基

于

iBeacon

设

备

蓝

牙

定

位

算

法

SDK

开

发

者

文

档

李家驹

2019.4.19

目 录

1. 摘要
2. SDK总览
3. 类之间的关系
4. 数据格式
5. 类详解
6. 精度及创新点

七、需要改进之处

1. 摘要

本文算法基于iBeacon设备，通过RSSI无线电信号衰减方程构建测距模型，使用高斯滤波、卡尔曼滤波等多种信号处理手段，采用六点质心、三点质心和最优策略三交点质心三种定位算法融合（也可单独使用某一算法进行定位），筛选扫描设备优化方案，进行蓝牙室内定位，在实际测试环境中取得了不错的定位精度。

关键词：RSSI、卡尔曼滤波、定位算法融合、蓝牙室内定位

1. SDK总览

本文算法采用java语言编写，Android Studio集成开发环境，minSdkVersion设置为24。

算法一共4个文件夹，13个类，一个外部依赖包，其中MainActivity类作为测试demo的入口，可实现手机自动接受蓝牙设备的广播并得到定位坐标。

iBeacon文件夹主要存放iBeacon设备的信息，在iBeacon文件夹下iBeacon类用于创建iBeacon对象，iBeaconDevice\_ceshi类用于存储用户已安装iBeacon对象。

positioning文件夹是存放整个定位核心算法的文件夹，在定位算法文件夹(positioning)下，Positioning类用于构建整个定位算法的框架，Circle类用于创建圆对象，TwoCircleStrategy类用于实现计算两圆之间相互关系的策略点，CirIntersect类用于计算两圆交点，Combination类用于得到集合中选出三个设备的排序可能，ThreeCircleCentroid类用于得到这三个设备通过两圆策略计算的最终质心结果。

utils文件夹用来存放所有工具类的，在utils文件夹下，GaussFilter类实现高斯滤波，Kalman类实现对RSSI值和定位结果坐标的滤波，utils类实现用处较多的静态工具如求均值等，WirelessPropagation类用于构建测距模型。

libs文件下用于存放外部依赖包，本算法共使用一个开源包，jama-1.0.2是实现java矩阵运算的基础工具包。

三、类之间的关系

MainActivatiy类作为demo入口，读取已安装的iBeaconDevice，然后在扫描得到iBeacon对象后，调用Positioning类进行定位得到结果。iBeacon类在安装完成设备后通过调用无线电测距模型类，在距离设备1米处测试，得到的结果进行高斯滤波得到txPower和环境因子n。Positioning类通过获取扫描到的iBeacon对象，获得多个RSSI值，对这些RSSI进行卡尔曼滤波，求均值，然后排除距离过远的设备，得到进行定位所需要的iBeacon对象列表，将这些对象进行三个一组排列组合，每三个调用ThreeCircleCentroid获得算法计算后的质心，再将这些质心进行坐标卡尔曼滤波，得到最终的定位坐标。ThreeCircleCentroid类中主要是选择三种定位算的哪种还是如何做算法融合，调用TowCircleStrategy类中的三种算法得到一系列策略坐标后，调用utils类中的质心计算工具得到质心定位坐标。

四、数据格式

整个算法数据格式统一使用如下：

iBeacon设备MAC地址和UUID使用String类型

单一RSSI值使用double类型

多个RSSI值使用List<Double>类型

单一坐标值使用double[]类型，如果需要初始化数组长度为2

多个坐标值使用List<double[]>类型

五、类详解

1. MainActivity类

安卓APP开发中接收蓝牙广播需要打开权限，应先在AndroidManifest中添加许可。通过重写onLeScan方法得到扫描回调Callback，从Callback中可以得到设备的地址和rssi值，将扫描到的设备和已安装的设备地址做比对，判断接受到的蓝牙设备是不是所需要的，将扫描时间内同意设备的rssi值存到对象的RSSI列表中，并将设备discover属性更改为true。

在一轮扫描结束后将discover为true的设备存储到用于定位的scan\_iBeacon列表中，然后运行runPositioniong进行定位，得到坐标结果。然后清空scan\_iBeacon设备，删除残留的RSSI值，并将设备的discover设置为false。

通过递归调用scanLeDevice实现自动扫描得到蓝牙设备，然后进行连续定位。

1. iBeacon类

iBeacon类用于构建iBeacon对象，每个对象有12个属性，分别是discover用于标记是否被用户发现、txPower一米处的RSSI值（可存到iBeacon设备中），para\_n用于设置环境因子参数，通过安装后实地测出一米处RSSI强度和精确的距离，通过setPara进行设置txPower和para\_n的值（可使用高斯滤波或者卡尔曼滤波对多个测量值进行滤波，然后在进行计算）、RSSI\_Value用于存放扫描得到的多个rssi值、rssi用于存放单次扫描得到的rssi、name用于存放iBeacon的名称、proximityUuid值用于存放iBeacon类型唯一标识码、bluetoothAddress用于存放iBeacon设备的MAC地址、coord用于存放安装后的设备坐标、major和minor用于存放用户自定义参数，用于扩展功能、distance用于存放用户到该iBeacon对象的距离，通过WirelessPropagation进行设置。

1. iBeaconDevice\_ceshi类

iBeaconDevice\_ceshi类用于存放安装的设备信息，后期需要建立数据库来取代该类。

1. Circle类

创建圆对象，设置圆心和半径。

1. CirIntersect类

用于计算两圆交点的坐标，需要改进的是两圆不相交和无解的情况，可以使用三点质心法里的思想，将比例算出来，计算出相交时的点位坐标，然后将坐标按比例平移，就可以计算出不相交的坐标。

1. Combination类

用于将多个iBeacon对象按照每3个设备为一组的原则进行排列组合。

1. Positioning类

用于将整个定位算法串联起来，构建整个算法的框架，在类中首先创建了两个卡尔曼滤波器，一个用于rssi值的滤波，一个用于坐标滤波，已经将滤波器初始化过了。通过执行静态方法runPositioning进行定位算法传入的值为LIst<iBeacon>列表。首先筛选出可用的iBeacon存入到新的列表中，同时可以进行rssi卡尔曼滤波平滑数据，滤波后的rssi值通过setDistance计算出用户到iBeacon设备的距离作为圆心，并将距离大于10米的iBeacon设备滤去。

得到要用作定位的iBeacon设备后，通过排列组合，每3个设备进行融合算法三圆质心计算(ThreeCircLeCentroid)，将所有的质心计算结果进行坐标卡尔曼滤波得到最终坐标。

1. ThreeCircleCentroid类

该类得到三圆质心，通过三种算法融合，或通过某一单一算法需要在该类中进行修改。

1. TwoCircleStrategy类

该类用于通过算法计算两圆策略坐标，共三种算法，分别六点法，三点法和交点法。

六点法如图1所示，选取两圆圆心连线交与两圆的两交点，三个圆两两相交得到六个策略坐标，然后计算这六个坐标的质心得到最终的定位结果。

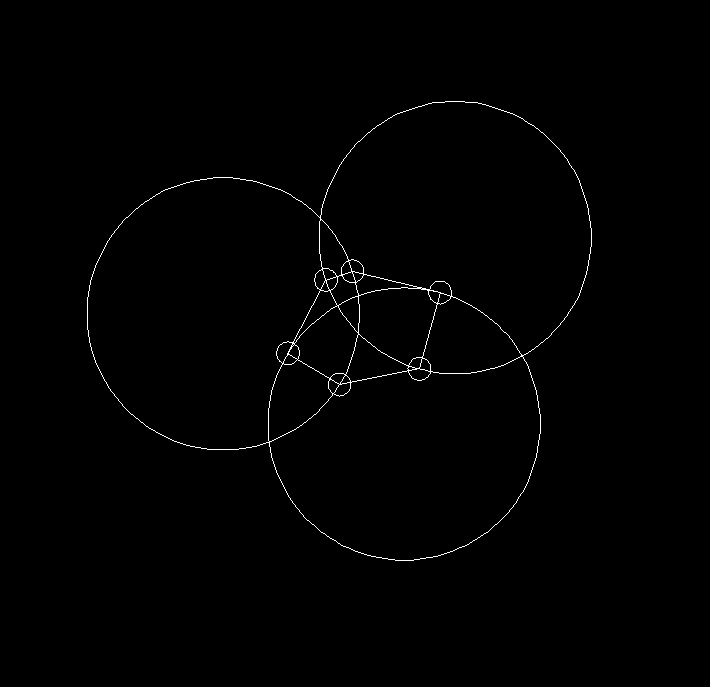


图1 六点质心定位法

三点法如图2所示，选取两圆交点连线与两圆圆心连线的交点，三个圆一共得到三个策略坐标，然后计算这三个坐标的质心得到最终的定位结果。

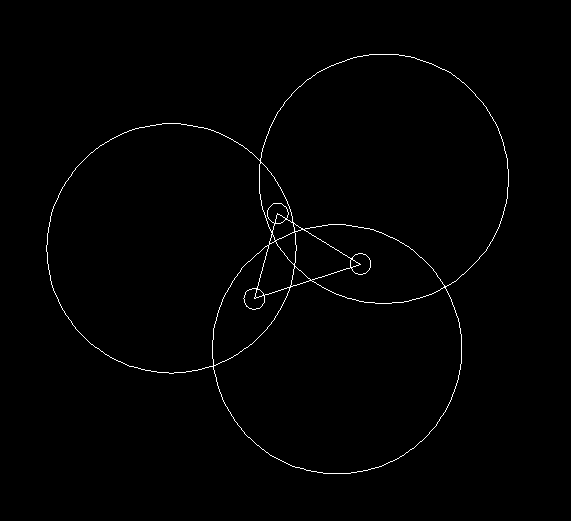


图2 三点质心定位法

以上两种方法都是通过比例关系得到的策略坐标，不需要考虑两圆相离的情况。最近三交点定位法采用求交点的方法进行计算，需要将想离的情况也实现，尚未完成。还有一种构思是使用两圆交点连线，三条线交点进行定位，尚未实现测试。

最近三交点质心定位法如图3所示，通过选取三个圆两两相交中最近的三个交点，求得质心，得到结果。

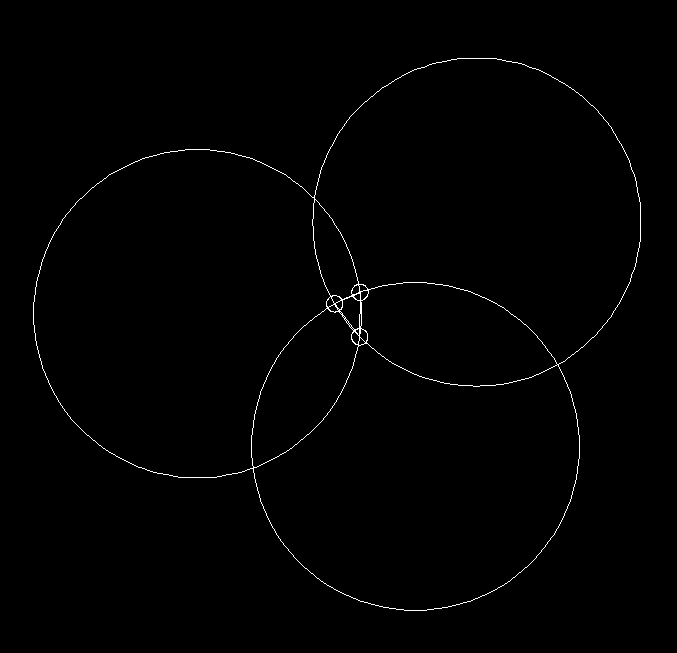


图3 最近三交点质心定位法

1. GaussFilter类

自然噪声总是符合高斯分布，符合高斯分布的噪声可以通过高斯滤波对其值进行平滑处理，左右阈值设置为1是标准差规则滤波，左右阈值设置为0.85是常规的高斯滤波。runGF方法传入一个List<Double> 输出一个高斯滤波后的List<Double>，所以可以对一组值先进行MSE滤波，在进行常规滤波，得到最终滤波结果。

1. Kalman类

Kalman类需要对坐标进行操作，所以需要引入矩阵运算包，开源包Jama提供对Matrix的相关操作，包括创建矩阵、矩阵四则运算、转置和求逆等等常规操作。

卡尔曼滤波是对高斯白噪声的一种线性滤波，是一种最优化自回归数据处理算法。原理是根据系统的实际测量值和预估值对状态向量进行重新构造，并以预估-校正的模型思想进行递归，通过系统的实际测量值和预估值来消除随机噪声，重新构造系统状态。

卡尔曼滤波两个时间方程，三个状态更新方程对下一刻的结果进行预测，通过计算出卡尔曼增益，即预估值和测量值之间的比重，来得到一个新的值作为下一刻的值。

如果不懂卡尔曼滤波可以找相关的论文、视频和例子多看看理解了五个方程就掌握了线性卡尔曼滤波。

1. utils类

工具类，分别是：

AverageValue方法，得到一个列表中数据的均值，或者得到一个数组列表中某一列的均值存到一个新的double[]中。

getCentroid方法，得到一系列点坐标，计算这些点组成多边形的质心坐标。

Equal\_iBeaconDevice方法，在List<iBeacon>中查找是否有相同的iBeacon设备。

find\_iBeacon方法，将找到的iBeacon设备选中，返回该iBeacon对象。

ToTwoDecimals方法，将结果保留两位小数。

getResultCoord方法，得到六个点中距离最近的三个点的坐标组成的三角形的质心坐标。

1. WrielessPropagation类

无线电测距模型算法，通过rssi值反算距离。

六、精度及创新点

算法精度较高，在网型条件良好的情况下可以双轴都达到1m以内的精度。

创新点，采用高斯滤波和卡尔曼滤波融合平滑RSSI数据，将接收到的多个设备排列组合成3个一组进行处理，可以得到更多的多余观测量，定位方面采用三种算法相融合的方式，提高了算法的准确性，可用作工业生产使用。

1. 还需改进之处
2. 通过安卓的方式实现，蓝牙扫描不稳定，APP开发中的安卓自带的蓝牙扫描线程容易拥堵，造成不好的体验，应该使用一些开源别人优化过的iBeacon扫描工具提高扫描的稳定性，或者研究多线程的方式提高稳定性。
3. 蓝牙设备本身具有不稳定性，可以结合多传感器，如陀螺仪三轴加速度定位、WiFi指纹法等方式得到定位数据，进行数据融合，提高整个系统的稳定性。
4. 多余观测没有充分利用，可以进行平差进一步提高算法的准确性，没有测试极限RSSI稳定距离，无法在安装设备时将利用率达到最大化。