实现方法同可见光数据采集模块。

5.2.4 多边定位模块

1. 接收数据：

实现方法同可见光数据处理模块。

2. 三边定位

首先判断接收到的数据是光强数据还是幅度数据，如果是幅度数据需要用可见光数据处理模块提供的参数将幅度变为光强数据。其次构造非线性方程组，并使用 SciPy.optimize.fsolve 方法求解，其实现方法如图 5-15。

```
#计算方程组误差的函数
def fs(x):  # 参数x接受一个数组
    x0,y0,z0 = x.tolist() # 将数组转换为python的标准浮点数列表
                          # 浮点数类型缩短时间
    #print(str(x0)+' '+str(y0)+' '+str(z0))
    return [
        lux_2k-A2k*(z0-coordinate_2k[2])**2/((x0-coordinate_2k[0])**2+(y0-coord
        lux_3k-A3k*(z0-coordinate_3k[2])**2/((x0-coordinate_3k[0])**2+(y0-coord
        lux_4k-A4k*(z0-coordinate_4k[2])**2/((x0-coordinate_4k[0])**2+(y0-coord
    ]

#进行优化，求出坐标
init_paras = [0.1,0.1,1.0]
result = fsolve(fs,init_paras)
```

图 5-15 fsolve 调用方法

3. 绘制结果图

计算出定位结果后，使用 mpl_toolkits.mplot3d.Axes3D 以及 matplotlib.pyplot 两个包来绘制 3D 结果图，实现方法如图 5-16。

```
#画出3D定位图
fig = plt.figure()
ax = fig.add_subplot(111,projection='3d')
ax.set_xlim(0, 90)
ax.set_ylim(0, 58)
ax.set_zlim(0, 120)
ax.set_xlabel('Length')
ax.set_ylabel('Width')
ax.set_zlabel('Height')

#2K灯
ax.scatter(coordinate_2k[0],coordinate_2k[1],coordinate_2k[2],c='r',marker='o')
label="   2K" + coord_2k
zdir=None
ax.text(coordinate_2k[0],coordinate_2k[1],coordinate_2k[2],label,zdir)
#3K灯
ax.scatter(coordinate_3k[0],coordinate_3k[1],coordinate_3k[2],c='r',marker='o')
label="   3K" + coord_3k
zdir=None
ax.text(coordinate_3k[0],coordinate_3k[1],coordinate_3k[2],label,zdir)
#4K灯
ax.scatter(coordinate_4k[0],coordinate_4k[1],coordinate_4k[2],c='r',marker='o')
label="   4K" + coord_4k
zdir=None
ax.text(coordinate_4k[0],coordinate_4k[1],coordinate_4k[2],label,zdir)
#Sensor
ax.scatter(a[0],a[1],a[2],c='b',marker='o')
label="   Sensor" + b
zdir=None
ax.text(a[0],a[1],a[2],label,zdir)
```

图 5-16 绘图部分实现代码

4. 返回结果：

使用 socket 将结果返回至可见光数据采集模块。

5.2.5 计步器定位模块

1. 获取航向角数据：

实现方法同手机姿态计算模块。

2. 平滑数据：

实现方法同可见光数据采集模块。

3. 计步：

计步器模块主要由 StepDetectionProvider 类实现，其类图如图 5-17。

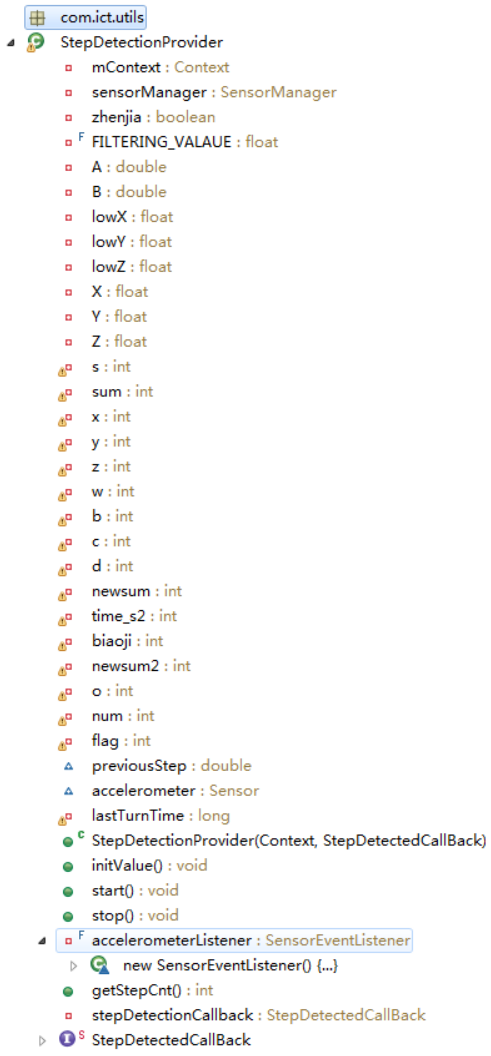


图 5-17 StepDetectionProvider 类图

使用回调函数进行调用，如图 5-18。

```
212    private void startStepCount(){
213        StepDetectedCallBack stepDetectedCallBack = new StepDetectedCallBack() {
214
215            @Override
216            public void onStepDetected(int stepCount) {
217                stepCountTV.setText("step count: " + stepCount);
218                currentCoords = calculateCoords(currentCoords, currentDir);
219                coordsTV.setText("current coordinates: (" + currentCoords[0] + ", " + currentCoords[1] + ")");
220            }
221        };
222        stepDetectionProvider = new StepDetectionProvider(this,
223                stepDetectedCallBack);
224        stepDetectionProvider.start();
225    }
```

图 5-18 回调函数实现代码

4. 设置步长：

用户设置步长，正常人步长在 0.5m 左右。

5. 推算坐标：

计步器步数每增加 1 步，就对坐标进行 1 次更新，其实现方法如图 5-19。

```
227    private float[] calculateCoords(float[] currentCoords, float currentDir){
228        double r = Math.toRadians(currentDir);
229        double x = Math.sin(r);
230        double y = Math.cos(r);
231
232        currentCoords[0] += x * stepLength;
233        currentCoords[1] += y * stepLength;
234        return currentCoords;
235    }
```

图 5-19 回调函数实现代码

5.2.6 单灯校准模块

1. 持续检测对应频率幅度值：

不断调用可见光数据采集模块获取幅度值。

2. 位置校准：

如果当前某个灯对应的幅度值超过了预先设置的阈值，则将手机的坐标位置校准至灯的位置。

5.3 成果展示

5.3.1 可见光数据采集以及手机姿态计算模块

打开 App 选择多灯测距定位模式后界面如图 5-20，记录光强数据时如图 5-21。

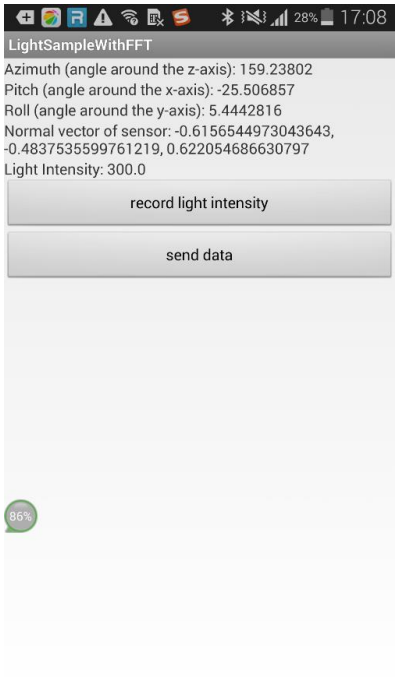
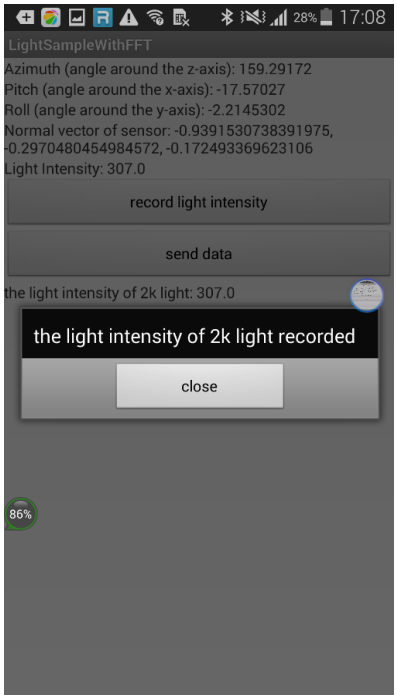


图 5-20 模块 1、3 截图



5-21 模块 1、3 截图

5.3.2 可将光数据处理模块

该模块拟合得到的结果如图 5-22 至 5-27。

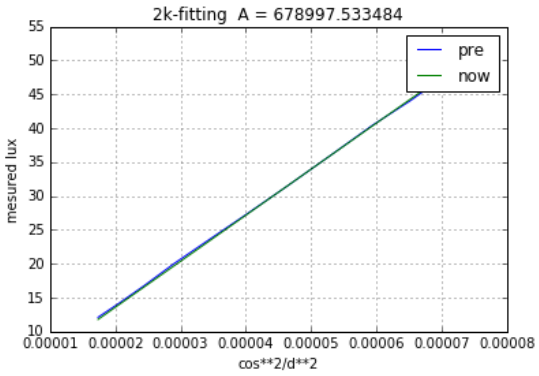


图 5-22 2K 灯常数 C 拟合结果

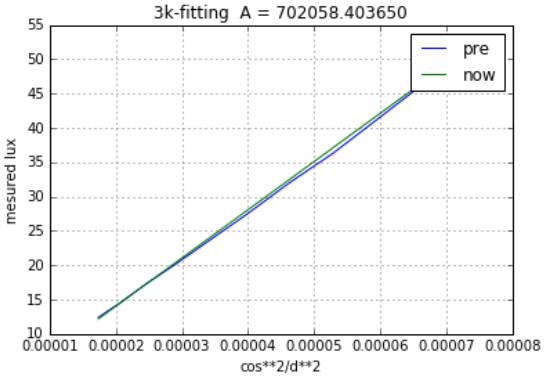


图 5-23 3K 灯常数 C 拟合结果

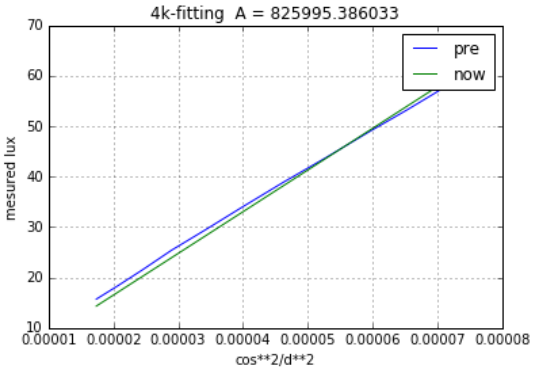


图 5-24 4K 灯常数 C 拟合结果

(3.7034772454881788e-07, -0.001285785957164003, 12.160357260354676)

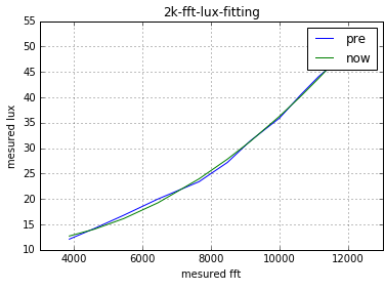


图 5-25 2K 灯幅度光强关系拟合结果

(3.7000816274180331e-07, 0.0050354321325259879, 5.5668500629334936)

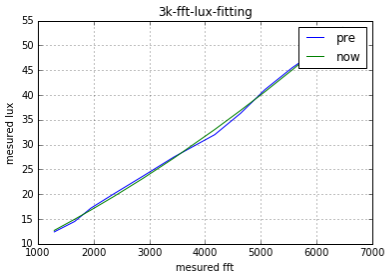


图 5-26 3K 灯幅度光强关系拟合结果

(-3.575461919832535e-07, 0.012649408185320784, -1.4957063680872089)

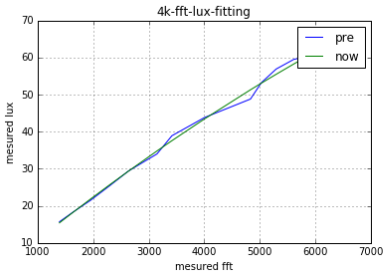


图 5-27 4K 灯幅度光强关系拟合结果

5.3.3 多边定位模块

该模块计算得到的坐标以及绘制的结果图如图 5-28 所示。

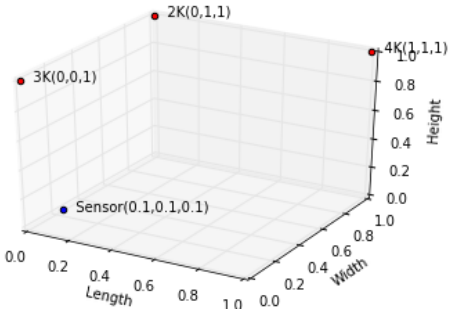


图 5-28 定位结果图

5.3.4 计步器定位以及单灯校准模块

打开 App 后，选择计步器与单灯校准融合定位模式后，界面如图 5-29，5-30 所示。

图 5-29 模块 5、6 截图

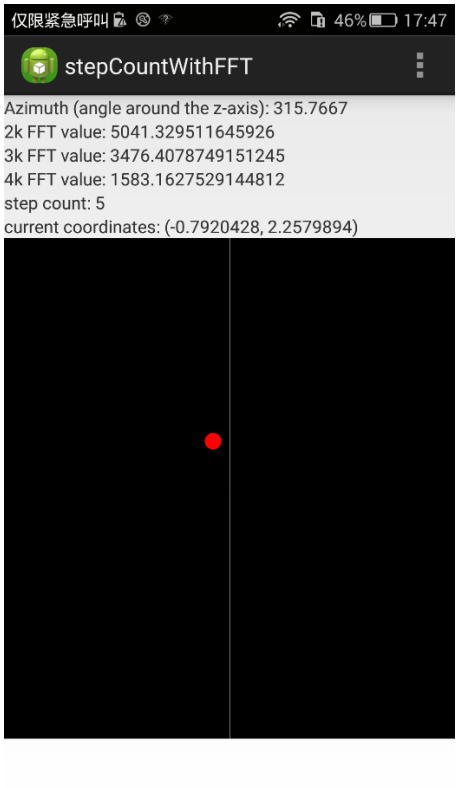


图 5-30 模块 5、6 截图

5.4 本章小结

本章介绍了本软件关键部分的代码实现和成果展示，分为 6 大模块：可见光数据采集模块、可见光数据处理模块、手机姿态计算模块、多边定位模块、计步器定位模块、单灯校准模块。介绍了开发环境、实现方法并展示了本软件的截图。

第六章　可见光测距定位软件的实验测试

本章重点描述了测试环境、功能测试、性能测试以及准确性测试。

6.1　测试环境

手机 App 客户端测试环境如表 6-1：

表 6-1 客户端测试环境

设备名称	Sumsung Galaxy S5
CPU 型号	高通骁龙 801
CPU 主频	2.5GHz
内存	RAM：2G，ROM：16G
操作系统	Android 4.4.2

电脑 Python 服务器测试环境如表 6-2：

表 6-2 服务器测试环境

设备名称	长城
CPU 型号	Intel Core i5-6500
CPU 主频	3.2GHz
内存	8GB
操作系统	Windows 7 64 位

6.2 功能测试

6.2.1 可见光数据采集模块功能测试

1. 获取光强数据功能测试，测试用例如表 6-3 所示：

表 6-3 获取光强数据功能测试

用例编号	001
用例目的	该用例用于测试获取光强数据功能。
描述	该用例用于测试可见光数据采集模块是否能正确调用光线传感器 API 获取光强数据。
前提条件	多灯测距定位模式下使用光强进行定位并点击“记录光强”按钮。
测试步骤	观测界面上显示的光强数值。

续上表

预期结果	在同一位置的光强数值保持稳定，手机离近光源数值变大，反之数值变小。
测试结果	测试通过。

2. 平滑数据功能测试，测试用例如表 6-4 所示：

表 6-4 平滑数据功能测试

用例编号	002
用例目的	该用例用于测试平滑数据功能。
描述	该用例用于测试可见光数据采集模块对于数据的平滑功能是否正确，平滑功能在本软件的许多地方用到，例如对于光强、幅度以及方向数据的处理。
前提条件	获取到数值之后。
测试步骤	观测界面上显示的或文件记录下的数值。
预期结果	在同一环境条件下的平滑后的数值赢保持稳定。
测试结果	测试通过。

3. 录音功能测试，测试用例如表 6-5 所示：

表 6-5 录音功能测试

用例编号	003
用例目的	该用例用于测试录音功能。
描述	该用例用于测试可见光数据采集模块是否能正确调用录音 API 进行录音。
前提条件	多灯测距定位模式下使用幅度进行定位并点击“记录幅度”按钮或选择计步器与单灯校准融合定位模式之后。
测试步骤	将录制好的 pcm 文件取出，并使用播放器播放。
预期结果	可以听到声音。
测试结果	测试通过。

4. 读取数据功能测试，测试用例如表 6-6 所示：

表 6-6 读取数据功能测试

用例编号	004
用例目的	该用例用于测试读取数据相关功能。
描述	该用例用于测试可见光数据采集模块能否正确地从 pcm 文件中将 byte 流数据读取到内存中，并以 double 数组的形式存储。
前提条件	录音结束。
测试步骤	编码将读取之后的 double 数组写入文件，并观测文件中的数据。

续上表

预期结果	采样时间为 1s，采样频率为 44.1K 的情况下，文件中应有 44100 个数据。
测试结果	测试通过。

5. 快速傅里叶变换功能测试，测试用例如表 6-7 所示：

表 6-7 快速傅里叶变换功能测试

用例编号	005
用例目的	该用例用于测试快速傅里叶变换相关功能。
描述	该用例用于测试可见光数据采集模块是否能正确对数据进行快速傅里叶变换。
前提条件	读取数据结束。
测试步骤	编码将手机上的快速傅里叶变换之后的结果写入文件，并将时域数据放进 Matlab 进行快速傅里叶变换，比较其结果。
预期结果	对相同的时域数据，在手机上以及在 Matlab 里进行的 FFT 结果应相同。
测试结果	测试通过

6. 取出对应频点数据功能测试，测试用例如表 6-8 所示：

表 6-8 取出对应频点数据功能测试

用例编号	006
用例目的	该用例用于测试取出对应频点数据相关功能。
描述	该用例用于测试可见光数据采集模块是否能正确取出 FFT 之后的对应频点的幅度数据。
前提条件	快速傅里叶变换结束。
测试步骤	记录取出的对应频点的幅度数据，并将其与之前写入的 FFT 文件进行比对。
预期结果	结果相同。
测试结果	测试通过。

7. 传输与接收数据功能测试，测试用例如表 6-9 所示：

表 6-9 传输与接收数据功能测试

用例编号	007
用例目的	该用例用于测试传输与接收数据功能。
描述	该用例用于测试本软件的客户端与服务器之间的通信是否正常。
前提条件	打开服务器并点击客户端上的“传输数据”按钮。
测试步骤	记录客户端或服务器发送以及接收到的数据，并进行比对。

续上表

预期结果	发送的与接收的数据相同。
测试结果	测试通过

6.2.2 可见光数据处理模块功能测试

1. 整理数据功能测试，测试用例如表 6-10 所示：

表 6-10 整理数据功能测试

用例编号	008
用例目的	该用例用于测试整理数据相关功能。
描述	该用例用于测试可见光数据处理模块能否分辨传来的数据并将其放入对应的数组当中。
前提条件	接收到数据。
测试步骤	记录接收到的数据并打印其数据形式和对应数组，比较对应结果。
预期结果	实际的数据形式和对应数组应与理论相同。
测试结果	测试通过。

2. 参数拟合功能测试，测试用例如表 6-11 所示：

表 6-11 参数拟合功能测试

用例编号	009
用例目的	该用例用于测试参数拟合功能。
描述	该用例用于测试可见光数据处理模块能否正确拟合出相关参数。
前提条件	整理数据结束。
测试步骤	使用 Python 绘图库将带入参数的结果与实际数据绘制在同一张图中。
预期结果	两条曲线基本接近。
测试结果	测试通过。

6.2.3 手机姿态计算模块功能测试

1. 获取方向数据功能测试，测试用例如表 6-12 所示：

表 6-12 获取方向数据功能测试

用例编号	010
用例目的	该用例用于测试获取方向数据相关功能。
描述	该用例用于测试手机姿态计算模块是否能够正确调用方向传感器 API 获取方向数据。

续上表

前提条件	选择多灯测距定位模式。
测试步骤	首先对手机进行 8 字校准法，然后不同角度地翻转手机并观察方向数据的数值。
预期结果	航向角范围为 0°至 360°。北为 0°，东为 90°，南为 180°，西为 270°。俯仰角范围为-180°至 180°。当 z 轴向 y 轴转动时，角度为正值。横滚角范围为-90°至 90°。当 x 轴向 z 轴移动时，角度为正值。
测试结果	测试通过。

6.2.4 多边定位模块功能测试

1. 三边定位功能测试，如表 6-13 所示：

表 6-13 三边定位功能测试

用例编号	011
用例目的	该用例用于测试三边定位功能。
描述	该用例用于测试多边定位模块能否正确地求解非线性方程组得到定位结果。
前提条件	多边定位模块接收数据完毕。
测试步骤	定位时记录实际坐标位置，与计算结果进行比较。
预期结果	实际坐标与计算结果接近。
测试结果	测试通过。

2. 绘制结果图功能测试，如表 6-14 所示：

表 6-14 绘制结果图功能测试

用例编号	012
用例目的	该用例用于测试绘制结果图功能。
描述	该用例用于测试多边定位模块能否正确地将结果以 3D 图的形式展示。
前提条件	定位结果计算完毕
测试步骤	观察绘制的 3D 图
预期结果	绘制的图中的坐标与定位结果相同。
测试结果	测试通过

6.2.5 计步器定位模块功能测试

1. 计步功能测试，如表 6-15 所示：

表 6-15 团队管理功能测试

用例编号	013
用例目的	该用例用于测试计步功能。
描述	该用例用于测试计步器定位模块能否准确地计步。
前提条件	进入计步器与单灯校准融合定位模式。
测试步骤	拿着手机走路。
预期结果	步数应与实际相同。
测试结果	测试通过

2. 推算坐标功能测试，如表 6-16 所示：

表 6-16 好友管理功能测试

用例编号	014
用例目的	该用例用于推算坐标功能。
描述	该用例用于测试计步器定位模块能否准确地推算坐标。
前提条件	航向角、步数、步长数据采集完毕。
测试步骤	拿着手机走路，每走一步就记录一次当前坐标，并与推算的坐标进行比对。
预期结果	推算的坐标应与记录的坐标接近。
测试结果	测试通过。

6.2.6 单灯校准模块功能测试

1. 持续检测对应频率幅度值功能测试，如表 6-17 所示：

表 6-17 持续检测对应频率幅度值功能测试

用例编号	015
用例目的	该用例用于测试持续检测对应频率幅度值功能。
描述	该用例用于测试单灯校准模块能否正确地调用可见光数据采集模块并返回正确的幅度值。
前提条件	进入计步器与单灯校准融合定位模式。
测试步骤	将结果输出在屏幕上，并观察结果。
预期结果	开某个灯的时候其对应幅度值增大，并符合变化规律。
测试结果	测试通过

2. 位置校准功能测试，如表 6-18 所示：

表 6-18 位置校准功能测试

用例编号	016
用例目的	该用例用于测试位置校准功能。
描述	该用例用于测试单灯校准模块能否正确地校准坐标。
前提条件	对应频率的幅度值大于设定阈值。
测试步骤	拿着手机走过灯照射范围，观察坐标数据。
预期结果	当走过灯的照射范围时，坐标应被校准至灯的坐标。
测试结果	测试通过。

6.3 性能测试

6.3.1 可靠性测试

本节使用 Eclipse 中的 DDMS 对本软件的 App 端的 CPU 占用率和内存泄漏进行了测试，当使用多灯测距定位模式时，CPU 占用率保持在 50%左右，内存总量维持在 1.3MB 左右，如图 6-1 所示；当使用计步器与单灯校准融合定位模式时，CPU 占用率在 70%左右，内存总量维持在 1.5MB 左右，如图 6-2 所示。可以保障软件的可靠性。

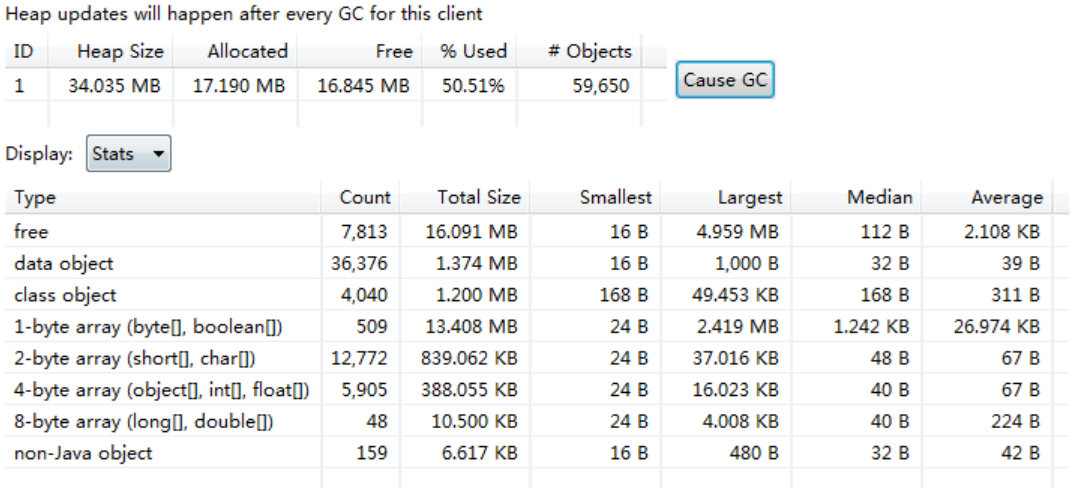


图 6-1 多灯测距定位模式 CPU 及内存占用情况

Heap updates will happen after every GC for this client

ID	Heap Size	Allocated	Free	% Used	# Objects
1	23.656 MB	17.066 MB	6.590 MB	72.14%	54,946

Cause GC

Display: Stats

Type	Count	Total Size	Smallest	Largest	Median	Average
free	14,675	5.334 MB	16 B	956.664 KB	64 B	381 B
data object	49,072	1.529 MB	16 B	1,000 B	32 B	32 B
class object	4,067	1.205 MB	168 B	49.453 KB	168 B	310 B
1-byte array (byte[], boolean[])	507	13.301 MB	24 B	2.419 MB	1.242 KB	26.863 KB
2-byte array (short[], char[])	12,820	841.984 KB	24 B	37.016 KB	48 B	67 B
4-byte array (object[], int[], float[])	5,952	510.164 KB	24 B	32.023 KB	40 B	87 B
8-byte array (long[], double[])	56	404.805 KB	24 B	330.023 KB	40 B	7.229 KB
non-Java object	141	6.078 KB	16 B	480 B	32 B	44 B

图 6-2 计步器与单灯校准融合定位模式 CPU 及内存占用情况

6.3.2 可移植性测试

将本软件的 App 安装在不同型号的手机上，都能正常的运行，手机包括 HUAWEI Mate7、Sumsung Galaxy S3、Google Nexus 6p。

6.4 准确性测试

6.4.1 多灯测距定位模式准确性测试

使用光强数据进行定位的结果如表 6-19。

表 6-19 多灯光强定位结果表

计算得到的坐标(cm)			实际测得的坐标(cm)			误差(cm)
x	y	z	x	y	z	error
76.51995	62.20628	-1.38371	84	58.5	4	9.933374
65.54201	63.98567	-0.22808	74	58.5	4	10.93192
57.58676	62.82459	0.800365	64	58.5	4	8.370746
48.54768	63.46996	2.562468	54	58.5	4	7.516303
38.62577	62.95466	3.576874	44	58.5	4	6.993241
29.35625	64.00082	5.466054	34	58.5	4	7.346618
21.86932	63.2078	5.025588	24	58.5	4	5.268297
13.34138	63.58001	5.642977	14	58.5	4	5.379562
74.70062	43.57743	-3.37421	84	44.5	4	11.90414
66.56524	44.6783	-1.55088	74	44.5	4	9.28007
57.23739	44.05044	-0.18413	64	44.5	4	7.965048

续上表

47.82519	44.38902	1.314544	54	44.5	4	6.734409
39.33462	44.05363	1.957592	44	44.5	4	5.112386
29.88254	44.54905	3.38706	34	44.5	4	4.16312
22.29925	43.90633	3.145586	24	44.5	4	1.993746
12.60193	44.2479	4.500426	14	44.5	4	1.506183
60.677	24.07566	-0.60087	64	20	4	6.987229
49.33692	24.28485	0.057053	54	20	4	7.459966
39.90422	25.67286	0.720007	44	20	4	7.727557
29.15403	26.34038	2.117656	34	20	4	8.199211
28.0676	21.63101	2.666725	30	20	0	3.6750212
26.43721	22.53802	4.253624	30	20	0	6.1015003
21.38258	23.82332	6.487944	30	20	0	11.444259
29.72773	45.47219	1.450664	30	40	0	5.6677561
32.1844	45.41214	1.443144	30	40	0	6.0121155
29.72773	45.47219	1.450664	30	40	0	5.6677561
64.52853	24.89986	2.485884	60	20	0	7.1200973
60.95307	24.47609	3.262854	60	20	0	5.6204907
65.88142	24.31101	3.13927	60	20	0	7.9391998
54.15768	47.90416	3.76415	60	40	0	10.525079
62.33256	48.05445	4.3426	60	40	0	9.4431598
54.15768	47.90416	3.76415	60	40	0	10.525079

其误差 CDF 图如图 6-3，可以看出平均误差在 7cm 左右，最大误差不超过 12cm，定位结果较为理想。

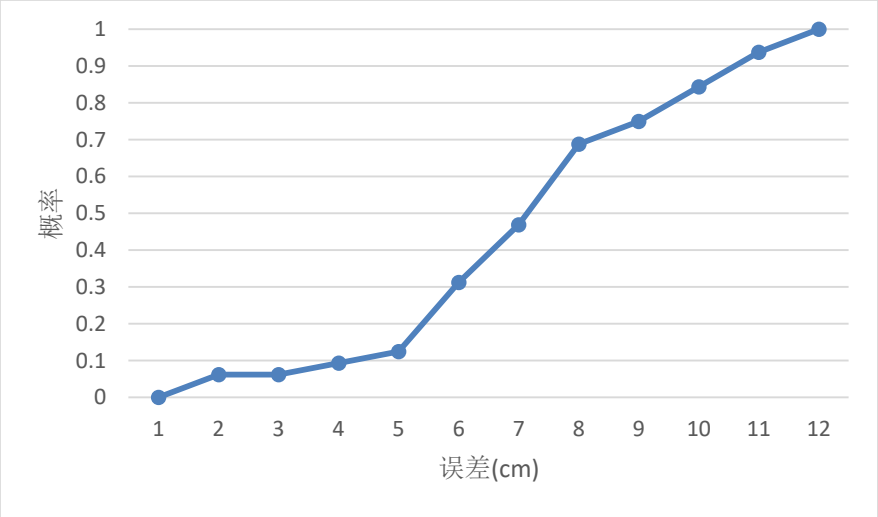


图 6-3 多灯光强定位误差 CDF 图

选取一部分定位结果绘制 3D 结果图，如图 6-4。以及平面结果图，如图 6-5。

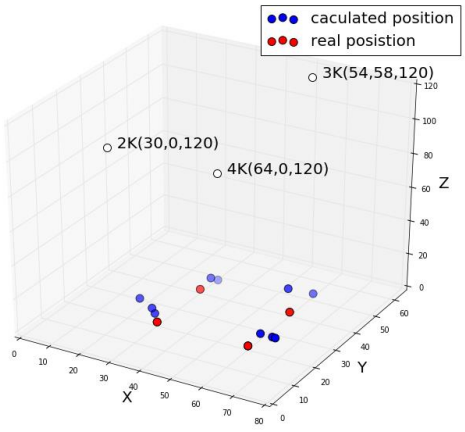


图 6-4 多灯光强定位 3D 结果图

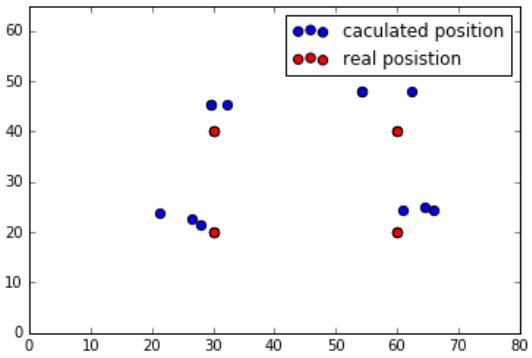


图 6-5 多灯光强定位平面结果图

使用幅度数据进行定位的结果如表 6-20。

表 6-20 多灯幅度定位结果表

计算得到的坐标(cm)			实际测得的坐标(cm)			误差(cm)
x	y	z	x	y	z	error
76.29803	70.95326	-2.14177	84	58.5	4	15.87845
61.02195	68.60604	-2.33979	74	58.5	4	17.62823
46.82411	65.13608	0.727717	64	58.5	4	18.70178
37.52354	63.06475	5.296536	54	58.5	4	17.14618
30.37767	62.93919	7.886526	44	58.5	4	14.84518
23.23778	62.85422	10.27571	34	58.5	4	13.19731
16.64834	62.80678	12.23286	24	58.5	4	11.848
10.76128	64.41181	15.4516	14	58.5	4	13.28826
61.50043	48.36247	1.157489	84	44.5	4	23.00498
54.29395	47.35647	1.380453	74	44.5	4	20.08358
46.81077	47.42079	2.749428	64	44.5	4	17.4804
40.7331	46.88853	3.56366	54	44.5	4	13.48726
34.52657	46.99016	5.548611	44	44.5	4	9.9169
25.3705	47.27688	8.870994	34	44.5	4	10.29106
18.59072	47.35573	11.15687	24	44.5	4	9.414688
11.27175	47.89164	14.77878	14	44.5	4	11.62449
44.76955	31.68909	0.747732	64	20	4	22.73812
39.94415	31.97341	3.942436	54	20	4	18.46436

续上表

32.30063	32.16373	7.647469	44	20	4	17.2666
25.62886	31.36444	8.81674	34	20	4	14.914

其误差 CDF 图如图 6-4，可以看出平均误差在 15.5cm 左右，最大误差不超过 24cm。

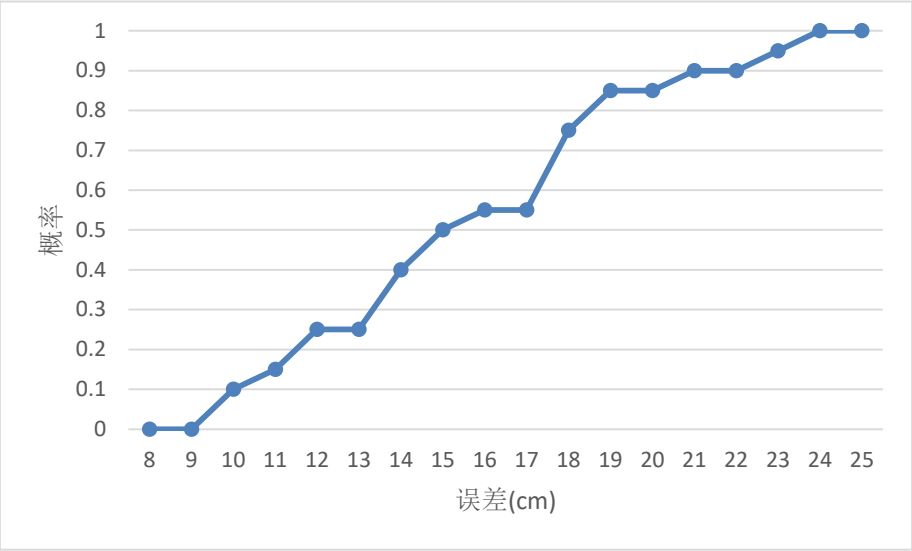


图 6-4 多灯幅度定位误差 CDF 图

6.4.2 计步器与单灯校准融合定位模式准确性测试

使用计步器进行定位实验，在实验之前对手机罗盘进行 8 字校准法，并设置与实验者近似的步长。其定位结果如表 6-21，该实验结果去掉了因为计步器模块计步不准确或罗盘未较准而产生的异常数据，因此实验精度相较实际精度偏高。

表 6-21 计步器定位结果表

推算坐标(cm)		实际坐标(cm)		误差(cm)
x	y	x	y	error
49.3	-7.9	56	0	10.35857
99.7	5.4	104	0	6.902898
149.6	7.5	150	0	7.510659
-99	-12.7	-105	0	14.046
49.7	5.4	50	0	5.408327
99.4	10.7	100	0	10.71681
149.4	10.8	150	0	10.81665
199.3	13.8	200	0	13.81774
249.3	15.6	250	0	15.6157

续上表

299.2	13.4	300	0	13.42386
349.1	10.7	350	0	10.73778
-48.7	10.8	-50	0	10.87796
-98.3	17.5	-100	0	17.58238
-148.2	21.3	-150	0	21.37592
-198.1	23.9	-200	0	23.9754
-248.1	24.3	-250	0	24.37417
-348.1	22.3	-350	0	22.3808
-397.7	16.2	-400	0	16.36246
-447.2	8.8	-450	0	9.234717
-2.1	-49.9	0	-50	2.10238
4.8	-99.8	0	-100	4.804165
9.8	-149.5	0	-150	9.812747
10.2	-199.5	0	-200	10.21225
13.6	-249.4	0	-250	13.61323

其误差 CDF 图如图 6-4，可以看出平均误差在 12.75cm 左右，最大误差不超过 24cm，由于坐标累加的原因，累计误差会比较大。后续提高计步器模块的精度、航向角的精度以及添加动态测算步长的功能后，可以使实际精度更加贴近实验精度。

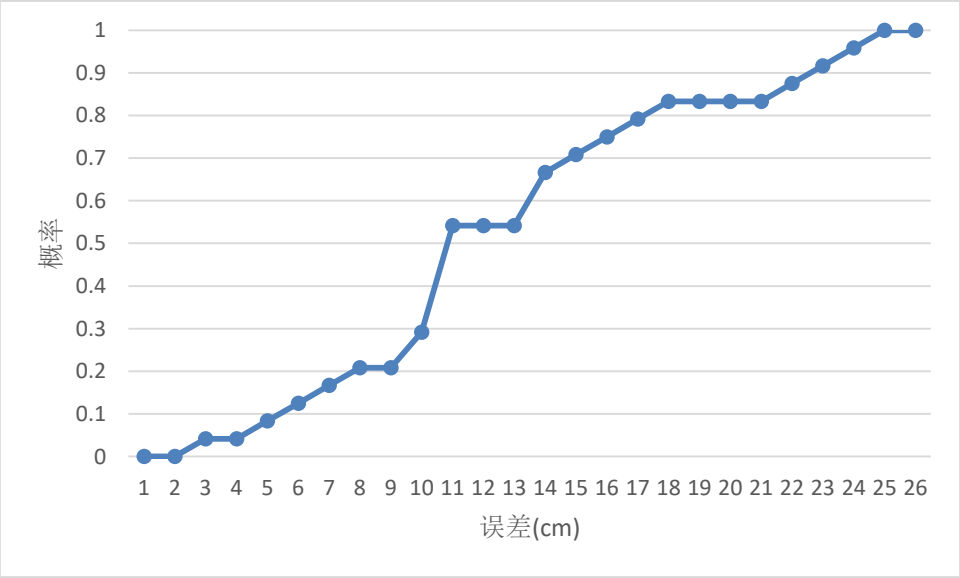


图 6-4 计步器定位误差 CDF 图

对于测试机 HUAWEI MATE 7 来说，当光源安装在距离手机大约 1.8m 的位置时，2K 灯幅度阈值设置为 $2.6E^7$，误差在 11cm 以内；3K 灯幅度阈值设置为 $4.0E^7$，误差在 7cm 以内；4K 灯幅度阈值设置为 $2.2E^7$,误差在 9cm 以内。

6.5 本章小结

本章介绍了对基于 Android 平台的可见光测距定位软件的客户端测试，主要包括功能测试、性能测试以及准确性测试。本软件功能齐全，性能达标，定位精度可以达到厘米级。

第七章 总结与展望

7.1 项目总结

本文致力于基于 Android 平台的可见光测距定位软件的研究和实现工作。在本项目中，作者对 GPS 定位以及多种室内定位技术进行了一定的了解和比较，并着重于可见光测距定位技术的研究。在广泛阅读了可见光定位技术的相关论文，比较了不同定位算法之后，选定基于测距的定位方式作为本软件的实现方式，并对需要用到的相关技术进行研究。在软件开发方面，本软件主要分为两种定位模式：多灯定位模式和计步器与单灯校准融合定位模式。功能模块主要有：可见光数据采集模块、可见光数据处理模块、手机姿态计算模块、多灯定位模块、计步器定位模块、单灯校准模块等共 6 个模块。本软件满足功能需求，测试结果优异，多灯定位模式和计步器与单灯校准融合定位模式的定位误差均可以达到厘米级，实验中的数据误差均不超过 0.3m。

由于作者在之前的本科学习中没有接触过本次毕设的相关研究内容，其中许多知识和概念对于作者都是陌生的，学习起来也比较困难，因此本次毕设历时较长，约为 11 个月（从 2016 年 8 月至今）。相关文献的阅读、实现算法的选定、项目的需求分析、模块的设计、程序代码的实现、测试与评估，所有工作均由本人独立完成。

由于对于硬件以及通信领域的陌生，完成本次毕设的过程中遇到了很多困难。例如，在最开始时，因为不知道使用的华为测试机的耳机接口顺序与其他手机不同，将可见光数据采集模块的接线焊错了位置，导致接收不到数据。又例如，在多光源同时照射的情况下，光敏电阻的响应是非线性的，导致 FFT 之后的幅值产生偏差。这些问题虽然耗费了作者很长的时间，但是所幸在指导老师以及学长的悉心指导下基本都得以解决。

本次毕设中，我编写代码超过 5600 行，调研了大量的相关文献，为学习相关技术翻阅很多博客、书籍。而且中期检查之后新添加了第二种定位模式，需求进行了变更，工作量和工作难度都可谓很大。但是，经过本次毕设之后，本人对科研有了更深层次的理解，科研是一门苦差事，在这个过程中经常会一直面对未知不熟悉的领域，需要一直不停的进行学习，同时会不断地遇到问题、分析问题、解决问题，甚至有很多问题是暂时无法解决的。另外，本次毕设对于本人的软件工程能力和开发能力都得到了提升，为本人今后出国进行科研和学习打下良好基础。

7.2 下阶段研究工作

目前，基于 Android 平台的可见光测距定位软件已经完成了需求、设计、实现、测试、评估等过程，功能完善，结果准确。但是本项目还存在着一些问题，没有妥善地解决，这一部分也是下阶段的主要研究工作：

1. 由于光敏电阻元件对于多光源的响应非线性的原因，在多光源同时照射的情况下，FFT 之后得到的幅值会变低，因此现在本项目在采集多光源数据时采用分时复用的方式。下阶段会对更多的传感器进行研究，选用精度更高，并且可以对多光源进行响应的元件；
2. 目前，在本项目的计步器与单灯校准融合定位模式中，步长是根据经验设置的，然而每个人的步长不同，因此会产生较大的误差。下阶段会对惯性导航进行研究，研究出动态测算步长的方法并实现。

参考文献

[1] Rycroft M J. Understanding GPS. Principles and Applications[J]. Journal of Atmospheric and Solar-Terrestrial Physics, 1997, 59(5):598-599.

[2] M Yasir, SW Ho, BN Vellambi. Indoor Localization Using Visible Light and Accelerometer [J]. Journal of Lightwave Technology, 2014.

[3] L Li, P Hu, C Peng, G Shen, F Zhao. Epsilon: A Visible Light Based Positioning System [J]. Usenix Org, 2014: 855-855.

[4] M. Vossiek, L. Wiebking, P. Gulden, J. Wieghardt, C. Hoffmann, and P.Heide, “Wireless local positioning,” IEEE Microw. Mag., vol. 4, no. 4, pp. 77–86, Dec. 2003。

[5] N. Arrue, M. Losada, L. Zamora-Cadenas, A. Jimenez-Irastorza, and I. ´Velez, “Design of an IR-UWB Indoor localization system based on a novel ´ RTT ranging estimator,” in Proc. 1st Int. Conf. Sens. Device Technol. Appl., Jul. 2010 pp. 52–57.

[6] A. Yazici, U. Yayan, and H. Yucel, “An ultrasonic based indoor positioning system,” in Proc. Int. Symp. Innov. Intell. Syst. Appl., Jun. 2011 pp. 585–589.

[7] 视距传播中地面起伏对传播中值衰减的影响. 电讯技术. 2003 年 5 期：103-106 页.

[8] Pimputkar S, Speck J S, Denbaars S P, et al. Prospects for LED lighting[J]. Nature Photonics, 2003, 3(4):180-182.

[9] PHILIPS. A Long Lifespan I LED. http://www.lumec.com/newsletter/architect_06-08/led.htm.

[10] 人文网. 占空比. http://www.renwen.com/wiki/%E5%8D%A0%E7%A9%BA%E6%AF%94

[11] Moreland, Kenneth, Angel, et al. The FFT on a GPU[J]. Acm Siggraph Graphics Hardware, 2003:112-119.

[12] Walden R H. Analog-to-digital converter survey and analysis[J]. Selected Areas in Communications IEEE Journal on, 1999, 17(4):539-550.

[13] Surhone L M, Tennoe M T, Henssonow S F. Smoothing[M]. Betascript Publishing, 2010.

[14] 邓雪. 简单平均法预测误差平方和的进一步研究[J]. 数学的实践与认识, 2008, 38(12):60-65.

[15] Axelsson O. A generalized conjugate gradient, least square method[J]. Numerische Mathematik, 1987, 51(2):209-227.

[16] Shabtai A, Fledel Y, Kanonov U, et al. Google Android: A Comprehensive Security Assessment[J]. IEEE Security & Privacy, 2010, 8(2):35-44.

[17] CSDN 博客. Android 的优点与不足. http://blog.csdn.net/jackycoder/article/details/52586094

[18] Oliphant T E. Python for Scientific Computing[M]. IEEE Educational Activities Department, 2007.

[19] Surhone L M, Tennoe M T, Henssonow S F. Photoresistor[M]. 2013.

[20] 高茜. 用电桥研究光敏电阻的光电特性[J]. 物理实验, 2004, 24(9):13-15.

[21] 开源中国. Android 操作系统 11 种传感器介绍. https://www.oschina.net/question/163910_

28354

[22] SciPy.optimize.leastsq 介绍. https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.leastsq.html

[23] Baldi P. Gradient descent learning algorithm overview: a general dynamical systems perspective.[J]. IEEE Trans Neural Netw, 1995, 6(1):182-195.

[24] Google Android 开发官方文档. 2016. http://developer.android.com

[25] SciPy.optimize.fsolve 介绍. https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.fsolve.html

[26] Zhang J Z, Xu C X. Trust region dogleg path algorithms for unconstrained minimization[J]. Annals of Operations Research, 1999, 87:407-418.

[27] Li Y. Centering, Trust Region, Reflective Techniques for Nonlinear Minimization Subject to Bounds[J]. Cornell University, 1993.

[28] Moré J J. The Levenberg-Marquardt algorithm: Implementation and theory[J]. Lecture Notes in Mathematics, 1978, 630:105-116.

致谢

首先，我要感谢我的毕业设计指导老师——赵方老师。赵方老师是我的学业导师，在大学的四年中对我谆谆教诲，并给予了我许多帮助。在毕设期间，即使平时工作十分繁忙，赵老师仍然会抽出时间认真负责地检查我们的工作进度，为我们提出建议，还会仔细到标点符号地审读我们的论文，帮助我们改进。能在本科期间遇到像赵老师这样的好老师，我觉得很幸运。赵老师不但是我们心灵的伙伴，更是我们人生的导师。

其次，我要感谢在项目上给予我很多指导的罗海勇老师，由于对可见光以及通讯领域的陌生，项目对于我来说困难重重。我经常找罗老师探讨问题，不管问题多浅显多简单，罗老师都会十分耐心地为我解答，也会提出很多自己的想法。可以说，如果没有罗老师，我也许根本不可能完成这次毕设。我毕业后申请到美国去读研究生，在没有收到邀请函之际，罗老师还安慰我，告诉我实力才是决定人生高度的最重要因素，不要为了一时的起伏就意志消沉，而要一直不断地提升自己的实力。

同时，我要感谢罗老师实验室的学长学姐们对我的关心和帮助，他们的技术支持让我不必重新研究不熟悉的知识，他们的零食也让我倍感温暖。

我还要感谢我的父母，他们时常关心我的论文的进度。

最后，向所有在毕设期间以及大学期间给予我帮助和关心的老师们、同学们、朋友们致谢，正因为有他们，我的大学才倍加精彩！