Содержание

|  |  |  |  |
| --- | --- | --- | --- |
| Введение | | | 3 |
| 1 | Постановка задачи | | 5 |
| 2 | Глоссарий | | 6 |
| 3 | Анализ задачи | | 8 |
|  | 3.1 | Анализ примеров успешного внедрения . . . . . . . . . . . . . . | 11 |
|  | 3.2 | Анализ существующих платформ . . . . . . . . . . . . . . . . . | 13 |

1. Выбор типа маячков . . . . . . . . . . . . . . . . . . . . . . . . . 14
2. Физическая реализация маячка . . . . . . . . . . . . . . . . . . . 15

3.5 Анализ формата входных данных . . . . . . . . . . . . . . . . . 16

1. Анализ альтернативных технологий для навигации внутри по-мещений . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 18
2. Анализ безопасности . . . . . . . . . . . . . . . . . . . . . . . . . 21
3. Выбор языка и технологий программирования . . . . . . . . . . 26

|  |  |  |
| --- | --- | --- |
| 3.9 | Анализ предоставляемого API . . . . . . . . . . . . . . . . . . . | 27 |
| 3.10 | Анализ лиейного фильтра Калмана . . . . . . . . . . . . . . . . | 27 |

1. Анализ оптимального расположения маячков в помещении . . . 29
2. Анализ задачи выбора оптимального набора маячков . . . . . . 30

|  |  |  |
| --- | --- | --- |
| 4 Реализация | | 34 |
| 4.1 | Реализация алгоритма, основанного на сравнении локационных |  |
|  | отпечатков . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 34 |
| 4.2 | Реализация алгоритма локализации по Монте-Карло . . . . . . | 36 |

1. Реализация алгоритма трилатерации, основанного на опреде-лении силового центра . . . . . . . . . . . . . . . . . . . . . . . . 38
2. Реализация алгоритма трилатерации, основанном на пересече-

|  |  |  |
| --- | --- | --- |
| нии сфер . . . . . . . . . . . . . . . . . . | . . . . . . . . . . . . . . | 40 |
| 4.5 Адаптивный геометрический алгоритм | . . . . . . . . . . . . . . | 41 |

1. Основные классы библиотеки . . . . . . . . . . . . . . . . . . . . 43
   1. BeaconLocation . . . . . . . . . . . . . . . . . . . . . . . . 43
   2. Floor . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 44

1

* 1. Processor . . . . . . . . . . . . . . . . . . . . . . . . . . . . 44
  2. Attractor и Spot . . . . . . . . . . . . . . . . . . . . . . . . 45

1. Структура библиотеки . . . . . . . . . . . . . . . . . . . . . . . . 45
2. Экспериментальные результаты применения синергетического метода . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 46

|  |  |
| --- | --- |
| Заключение | 49 |
| Список литературы | 50 |
| Приложения | 55 |

2

Введение

В последнее время всё более актуальной становится проблема навигации внутри помещений, а также предоставления посетителям услуг, основанных на их местоположении и предпочтениях. Здания становятся всё более объ-ёмными и нередко имеют довольно сложную структуру, ориентироваться в которой могут лишь те, кто постоянно посещает такие здания, а для неподго-товленного человека ориентирование в таких местах превращается в пытку.

Благодаря indoor-навигации (навигации внутри помещений) появляются новые инструменты для маркетинга: проходя мимо магазина, человек может моментально узнать о проводимых в нем акциях, мероприятиях, товарах и предоставляемых услугах, благодаря всплывающему сообщению на экране своего телефона (так называемом “Geo-fencing”, причём предложенные ему предложения будут учитывать его интересы, так как можно учитывать ин-формацию о его прошлых покупках), либо просто получить уведомление при приближении к определенному месту (второе направление indoor-навигации, называемое ¾Geo-aware¿), а владельцы – получать статистическую информа-цию (¾тепловые карты¿ посетителей – своеобразный и очень мощный oﬄine-аналог Google Analytics), основанные на перемещениях клиентов внутри тор-говых залов (таким образом понять, какие отделы и товары пользуются по-вышенным интересом, очень легко). Рынок подобной геоконтекстной рекла-мы уже измеряется миллиардами долларов, и с развитием систем indoor-навигации ожидается его стремительный рост.

Кроме того, решения, применяемые в indoor-навигации, помогают и в ори-ентировании вне зданий, на улице – там, где в условиях плотной застройки использование систем спутниковой навигации затруднено. Особенно эта про-блема актуальна для Японии с высокой плотностью городской застройки.

Bluetooth-маяки, представляющие собой один из способов решения про-блемы локации, также являются частью Интернета вещей, и способны изме-нить представление о самом Интернете вещей. Google представила концеп-цию "physical web которая призвана объединить два мира: реальный и вир-туальную Интернет-сеть. Смартфон, сканируя ближайшие к нему маячки, получает из их сигнала встроенные URL-ссылки, доступные пользователю.

3

Таким образом, пользователь может мгновенно получить доступ к инфор-мации, определенной в ближайшем локационном контексте. Кроме того, это шаг навстречу унифицированному, бесшовному интерфейсу взаимодействия с цифровым миром.

Но современный этап развития технологии внутренней навигации пред-ставлен в основном технологиями для навигации роботов по маякам. Кроме того, существует множество готовых продуктов и платформ от производите-лей с мировым именем. В то же время сообщество программистов не распо-лагает подходящими материалами, чтобы оценить существующие варианты алгоритмов трилатерации и выбрать лучший способ в рамках собственных приложений или свободно распространяемыми библиотеками, которые мож-но было бы легко подстроить в рамках решаемой задачи.

Именно это и подтолкнуло меня к решению задач, поставленных в рамках настоящей работы.

4

* Постановка задачи

Цель работы:

1. Проанализировать и сравнить различные методы трилатерации, исполь-зуемые при решении задачии навигации внутри помещений:
   * 1. метод, основанный на поиске области пересечения сфер;
     2. метод, основанный на поиске силового центра;
     3. продвинутый геометрический алгоритм;
     4. метод, основанный на фильтрации частиц;
     5. метод, основанный на сравнении отпечатков измерений.
2. Разработать библиотеку, содержащую реализации наиболее перспектив-ных и гибких алгоритмов из перечисленных выше, выбранных на основе предварительного анализа.
   * разрабатываемому проекту выдвинуты следующие требования:
3. Поддержка различных методов трилатерации;
4. Работа алгоритмов должна быть максимально оптимизирована как по скорости вычислений, так и по объему занимаемой памяти;
5. Библиотека должна легко подключаться в проект при помощи мене-джеров управления зависимостей CocoaPods или Carthage;
6. Первоначальная настройка библиотеки в проекте должна содержать минимум кода и быть интуитивно понятной.

5

* Глоссарий

LBS (“Location Based Service”) – сервис, основанный на местоположении. GPS (“Global Positioning System”) – глобальная система позиционирова-

ния.

LBA (“Location Based Advertising”) – геоконтекстная реклама.

RFID (“Radio Frequency Identification”) – радиочастотная идентификация. NFC (“Near Field Communication”) – ближняя бесконтактная связь.

BLE (“Bluetooth Low Energy”) – беспроводная технология Bluetooth с низ-ким энергопотреблением.

SDK (“Standard Developer Kit”) – стандартный набор разработчика.

API (“Application Programming Interface”) – интерфейс программирования приложений.

CMS (“Content Management System”) – система управления содержимым. REST (“Representational State Transfer”) – передача репрезентативного со-

стояния.

BVH (“Bounding Volumes Hierarchy”) – иерархия ограничивающих объе-мов.

DoS (“Denial of Service”) – отказ в обслуживании.

AGA (“Adaptive Geometric Algorithm”) – адаптивный геометрический ал-горитм.

Спуффинг маячков (“beacon spoofing”) – тип хакерской атаки, заключаю-щийся в выставлении поддельного маячка, настроенного с параметрами ма-яков некоторой существующей группы маяков.

Пиггибэкинг маячков (“beacon piggybacking”) – тип хакерской атаки, осно-ванный на использовании параметров маячков, определенных в одном при-ложении, в стороннем приложении, принадлежащем третьим лицам. Сам же термин “piggybacking” с английского можно перевести как ¾несанкциониро-ванное проникновение вслед за зарегистрированным пользователем¿.

Fingerprints (с англ. буквально ¾отпечатки пальцев¿) – способ нахожде-ния положения пользователя на основе сравнения ряда измерений расстоя-ний до маяков с набором эталонных измерений, определенных для известных локаций.

6

Фильтр частиц (“particle filter”) – метод трилатерации, в котором в процес-се генерации частицы (точки в двумерном пространстве) все лучше прибли-жают истиное положение пользователя, а наименее значимые исключаются из рассмотрения.

7

* Анализ задачи
  + - случае спутниковой навигации (GPS/Глонасс) существуют сервисы Out-Door, благодаря которым пользователь может узнать о ближайших кафе, ресторанах, гостиницах и других местах благодаря тому, что известно его текущее местоположение. А благодаря сервисам indoor-навигации возможно без проблем и оперативно найти ближайшую стойку регистрации в здании аэропорта, экспонат в музее, отдел и полку с нужным вам товаром в магазине, свободное место на парковке, и многое другое.
    - качестве наиболее значимых примеров можно перечислить следующие:

Розничная торговля. Установив маячки, становится возможным при-ветствовать клиента, сообщать ему о новых поступлениях и предложе-ниях, когда тот находиться рядом, а заодно узнать, как он движется внутри заведения и у каких товаров проводит больше времени, - вер-ный ключ к оптимизации для маркетологов.

Городской туризм. Оказавшись рядом с объектом достопримечатель-ности, пользователь получает информацию о месте. Этот же механизм поможет туристам получать уведомления о прибытии интересующего их транспорта к указанному месту.

Управление кадрами. Используя механизм маячков, становится легко следить за тем, сколько времени тратит в офисе каждый из сотрудни-ков, если установить пару датчиков на вход. Также возможно быстро организовывать собрания: достаточно разослать оповещение каждому, кто не находится на рабочем месте.

Фитнес-услуги. Новички могут получать информацию о предназначе-нии того или иного тренажера, а также инструкцию по его правильному использованию. Кроме того, для любого посетителя может оказаться полезной индивидуальная программа тренировок, которая бы не толь-ко замеряла время на подход и перерывы в занятиях, но и указала бы, где найти тренажер, наиболее эффективный для следующего этапа тре-нировок.

8

Ресторанный бизнес. В данной сфере оригинальной идеей будет выде-ление уникальных посетителей заведения: это не только информирует владельца ресторана, но и позволит разработать систему персональных скидок и условий. К примеру, клиент-завсегдатай может сформировать свой заказ заранее, и, как только он войдет в заведение, заказ автома-тически поступит на кухню.

Дом. Bluetooth-маячки способны расширить функциональность умно-го дома или заменить многие сенсоры. В способы применения входят, например, умное освещение или блокирование входной двери. В данной сфере развивается, например, команда "airfy Beacon"[1].

Гостиницы. Маячок на ресепшне улавливает гостей и сразу позволя-ет отобразить информацию о постоянных посетителях с их историей пребывания, что в итоге уменьшает время заселения. Такой системой пользуется сервис Mahana [2].

Концертные площадки и стадионы. Поклонники музыкантов и спортс-менов получают подробную справочную информацию и статистику. Же-лающие могут приобрести билеты на будущие события. С нового сезо-на так работает приложение Главной лиги бейсбола США, на основных стадионах которой поставили маячки [3].

Дейтинг. Маячки в барах улавливают людей с поддерживаемыми при-ложениями. Если посетители хотят с кем-нибудь познакомиться, то по-лучат уведомление, через которое смогут увидеть профили тех, кто находится поблизости. Функцией уже вовсю пользуются конкуренты Tinder например, Mingleton [4].

Школы и университеты. Преподавателям не нужно отмечать, кто из школьников и студентов ходит на занятия: система сама учитывает всех пришедших и, если необходимо, оповещает родителей. Кроме того, спе-циальное приложение может предлагать учащимся вакансии, делить аудитории, высылая в разные части разный контент. Также в прило-жении может быть встроенный мессенджер и специальная кнопка для

9

застенчивых студентов, которые стесняются задать вопрос на занятии. Уже есть приложения, которые справляются с некоторыми этими зада-чами, например, BeHere.

Фестивали и конференции. Посетителям высылают карту и расписа-ние, справочную информацию о тех, кто выступает, и предлагают по-дойти к стендам. Крупнейшие площадки уже опробовали технологию: на SXSW с помощью iBeacon устраивали опросы, обсуждения среди участников и помогали разобраться, где что находится, а на кинофести-вале ¾Трайбека¿ зрители могли сориентироваться, когда и где покажут нужный им фильм. Необычнее же всего технологию использовали на Каннском кинофестивале. Официальное приложение показывало, кто из гостей где находится, а также отображало ссылку на профиль каж-дого в LinkedIn так было удобнее налаживать бизнес-контакты.

Благодаря большим коммерческим перспективам, направление indoor-на-вигации становится всё более востребованным и уже привлекло внимание таких крупных игроков на рынке, как Google, Apple, Qualcomm, Broadcom, Sony и другие, и в это, без сомнения, перспективное направление уже инве-стируются сотни миллионов долларов [5].

Всемирно известная компания PayPal обозначила свой курс развития, как ¾инвестиции в мобильные технологии, а также возрождение розничной тор-говли с помощью, опять же, технологий¿ [6].

Разрабатываемое ими устройство – PayPal Beacon – как раз реализован-ный шаг в данном направлении. Оно представляет собой портативный считы-ватель дебетовый и кредитных карт, по функциональности не отличающийся от тех, что зачастую установлены на кассах. С помощью него, кроме непо-средственно оплаты, возможно и локационное взаимодействие: местоположе-ние пользователя является связующим звеном, помогающим определить, что он находится в конкретном заведении, и мгновенно предложить авторизацию для совершения дальнейших покупок.

10

1. Анализ примеров успешного внедрения

Международная платежная система MasterCard, Мультимедиа Арт Му-зей Москва, агентства Digitalizm и Insight ONE объявляют о запуске интер-активного гида ¾MasterCard Бесценные города – Твой МАММ¿ - первого в России интерактивного гида по музею, разработанного с применением техно-логии iBeacon. Таким образом проект решает современными инновационными средствами образовательные и маркетинговые задачи в пространстве музея [7].

Крупная сеть аптек Rite Aid установила маячки в более чем 4500 торго-вых точках, воспользовавшись услугами компании inMarket. inMarket знаме-нит услугами по изготовлению и установке Bluetooth-маячков собственного, защищенного от атак хакеров, формата [8].

В другом материале InMarket описывает кейс американского производи-теля продуктов питания Hillshire Brands, который хотел повысить продажи определенного сорта выпускаемых им сосисок. Разработав и внедрив прило-жение, работающее на основе iBeacon, компания получила увеличение узна-ваемости марки сосисок на 36% и рост общего объема продаж в 10 американ-ских магазинах, где были установлены маячки iBeacon [9].

Zatarain - первая в мире компания по производству фасованных потреби-тельских товаров, которая внедрила технологию iBeacon. В своих магазинах компания в определенных контрольных точках установила маячки iBeacon. Оказываясь рядом с такими точками, посетители получали push-уведомления, которые мотивировали покупателей искать продукты Zatarain в магазине. Люди сканировали продукты бренда и получали за это призовые очки. Поку-патели взаимодействовали с брендированнной мобильной страницей, держа в руках сам продукт от Zatarain. В итоге, люди, которые получали iBeacon-уведомления от компании, в 5 раз чаще использовали ее приложение, что соответственно привело к большому вовлечению потребителей в магазине. Оказалось, что посетители магазина, которые взаимодействовали с продук-том на контрольных точках, в 130 раз охотнее покупали его, чем остальные. Для сравнения - обычный мобильный баннер увеличивает намерение делать покупки в среднем на 3% [10].

11

Известный продавец обуви Timberland тестировал технологию iBeacon от Swirl в двух своих магазинах в Нью-Йорке и Бостоне. Результаты получились следующими: пользователи мобильного приложения просматривали 72% ре-кламных предложений от общего количества приходящих на их смартфоны, а 35% из них купили рекламируемый товар. Около 750 покупателей получили 20% скидку от Timberland [11].

“Virgin Atlantic” и аэропорт ¾Хитроу¿ тестируют сервис информирования авиапассажиров на основе маяков [12]. В Московском аэропорту ¾Шереме-тьево¿ данная система уже частично в ходу, инженерам потребуется около 7500 устройств, чтобы покрыть площадь в 500 тысяч квадратных метров. Следом за ¾Шереметьево¿ к использованию iBeacon-навигации должны под-ключиться аэропорты ¾Домодедово¿ и ¾Внуково¿ [13].

Свидетельством этого является следующий факт: один из крупнейших мировых магазинов "Macy’s"планируют установить в залах своих торговых центров около 4000 таких устройств [14].

В прошлом году на Российском Интернет Форуме ¾РИФ+КИБ¿ в здании было установлено 200 маячков, и всем посетителям была наглядно продемон-стрирована система навигации внутри помещения [15].

Выводы о внедрении: Компания - производитель маячков Swirl провела исследование, результаты которого показали, что пользователи мобильных приложений, в первую очередь (80%), желают получать push-уведомления о скидках. По данным того же исследования люди специально будут отключать возможность получения push-уведомлений, если в них не будет полезной ин-формации (то есть тех же акций и скидок) или сообщения будут не релеванты их интересам или месторасположению.

Подтверждают эти выводы и исследования ювелирного магазина Alex&Ani, внедрившего в свою работу iBeacon технологии. Вот его результаты:

93% опрашиваемых покупателей согласилось с тем, что мобильные ре-кламные предложения, которые они получают в магазине увеличивают вероятность их покупки.

90% утверждают, что мобильные предложения заставляют их чаще по-сещать магазин.

12

88% сказали, что опыт пребывания в магазине становится более интуи-тивным и легким.

Cогласно исследованию Google, 84% посетителей пользуются мобильными устройствами, находясь в магазинах, причём 50% проводят в них не менее 15 минут [16].

По данным аналитики, сегодня во время визита в магазин в 2 раза больше покупателей обращается за информацией о товарах и скидках к мобильному приложению, нежели к продавцу-консультанту. Мобильные решения сегодня полностью меняют картину маркетинговой активности потребителей [17].

1. Анализ существующих платформ
   * + рамках рассматриваемой работы были проанализированы главные су-ществующие платформы, способные без программирования строить iBeacon-приложения, а также выделены их главные особенности.

Indoors (http://indoo.rs/). Компания представляет целую платформу для навигации в помещениях. В перечень инструментов входят инстру-менты для моделирования пространства, расстановки маяков и их ка-либровки. С результами работы можно ознакомиться в демонстрацион-ном приложении.

LabWerk (http://labwerk.com). Основной продукт данной корпорации - решение для музеев mApp. Как следствие, существует функциональ-ность для создания статей об объектах выставок и экспонатах, а также социальные элементы вроде создания опросов. Предусмотрена и кар-та заведения, отображающее текущее положение пользователя. Кроме того, система умеет собирать статистику о посещениях.

ShopJoy (http://shopjoy.se). Авторы данной платформы предоставляют возможность тонко настраивать push-уведомления для всех клиентов на основе их персонализации. За основу берется информация, указанная пользователем при регистрации.

13

LocalSocial (https://www.mylocalsocial.com). В целом у продукта мож-но выделить две сильные стороны. Во-первых, это система управления содержимым, через которую легко выделить географические зоны и за-регистрировать в них маячки. Во-вторых, это система лояльности поль-зователей: возможна настройка спецпредложений, промо-акций, начис-ление очков лояльности и выдача промо-кодов.

1. Выбор типа маячков

На рынке маячков уже десятки компаний предлагают собственные про-дукты, обладающие различными преимуществами и недостатками. Основны-ми критериями для сравнения маячков являются:

Базовая технология, то есть непосредственно канал передачи. В ос-новном это BLE, но возможны и другие варианты, в том числе и соче-тание нескольких.

Совместимость. Apple выдвинул собственный стандарт iBeacon, кото-рый не является единым и унифицированным, поэтому производитель вправе создать собственный стандарт передачи.

Параметры маячка: мощность, частота обновления и другие.

Настройка и перенастройка: прошивка маячков, их параметры; как производитель предполагает изменение этих параметров разработчика-ми.

Платформа и сервисы. Сюда можно отнести такие характеристики, как SDK, единое управление, CMS и другие.

Безопасность. Наличие шифрования, защиты от DoS-атак и прочие аспекты.

Среди основных производителей можно выделить следующие:

Estimote. Используют стандарт iBeacon через канал BLE. Особенностью является встроенный сенсор температуры и акселерометр.

14

Gimbal. Помимо iBeacon, используют собственный стандарт передачи. Обязательным требованием является регистрация приложений и маячков на базе Gimbal SDK. Платформа для предприятий удовлетворяет спецификации REST.

StickNFind. Использует стандарт iBeacon, предлагает варианты его рас-ширения. Опционально шифрует пакеты. Возможность получения темпера-туры и состояния батареи. Звуковая и световая сигнализация.

1. Физическая реализация маячка
   * плане физической реализации Beacon-маячки – это обычные Bluetooth
2. LE (Low Energy) устройства, таким образом, их роль может с успехом выполнять любое устройство, оснащённое BLE-чипом - например, cмартфоны на базе Android, а также iPhone, iPad, обычные ноутбуки, Raspberry Pi с usb bluetooth-донглом (электронным ключом) и т.д., на которое установлено специальное приложение, реализующее функции Beacon-маячка [18].

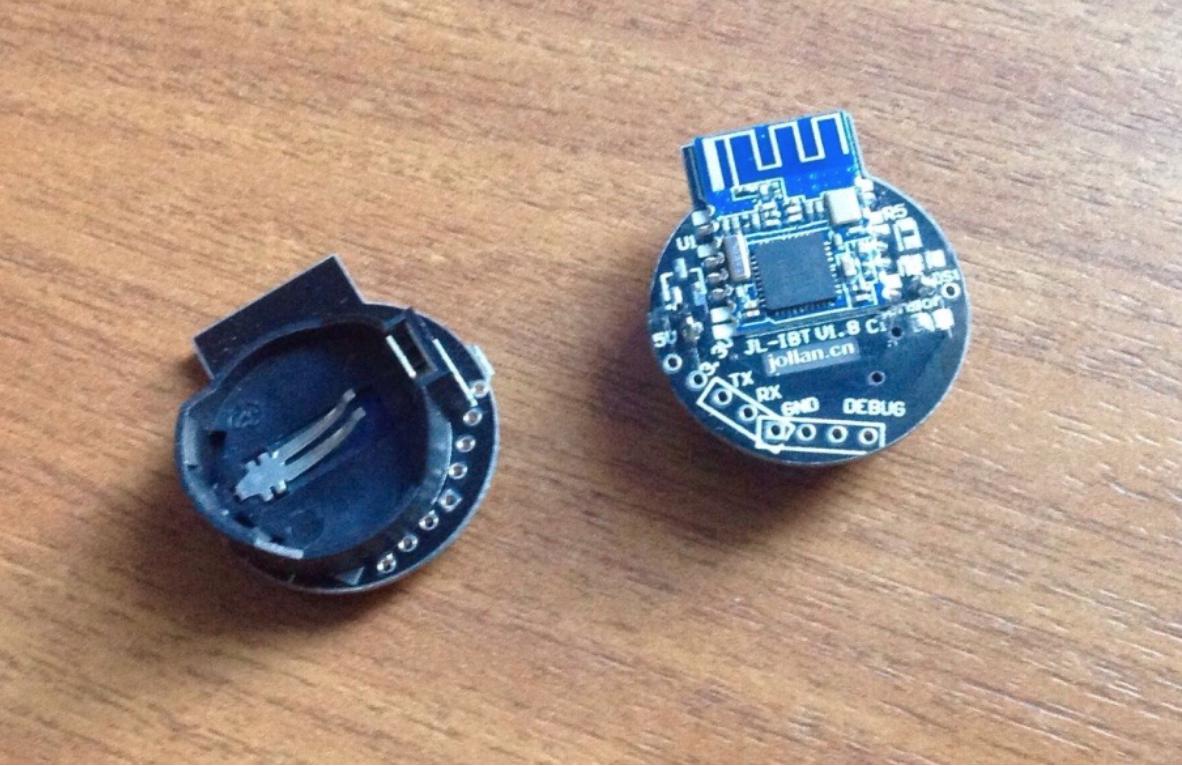


Рис. 1: Маячки, использованные в работе

Типичный маячок, показанный на рисунке выше, имеет довольно ком-пактные размеры, и способен проработать всего лишь от одной батарейки до двух лет. Схемотехнически состоит из батарейки и Soc (System-On-Chip) Texas Instruments CC2540/2541 (также применяется Nordic nRF51822), пред-ставляющий собой 8051 микроконтроллер, в который загружается прошив-ка для реализации функции маячка и периферийный модуль Bluetooth LE. Дальность действия маячка в среднем 10 метров (варьируется от 15-20см до

15

25-40м в зависимости от модели и настроек). Периодичность выдачи данных - 200мс, но это, опять же, настраивается: можно настроить и на более частую периодичность, и на более редкую. В рассматриваемой работе маячки были настроены на периодичность в 1с. Срок службы от одной батарейки в зависи-мости от модели - от чуть менее одного года до трёх лет (в среднем 2 года). Цена одного маячка составляет порядка 15-20 долларов. Маячок является простым устройством, который только выдаёт всем подряд в эфир свои дан-ные (в advertising-режиме), используя Bluetooth-профиль GATT (при этом к нему даже не нужно выполнять подключение), тем не менее, производите-ли, как правило, закладывают возможность подключения к маячку с целью его удалённого конфигурирования (редактирование данных, выдаваемых в эфир, периодичность выдачи данных и мощность излучения), а так же реше-ния вопросов безопасности.

1. Анализ формата входных данных
   * заданной периодичностью, циклически маячок выдает один и тот же набор данных [18, 19]:

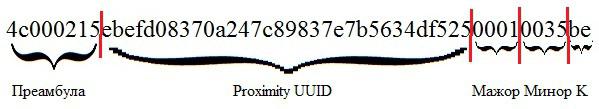


Рис. 2: Пример пакета, передаваемого маячком

Преамбула (4 байта) - префикс пакета, позволяющий установить, что мы имеем дело именно с Beacon-маячком. Всегда равен 4c000215. Преамбула состоит из 4-х полей: идентификатор компании (2 байта, в данном примере - 4c00), тип (1 байт, в примере - 0x02) и длина данных (1 байт, значение -

0x15).

Proximity UUID (16 байт) – Идентификатор группы маяков. Напри-мер, если существуют несколько торговых залов, в которых требуется раз-местить маяки, то во всех этих залах они будут иметь один и тот же UUID,

16

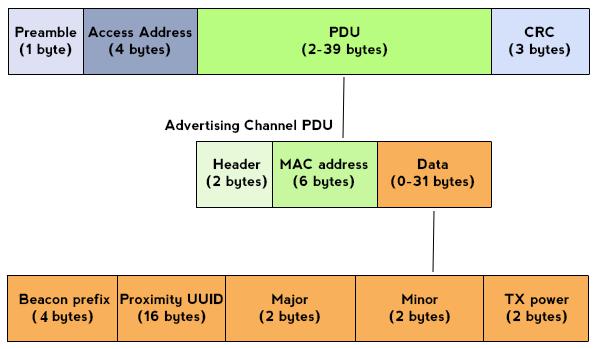


Рис. 3: Общая структура Bluetooth-пакета

указанный при конфигурации, и это позволит отличать маяки от других, посторонних.

Мажор (2 байта) – позволяет различать небольшой набор маяков внут-ри одной группы. То есть внутри одной большой группы маяков, идентифи-цируемой UUID, может быть несколько подгрупп, каждая из которых иден-тифицируется по номеру мажора. Например, согласно приведенному выше примеру, каждому залу можно присвоить свой номер мажора. Если маяка-ми требуется охватить несколько этажей здания - обычно с каждым этажом ассоциируют свой номер мажора.

Минор (2 байта) – номер, идентифицирующий сам маяк внутри мажо-ра. Связка uuid+мажор+минор позволяет нам однозначно идентифициро-вать маяк и по этим данным определить координату самого маячка (обычно используется таблица соответствия маячка и его коорднат).

TX Power (параметр K на рисунке выше, 2 байта) – эталонное значение мощности маячка, представляющее собой силу сигнала на рассто-янии в 1 метр от маячка. Данный параметр, как следствие, является RSSI-параметром с наложенными на него граничными условиями. TX Power из-

17

меряется и записывается в маячок единственный раз при его изготовлении. Данная константа используется при определения расстояния от пользователя до маячка. Первый бит является знаковым (1 - ¾ ¿, 0 - ¾+¿). Например, TX Power в нашем примере (см. рисунок выше, параметр ¾К¿) – 0xBE. Это 190 в десятичной системе счисления. Тогда эталонная сила сигнала на расстоянии 1м от маячка составляет 256 190 = 66dBm.

1. Анализ альтернативных технологий для навигации внутри помещений

RFID – способ автоматичесткой идендификации объектов, в котором посредством радиосигналов считываются или записываются данные, хранящиеся в так называемых транспондерах, или RFID-метках. По своей функциональности, RFID, как методы сбора информации, очень близки к штрих-кодам, наиболее широко применяемым сегодня для маркировки товаров. Следовательно, RFID хорошо подойдет, напри-мер, для отслеживания посылки по большому транспортному узлу, сто-ит лишь снабдить весь маршрут этой посылки и ее саму RFID-метками. То есть RFID применяется там, где система знает про все. Маячки пред-ставляют собой передатчики, и мы не ориентируемся на наличие некой глобальной информации о всем состоянии, есть только информация о локальном состоянии.

NFC - технология беспроводной высокочастотной связи малого радиу-са действия. Она предоставляет возможность обмена данными между устройствами, находящимися на расстоянии около 10 см. Вместе с суще-ствующей инфраструктурой бесконтактных карт, технология нацелена на использование в платежных системах и общественном транспорте. Надо заметить, что в своей сфере она обладает некоторыми важными преимуществами: NFC-метки потребляют мало энергии и легко встра-иваются.

Следует сделать замечание, что сам принцип работы технологий RFID и NFC не совсем подходит для решения круга задач, перечисленных ранее. На практике или их сложно встроить, ведь их требуется много

18

(RFID), или радиус действия слишком мал, как в случае NFC.

Навигация по Wi-Fi. Используется уже существующая инфраструктура сетей связи – точки беспроводных сетей Wi-Fi, и это наименее затрат-ный вариант. Методика определение координат следующая: устройство пользователя сканирует доступные Wi-Fi-точки доступа, затем инфор-мацию о них отправляет на сервер, где эти данные по базе данных со-поставляются с координатами этих точек доступа, по которым и вы-числяются координаты пользователя. К сожалению, координаты Wi-Fi точек точно не известны, плюс могут меняться (перенесли Wi-Fi точку в другое место или заменили её на другую – координаты уже оказыва-ются неверными).

Точность при таком подходе оставляет желать лучшего (погрешность - до 25 метров! При использовании специально созданной Wi-Fi инфра-структуры точность достигает 3-5 метров, но это уже требует ощутимых затрат на создание и обслуживание подобной системы), да и идентици-фировать клиентов по Wi-Fi, привязывая их расположение к карте по-мещений, проблематично: начиная с iOS 8, mac-адреса Apple-устройств (iPhone, iPad) постоянно меняются для предотвращения ¾рекламной¿ слежки.

Геомагнитное позиционирование. Основано на ориентировании по маг-нитному полю Земли и базируется на геомагнитных аномалиях как критериях для геомагнитного позиционирования (аномалии возникают вследствии неоднородности геомагнитного поля). Заключается в фик-сации геомагнитных аномалий и нанесении их на карту территории, на которой предполагается ориентироваться. В дальнейшем навигация производится по составленной карте устройством, в которое встроен магнитометр. Практический пример реализации – система IndoorAtlas команды учёных из финского университета Оулу.

Недостаток – высокая сложность реализации, невысокая точность. В помещениях очень много динамически меняющихся магнитных анома-лий (проводка, поле в которой меняется в зависимости от подключённой нагрузки и сильно меняет конфигурацию магнитного поля вокруг се-

19

бя; посетители со своими радиоэлектронными устройствами; стеллажи;

тележки), сильно усложняющих навигацию, основанную на указанном

способе ориентировании в пространстве.

Системы спутниковой навигации (GPS/Глонасс и другие) + инерци-альные навигационные системы (ИНС). Применимо, когда периоди-чески появляется сигнал систем спутниковой навигации. Его точности по мнению современных исследователей достаточна, чтобы вычислить положение пользователя вплоть до десятка сантиметров [20]! Однако когда пользователь въезжает в тоннель, ещё доступны актуальные ко-ординаты и направление движения с GPS/Глонасс-спутников, далее в самом тоннеле сигнал теряется, и поэтому используется уже инерциаль-ная навигационная система (ИНС, основанная на базе акселерометра, гироскопа, магнитометра), которая использует в качестве начальных условий последние актуальные данные с GPS/Глонасс до потери связи со спутником и поддерживает их актуальность на основе получаемых с датчиков данных о текущей скорости, ускорении и направлении дви-жения до возобновления связи со спутниками.

Стоит принимать во внимание, что в ИНС ошибки постоянно накапли-ваются, и со временем данные, полученные с ИНС, становятся все более и более отличными от действительности.

Ориентирование по базовым станциям операторов сотовой связи (GSM). В зоне видимости сотового телефона/GSM-модема постоянно находятся как минимум одна базовая станция GSM, а обычно их несколь-ко. Координаты расположения этих базовых станций известны (благо-даря многочисленным навигационным сервисам (например ¾Яндекс.На-вигатор¿), приложение получает информацию о видимых мобильным телефоном базовых станций и текущем положении по GSM/¾Глонасс¿, и отправляет эти сведения в ¾Яндекс¿, где на основе этих данных стро-ится база соответствий ¾Базовая станция-координаты¿, к которой име-ется свободный доступ через предоставляемое API). Далее в модем от-правляется команда “AT + CREG = 2”, в результате чего можно по-лучить сообщения +CREG: с информацией о текущей подключенной

20

базовой станции: LAC и CELLID (соответственно код зоны и иденти-фикатор базовой станции). Отправив эти данные на один из специаль-ных сервисов (предоставляемый ¾Яндекс¿, "Google"и другими компа-ниями), возможно определить координаты этой базовой станции. Мно-гие модемы позволяют получить список видимых базовых станций (БС) с указанием их LAC и CELLID - остаётся только через базы данных с координатами БС получить их координаты и методом триангуляции определить примерное местоположение пользователя.

Минусы: невысокая точность (БС может быть удалена на расстоянии в 35км от пользователя, некоторые БС являются мобильными и постоян-но меняют свою дислокацию).

Использование Bluetooth-маячков iBeacon даёт достаточную точность при приемлемом уровне финансовых затрат; перспективная техноло-гия, которая активно развивается, поэтому именно iBeacon был выбран в качестве предмета данной работы.

Навигация, основанная на синергетическом эффекте решает задачу определения текущего местоположения, используя все (или большин-ство) из перечисленных выше способов. Эффективность достигается за счёт того, что используется сразу несколько векторов определения ко-ординат, что способствует компенсации ошибок и повышению точности определения координат. На реализацию подобной системы в прошлом году фондом развития центра разработки и коммерциализации новых технологий ¾Сколково¿ был выделен грант в 1 млн долларов [18].

1. Анализ безопасности

Не стоит переоценивать данную технологию и забывать о таком важном аспекте, как безопасность. На текущий момент развития трафик в процессе передачи никаким образом не шифруется, поэтому злоумышленник потенци-ально может провести атаку.

Выделяются 2 главных типов атак: спуффинг и пиггибэкинг [21].

Beacon Spoofing. Атака основана на включении маячка или устройства,

21

работающего как маячок, с параметрами, идентичными параметрам некоторой группы маячков. Это может быть использовано для того, чтобы симулировать некоторое событие или оповещение в месте, от-личном от предполагаемого в контексте приложения.

Пример угрозы: предположим, есть некоторый магазин, использующий маячки для приветствия новых посетителей. Злоумышленник копирует настройки маячков, и впоследствие это позволяет ему разослать привет-ственное уведомление пользователям, которые находятся в абсолютно другой локации. Это может спровоцировать недоумение и, в конечном итоге, неудовлетворенность программным продуктом.

Beacon Piggybacking, или Hijacking. Атака основана на применении уже существующих маячков, предназначенных для некоторого приложения, в своем, стороннем приложении. Это может быть использовано для по-лучении аналитики, основанной на оригинальный маячках, а также в

рассылке сообщений и симуляции событий, не относящихся к предпо-лагаемым.

Пример угрозы: пусть кофейня ¾А¿ конкурирует с кофейней ¾Б¿. ¾А¿ начинает использовать приложение с использованием маячков. В ответ кофейня ¾Б¿ разрабатывает свое приложение с информацией об иден-тификаторах чужих маячков. В результате каждый раз, как пользова-тель с установленным приложением от кофейни ¾Б¿ заходит в кофейню ¾А¿, на его устройство приходит оповещение о скидках на кофе в ¾Б¿.

Неприятный инцидент произошел на выставке потребительской электро-нике (Consumer Electronics Show – CES) в 2014 году. Для всех желающих была организована ¾охота за сокровищами¿, в рамках которой через приложение пользователи должны были определить место, где якобы зарыт сундук с кла-дом. Однако еще до начала мероприятия неизвестная группа хакеров взяла apk-файл приложения, проанализировала его структуру через декомпилятор, и смогла извлечь параметры всех используемых маячков. Это давало всю необходимую информацию, и победить таким способом можно было даже не выходя из дома [22].

Выделяют 4 способа защиты:

22

Геолокационная проверка. После получения оповещения от одного из новых (в рамках текущей сессии) маячков, устройство использует геолокационный сервис, чтобы убедиться, что маячок физически дей-ствительно находится поблизости.

Идентификация на основе начального значения (seed). Исполь-зуются маячки с периодически меняющимся UUID. Алгоритм смены, в свою очередь, основан на некотором цифровом значении, хранимом отдельно. Через единый SDK происходит обновление, идентификация и синхронизация всего процесса. Маячок, не прошедший проверку – потенциальный вредитель – будет выкинут из рассмотрения.

Облачное подтверждение. Способ базируется на основе предыду-щего, но роль связующего звена на себя берет облачный сервис, а не локальный SDK. Этот механизм уже используется в маячках компа-нии Estimote, и известен под названием “UUID Rotation”. Секретный ключ, определяющий смену идентификаторов, хранится на платформе Estimote Cloud.

Управление на уровне аппаратных средств. Начальные парамет-ры и их возможное обновление берут на себя аппаратные средства – кон-троллеры. При этом оповещение об обновлении приходит от облачного сервиса отдельно на контроллеры и отдельно на устройство пользовате-ля. После этого контроллеры обновляют UUID маяков. В дальнейшем проверка маячков будет происходить на устройстве без использования сервиса.

Преимущества и недостатки перечисленных методов представлены в таб-

лицах 1 и 2, соответственно.

23

|  |
| --- |
| 24 |

Таблица 1: Достоинства различных способов защиты.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Способ защиты | Геолокационная | Идентификация | Облачное | Управление на |
|  | проверка | на основе | подтверждение | уровне |
|  |  | генерации |  | аппаратных |
|  |  | случайных чисел |  | средств |
| Преимущества | 1. Наиболее | 1. Позволяет | 1. Позволяет в | 1. Надежный |
|  | дешевый способ | защититься от | большей степени | уровень защиты |
|  | 2. Простой в | перечисленных | защититься от | 2. Предоставляет |
|  | конфигурации и | типов атак | перечисленнных | устройство для |
|  | поддержке | 2. Нет привязки к | способов атак | обновления и |
|  | 3. Может быть | геолокации или | 2. Не требуется | обслуживания |
|  | легко включен или | Интернет- | подключение | маячков |
|  | выключен в любой | соединению | дополнительных | 3. Изменения легко |
|  | момент | 3. Не требует | устройств | применить к |
|  | 4. Позволяет | дополнительного | 3. Усложняет | существующей |
|  | использовать | оборудования | процедуру | системе |
|  | традиционный |  | проведения атаки | 4. Позволяет |
|  | iBeacon-формат для |  | для | использовать |
|  | лучшей |  | злоумышленника | традиционный |
|  | совместимости |  |  | iBeacon-формат для |
|  |  |  |  | лучшей |
|  |  |  |  | совместимости |

|  |
| --- |
| 25 |

Таблица 2: Недостатки различных способов защиты.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Способ защиты | Геолокационная | Идентификация | Облачное | Управление на |
|  | проверка | на основе | подтверждение | уровне |
|  |  | генерации |  | аппаратных |
|  |  | случайных чисел |  | средств |
| Недостатки | 1. Не защищает | 1. Сложен в | 1. Сложен в | 1. Наиболее дорогой |
|  | против | модификации в | модификации в | тип размещения |
|  | piggybacking-атак | случае неполадок | случае неполадок | 2. Требует наличия |
|  | 2. Геолокационная | 2. UUID маячков | 2. Сложен в | дополнительного |
|  | относительность не | могут быть легко | развертывании | набора устройств |
|  | дает такой же | определены | 3. Задержки в | 3. Не работает для |
|  | точности, как | злоумышленником | использовании | маячков, |
|  | остальные методы | 3. Более сложное | могут отразиться на | расположенных на |
|  | 3. Определение | развертывание | удобстве | удалении от |
|  | локации может | 4. Формат | использовании | остальных |
|  | являться причиной | использования не | приложения | 4. Предполагает |
|  | задержки в начале | поддерживается в | 4. Предполагает | наличие Интернет- |
|  | сессии | приложениях Apple | наличие Интернет- | соединения |
|  |  |  | соединения | (периодически) |
|  |  |  | 5. Формат |  |
|  |  |  | использования не |  |
|  |  |  | поддерживается в |  |
|  |  |  | приложениях Apple |  |

Примечательно, что Apple iOS SDK на уровне реализации не позволяет приложению сканировать эфир на обнаружение BLE-пакетов. Обязательным условием является явная конфигурация UUID, major и minor-идентификато-ров [23].

1. Выбор языка и технологий программирования

Objective-C - объектно-ориентированный язык программирования. Язык во многом основывается на парадигмах, предложенных языком Smalltalk, и построенн на основе языка С. Из Smalltalk-черт можно выделить объектную модель: метод экземпляра не вызывается напрямую, для вызова объекту по-сылается сообщение. Аналогично, класс или объект могут определить, спо-собны ли они вызвать указанный метод с указанными параметрами.

Из-за родства с языком C, компилятор легко понимает как Objective-C, так и C код.

Компилятор Objective-C входит в GCC и доступен на большинстве основ-ных платформ. Язык используется в первую очередь для Mac OS X (Cocoa), GNUstep и iOS (Cocoa Touch).

Swift - мультипарадигменный объектно-ориентированный язык програм-мирования, созданный компанией Apple для разработчиков iOS и OS X. Swift работает с библиотеками Cocoa и Cocoa Touch и совместим с основной кодо-вой базой Apple, написанной на Objective-C. Swift задумывался как более безопасный язык в сравнении с Objective-C. Язык поддерживается в среде программирования Xcode; программы на нем компилируются при помощи Apple LLVM и используют рантайм Objective-C, что делает возможным ис-пользование обоих языков (а также чистого С и С++) в рамках одной про-граммы.

Для реализации библиотеки был выбран язык Objective-C, так как язык хорошо поддерживается стандартной IDE “XCode”, отлично документирован и обладает широким сообществом программистов. Кроме того, в Objective-C лучшая по сравнению со Swift поддержка вставок кода на C, а именно на нем написаны алгоритмы трилатерации, рассматриваемые в данной работе.

26

1. Анализ предоставляемого API

Для indoor-навигации Apple предоставляет библиотеку CoreLocation. Она

содержит несколько основных сущностей, использованных в работе:

CLBeacon. Класс, инкапсулирующий как идентификаторы маяка (uuid, major, minor), так и информацию, используемую при ранжировании (rssi, accuracy, proximity).

CLBeaconRegion. Класс, содержащий идендификаторы группы маяков (uuid, [major, [minor]]). С его помощью можно удобно разделять груп-пы маячков, находящихся, например, на разных этажах, и работать с каждой группой по отдельности.

CLLocationManager. Центральный класс API, который может отслежи-вать маячки в указанном регионе (CLBeaconRegion), рассылая событи-ями как факты входа или выхода в зону действия маячков (monitoring), так и прием пакетов для определения расстояния до источника (ranging).

Таким образом, Apple представила не только формат на BLE-маячки iBeacon, ставший уже широко распространенным, но и удобные средста для работы с ними.

1. Анализ лиейного фильтра Калмана

В рамках исследуемой работы возникает небходимость фильтровать неко-торые данные. Прежде всего, к ним относятся значения RSSI, получаемые от маячков.

Фильтр Калмана был выбран на основании нескольких положений [24]:

Фильтр имеет рекуррентную форму, благодаря чему он удобен для про-граммирования. Кроме того, так как новые оценки формируются на ос-нове старых, нет необходимости хранить весь массив наблюдений, что экономит память и время работы.

Алгоритм фильтрации одновременно представляет собой непосредствен-ное описание способа реализации фильтра.

27

Его легко распространить на нестационарные сигналы; это относится и к случаю, когда наблюдения начинаются в произвольный момент вре-мени.

Легко распространяется на многомерный случай.

* + - тому же, скользящее среднее при большем размере окна (обозачим раз-мер за N) будет отражать не актуальные данные, а лишь усреднение за по-следние N измерений. Калман же при большем доверии новому значению больше приближает показание к последнему измеренному, при этом не ис-ключая корректировки выбросов.

Фильтр Калмана использует динамическую модель системы (например, физический закон движения), известные управляющие воздействия и мно-жество последовательных измерений для формирования оптимальной оцен-ки состояния. Алгоритм состоит из двух повторяющихся фаз: предсказание

* корректировка.

На первом рассчитывается предсказание состояния в следующий момент времени (с учетом неточности их измерения). На втором новая информация с датчика корректирует предсказанное значение (также с учетом неточности и зашумленности этой информации). Важно отметить, что задача фильтрации

– получить наиболее близкое значение к реальной величине.

Рассмотрим алгоритм фильтра в общем случае для следующих имеющих-ся данных:

|  |  |
| --- | --- |
| (zk = xk + k | (2) |
| xk+1 = xk + uk + k | (1) |

где xk – измеряемая величина;

uk – член, отвечающий за контроль системы извне (закон, по которому изме-няется величина x);

k – ошибка модели, то есть разница между реальным значением и рассчи-танным;

k – ошибка, образовавшаяся в результате измерения сенсором; zk – полученные данные с сенсора;

xk+1 – новое значение величины на очередном шаге итерации.

28

Задача состоит в том, что, зная неверные показания сенсора zk, найти хорошее приближение для истинного значения xk. Это хорошее приближение мы будем обозначать как xoptk.

Будем рассуждать по индукции. Представим, что мы уже нашли отфиль-трованное значение с сенсора xoptk. Можно предположить, что на шаге k+1 си-стема эволюционирует согласно закону 1, и сенсор покажет значение, близкое к xoptk + uk. С другой стороны, зная соотношение 2, мы будем знать неточное значение сенсора zk+1.

Идея Калмана состоит в том, что чтобы получить наилучшее прибли-жение к истинной координате xk+1, мы должны выбрать золотую середину между показанием неточного сенсора и xoptk + uk нашим предсказанием то-го, что мы ожидали от него увидеть. Показанию сенсора мы дадим вес K а

на предсказанное значение останется вес (1 K):

xoptk+1 = Kzk+1 + (1 K)(xoptk + uk)

Коэффициент K называют коэффициентом Калмана.

Согласно выводам самого Калмана, коэффициент имеет свойство стаби-лизироваться после небольшого числа итераций [25].

Алгоритм обладает достаточной гибкостью: он позволяет задать априор-ную информацию о характере системы, связи переменных, и на основании этого строить более точную оценку. Кроме того, даже в случае отсутствия априорной информации фильтр дает отличные результаты [26, 27].

1. Анализ оптимального расположения маячков в по-мещении

Мы имеем начальную систему уравнений, основанную на методе трила-

терации:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 8 |  |  | 2 |  | 2 | 2 |  |
|  | (x0 |  | x1)2 + (y0 |  | y1)2 | = r12 |  |
| >(x0 | |  | x2) + (y0 |  | y2) = r2 | |  |
| > |  |  |  |  |  |
| > |  |  |  |  |  |  |  |
| < |  |  | 2 |  | 2 | 2 |  |
| >(x0 x3) + (y0 | | | | y3) | | = r3 |  |
| > |  |  |  |  |  |  |  |
| > |  |  |  |  |  |  |  |

:

Однако данная система не может дать однозначного решения в виде точ-

ки, если, например, общая область пересечения кругов не является точкой.

29

Назовем в таком случае область такого общего пересечения областью ошибки локализации. Введем также обозначение: пусть величина ошибки варьирует-ся в интервале ( ; ). Используя это, определим:

Cpi = f(x; y) 2 R2j(x xi)2 +(y yi)2 (ri + i)2; (x xi)2 +(y yi)2 (ri i)2g

* приведенном выше определении можно заметить, что i = 0; \iCpi об-разует единственную точку. В случае, когда i > 0; \iCpi образует область с площадью, отличной от нуля.
* исследовании [28] математически доказано, что данная площадь бу-дет минимальна (учитывая, что величина мала), если маячки установлены симметрично. То есть в случае трилатерации оптимальным вариантом будет расположение их в узлах равностороннего треугольника, а дальнейшая уста-новка новых устройств для покрытия большей площади будет выполнена, как на рисунке 4.

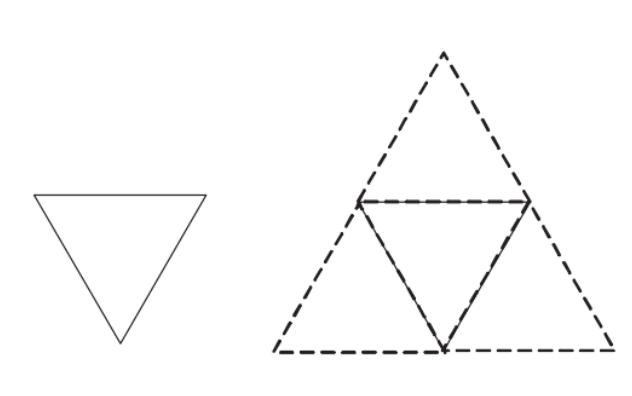


Рис. 4: Оптимальное расположение маячков

Расположение в узлах квадратов дает несколько худший результат (ошиб-ка в среднем больше на 5,3%), а случайное расположение маячков увеличи-вает ошибку на 34,9% [29].

1. Анализ задачи выбора оптимального набора маяч-ков

Существует целое семейство алгоритмов, называемое “OASIS” – Optimum Anchor Selection algorIthmS, что в переводе и значит ¾выбор оптимального набора маячков¿. Для работы с GPS существует функционал, называемый

GDOP – Geometric Dilution Of Precision, то есть ¾геометрическое ослабление

30

точности¿. Данная величина отражает, насколько на текущий момент взятое созвездие спутников вляет на результирующую точность.

Вычислить GDOP возможно по следующей формуле:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| GDOP (M; ) = s |  | M | |  |  | (3) |  |
|  |  |  |  |  |  |
| i | j;j>i | j | sin( ij) | 2 |  |
|  | P P |  | j |  |  |  |

где M – количество спутников, рассматриваемых в рамках нахождения те-кущей локации, а – угол между каждой из пар спутников. При этом чтобы вычислить 3 для отдельно взятой точки, необходимо знать ее положение. В приложениях, использующих GPS, начальное грубое вычисление исходной позиции может быть допущено ввиду больших дистанций между пользовате-лем и спутниками. В рамках задачи определения локации внутри помещений такой подход неприемлем и не может быть использован без каких-либо изме-нений.

Рассмотрим альтернативные методы OASIS:

Совместная кластеризация. Подход метода заключается в выборе k лучших по силе сигнала (RSSI) маячка для дальнейшего вычисления координат.

Простая выпуклая оболочка. В данном методе рассматриваются дистанции, определенные при анализе величины сигнала от каждого из маячков n1; n2; :::; nN . Фунционал выпуклой оболочки, обозначаемый С, может быть вычислен по формуле:

N N

* X

C f jnj : j 0g; j = 1

j=1 j=1

Сила сигнала. Используемая в данной работе техника,заключающаяся в выборе трех маячков с наиболее сильным сигналом вне зависимости от их расположения.

Выбор, основанный на определении площади. Согласно исследо-ванию, проведенному М.Миронововой и Х.Халвисом [30], большая пло-щадь, сформированная тремя спутниками, образует лучшее (то есть

31

меньшее) значение GDOP . Применяя это к поставленной задаче, мы перебираем все возможные треугольники, образованные тройками ма-ячков, выбирая наибольший по площади из них.

Периметр. Метод достаточно схож по своей сути с приведенным выше, но при рассмотрении треугольника учитывается его периметр. Пери-метр также показывает линейную независимость координат образовав-ших его маячков. С другой стороны, использование периметра бесполез-но в рамках некоторых сценариев (например, нахождение трех маячков на одной прямой).

Плотность. Алгоритм, рассматривая все комбинации маячков, вычис-ляет все возможные координаты пользователя. После этого область с наибольшей плотностью возможных координат пользователя выбирает-ся в качестве результата, а конечной точкой обычно берется центроид промежуточных результатов.

Мультилатерация. Если рассматривать сразу N маячков, то возмож-но составить систему уравнений для вычисления положения пользова-

теля, учитывая показания каждого из них.

Положим, что координата узла i есть bi. Необходимо вычислить поло-жение пользователя p. Показания маячков равны di, соответственно. В идеальной ситуации мы имеем:

jjbi pjj22 = d2i

Перепишем левую часть в виде

|  |  |
| --- | --- |
| jjbjj22 + jjpjj22 2biT p = di2 | (4) |

Учитывая, что всего участвуют в расмотрении M маячков, снова воз-можно переписать выражение в следующем виде для вычисления пози-ции пользователя:

|  |  |
| --- | --- |
| p = 0:5B+(a + ) | (5) |

при этом мы воспользовались заменой = jjpjj22 и [a]i = jjbjj22 d2i, а также вектор-столбец B = [p1; p2; :::; pM ]T . Нотация B+ означает псев-дообращение Мура-Пенроуза. Чтобы решить 5, необходимо вычислить

32

:

= pT p = [0:5B+(a + )]T [0:5B+(a + )]

* + нашем случае мы выбираем , которая бы давала наименьшую сред-нюю ошибку между каждым из значений p1; p2; :::; pM и искомой пози-цией (см. 4).
* работе [31], показано, что в рамках проводимого ряда экспериментов, наименьшая средняя ошибка достигается при использовании методов, осно-ванных на выборе по силе сигнала, совместной кластеризации и мультилате-рации.

Важно отметить, что те методы, которые не использовали RSSI-показатель сигнала явно (как например основанные на площади или периметре), давали худшие результаты.

Наконец, при увеличении вовлеченных в процесс вычисления маячков, разные подходы адаптировались лучше или хуже. Так, мультилатерация, несмотря на свою точность, значительно замедлялась в работе из-за реше-ния сверхдетерминированной системы уравнений.

Руководствуясь вышеприведенными рассуждениями, я выбрал метод, ос-нованный на использовании трех маячков с наилучшими сигнал-шумовыми показателями.

33

* Реализация

Реализованная библиотека находится в открытом доступе и расположена в репозитории по веб-адресу:

https://github.com/EnlightenedCSF/IBeaconLocation

1. Реализация алгоритма, основанного на сравнении локационных отпечатков

Локационные ¾отпечатки пальцев¿ – набор признаков, присвоенных неко-торой заведомо известной точке пространства [32]. В рамках задачи будем считать, что этим набором признаком является вектор из эталонных показа-

телей RSSI, принятых от каждого из маячков.

Смысл алгоритма тогда будет состоять в следующем: пользователь в неко-торый момент времени определяет ряд значений RSSI. Этот набор сравнива-ется с остальными с целью нахождения ближайшего ¾отпечатка¿, значения которого отличается минимально в рамках принятой метрики. И так как по-ложение каждого из ¾отпечатка¿ заранее известно, становится возможным

определить положение пользователя.

Данную задачу можно рассматривать как задачу классификации. Так,

каждый отпечаток F Pi в точке пространства i (i = 1; N) может быть пред-ставлен как F Pi = fRSSI1F Pi ; RSSI2F Pi ; :::; RSSIMFPi g, и соответствующая ему локация есть LFi P = (Xi; Yi).

Если представить положение пользователя как T , то ему сопоставим от-печаток S = fRSSI1T ; RSSI2T ; :::; RSSIMT g. После этого возможно найти ди-

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| станцию между S и F Pi: |  |  |  |  |  |  |  |
| DFT Pi | = v |  |  |  |  | (6) |  |
| N | (RSSIjF Pi |  | RSSIjT )2 |  |
|  | uj=1 | |  |  |  |  |
|  | uX | |  |  |  |  |  |

t

Будем считать M количеством маячков, участвующих в рассмотрении. Теперь на основе расстояния можем вычислить вероятность того, что объект

34

находится около отпечатка F Pi с измерениями S:

|  |  |  |  |
| --- | --- | --- | --- |
| P (F PijS) = | 1 | (7) |  |
| DFT Pi |  |

Тогда правило определения локации можно сформулировать так: искомая точка – i – если P (F PijS) > P (F PjjS), где i; j = 1; N. Ей соответствует локация LFi P .

В реальности искомая позиция пользователя не является дискретной, и поэтому возникает необходимость интерполяции. Эта процедура осуществля-ется с помощью следующего преобразования, представленного ниже:

N

X

LT (x; y) = P (F PijS)LFi P (x; y)

i=1

где (x; y) - координаты F P .

В работе [32] предложено дополнение для данного подхода, называемое Budgeted Dynamic Exclusion (BDE, "бюджетное динамическое исключение"). Суть модификации сводится к следующему: сравниваются показания j и от-печаток i. Если некоторая пара значений RSSI, соответствующих одному и тому же маяку, меньше установленной константы, именуемой бюджетом, то эти два значения выкидываются из вектора с остальными значениями RSSI. Это приводит к тому, что оцениваемое формулой 6 Евклидово расстояние уменьшается, и, соответственно, степень доверия, вычисляемая с помощью формулы 7, увеличивается.

Главным препятствием в работе такой системы является дифракция, от-ражение и рассеяние сигнала, принимаемого от маяков [33]. Кроме того, воз-никают сложности, связанные с выходом из строя маячков. Ведь в таком случае необходимо либо переснимать показания сигнал-шума от оставшихся маячков заново, либо предусмотреть дополнительную логику в приложении и работе алгоритма в целом. Оба способа достаточно затратны по времени.

Хотя данный метод нахождения пользовательской локации был исполь-зован, например, на конференции GeekPicnic, проведенного в Москве в 2015г. [34], в итоговую версию библиотеки, разработанной в рамках данной работы, алгоритм не включен.

35

1. Реализация алгоритма локализации по Монте-Карло

Метод также известен как ¾локализация с помощью фильтра частиц¿. В рамках данного метода алгоритм генерирует сотни частиц, представля-

ющих возможные будущие положения пользователя. Вычисляются рассто-яния до маяков и учитываются значения их сигналов. На основании этих актуальных показаний и несоответствия предсказаний этим данным, часть частиц фильтруется, то есть выходит из рассмотрения, а оставшаяся часть участвует в дальнейших вычислениях. Это происходит в результате обновле-ния весов, присвоенных каждой из частиц.

Проблему локализации можно сформулировать как проблему определе-ния апостериорной вероятности p(xkjz1:k). Начальное распределение p(x0)

считается равномерным по всем возможным локациям [x y]T . Рассматривая локализацию по Монте-Карло, требуемое распределение

p(xkjz1:k) может быть представлено в виде множества взвешенных значений:

Sk = fxi; !ig; i = 0; 1; 2; :::; Np

где !i - вес частицы - представляет собой фактор важности, для которого

выполняется равенство PNp !i = 1.

i=1

Апостериорная вероятность может быть аппроксимирована с помощью следующего выражения:

|  |  |  |
| --- | --- | --- |
| 1 | X (xk xki) |  |
| p(xkjzk) Np |  |

где - дельта-функция Дирака. Для достаточно большого Np аппроксимация отклоняется от истинного значения искомого распределения незначительно.

Алгоритм функционирует в несколько стадий:

1. Предсказание. На этом этапе апостериорное распределение

p(xkjxk 1; uk 1) в момент времени k предсказывается на основании со-стояния p(xk 1jz1:k 1) и управляющего вектора uk 1. Множество частиц

Sk 1 соответствует состоянию xk 1. Вектор uk 1 должен быть применен к каждой частице из Sk 1. Это дает новый набор Sk0 = x0i; !0i;

i = 0; 1; 2; :::; Np. Заметим, что !0i = !i.

36

1. Обновление. На этом шаге принимается во внимание модель измере-

ний: каждая частица из Sk0 изменяет вес на основании степени схожести с p(zkjxik); i = 0; 1; 2; :::; Np. После этого образуется новый набор ча-стиц Sk.

1. Вырождение. После нескольких итераций большинство частиц име-ют незначительный вес, и лишь немногие из них вносят значительный вклад в вычисления. Это стало причиной разработок многих алгорит-мов ресэмплинга, позволяющих выделить только необходимую часть частиц. Чтобы избежать накладных расходов на ресэмплинг во вре-мя каждой из итераций, зачастую вычисляется Eﬀective Sample Size, ESS. На основании этого функционала, ресэмплинг применяется лишь в случае, когда величина ESS опускается ниже определенного порога. Ресемплинг зачастую, помима отсева незначащих частиц, дуплицирует те, значения весов которых значительны.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | var(!i) | | 1 | | | Np |  |  |
| 2 |  |  | t |  |  |  | Xi | 2 |  |
|  |  |  |  |  | i |  |
| cvt | = |  |  | = | |  | (Np! | 1) |  |
|  | E2(!ti) | Np |  |
|  |  |  |  |  |  |  | =1 |  |  |
| ESSt | = |  | Np |  |  |  |  |  |  |
| (1 + cvt2) | | | | |  |  |  |

Однако, в рамках решения задачи локализации с помощью данного метода возникает так называемая проблема потери частиц ("particle deprivation"). Суть этой проблемы, подробно описанной в [35], сводится к тому, что ал-горитм во время проведения этапа ресэмплинга может отбросить частицы, являющиеся значимыми для локализации. Это особенно значимо для вари-антов реализации с небольшим (M 50) числом частиц.

В качестве возможного, хотя и не однозначного решения возможно ге-нерировать дополнительно некоторое количество частиц каждую итерацию работы алгоритма.

Тем не менее, из-за неустойчивости в формировании результата данным алгоритмом, было решено в рамках реализации библиотеки не включать дан-ный метод.

37

1. Реализация алгоритма трилатерации, основанного на определении силового центра

Другим вариантом реализации метода трилатерации может быть алго-ритм, предложенный бельгийскими исследователями В.Пьерло, М.Урбин-Шоф-фреем и М.Ван Другенброек, которые свели проблему к решению системы линейных уравнений [36].

Чтобы объяснить суть метода, прежде всего, необходимо ввести понятие о так называемом силовом центре (“power center” или “radial center”) трех окружностей – уникальной точке, обладающей равной мощностью (“power”) по отношению к этим окружностям. Мощность P точки при этом может быть найдена по формуле:

|  |  |
| --- | --- |
| Pc;p = (x xc)2 + (y yc)2 R2 | (8) |

где x; y – координаты точки, для которой вычисляется мощность, xc; yc –

координаты центра окружности радиуса R. Согласно формуле 8, если точ-ка лежит на окружности, мощность равна нулю; меньше нуля в случае, ко-гда точка находится внутри окружности и больше нуля, когда находится вне окружности.

Силовой линией (“power line”) двух окружностей назовем такое геометри-ческое место точек, обладающих равной мощностью по отношению к этим окружностям.

Она перпендикулярна прямой, соединяющей центры данных окружностей и проходит через область их взаимного пересечения (в случае наличия тако-вого). Если рассматриваются три окружности, такие, что никакие две из них не расположены в одной точке, и центры которых не коллинеарны, то для них можно определить силовой центр – точкой пересечения силовых линий. На приведенном ниже рисунке силовой центр обозначен точкой.

Нахождение координаты точки пересечения двух прямых – задача, име-ющая линейную сложность. В данном случае мы можем, учитывая формулу

38

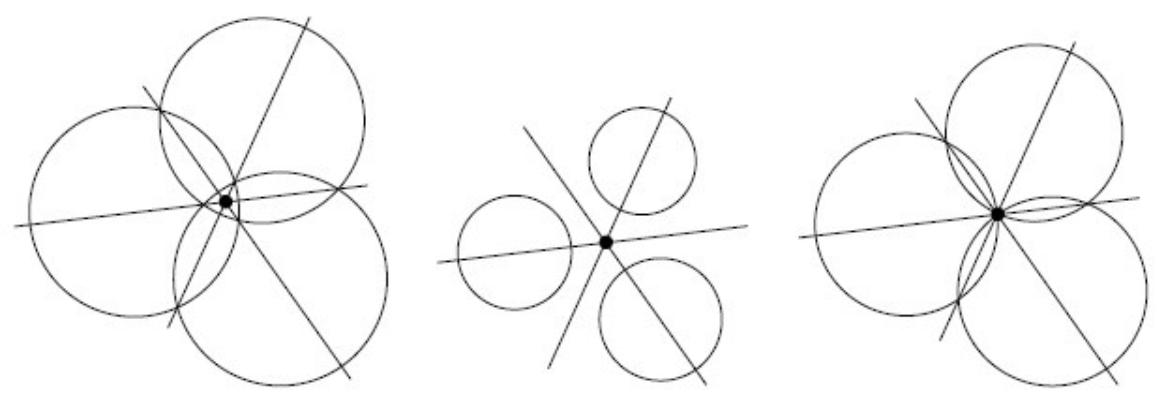


Рис. 5: Определение силового центра для окружностей различных радиусов

8, просто приравнять мощности.

(x x1)2 + (y y1)2 R12 = (x x2)2 + (y y2)2 R22

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ) | x(x | 1 | x | ) + y(y | 1 | y | ) = | x12 + y12 R12 |  | x22 + y22 R22 |  |
|  | 2 |  | 2 | 2 | | 2 |  |

) x(x1 x2) + y(y1 y2) = k1 k2

Введен дополнительный коэффициент k, равный мощности источника от-

носительно некоторой окружности, деленный на два:

x2 + y2 R2

kn = n n n

2

Наконец, чтобы определить координаты точки пересечения силовых ли-

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ний, достаточно решить систему линейных уравнений: | | | |  |
| 8x(x2 | x3) + y(y2 | y3) = k2 | k3 | (9) |
| x(x1 | x2) + y(y1 | y2) = k1 | k2 |  |
| > |  |  |  |  |
| < |  |  |  |  |

>

:

x(x3 x1) + y(y3 y1) = k3 k1

Можно заметить, что одно из уравнений в системе может быть получено

путем сложения двух других, что является, в свою очередь, доказательством пересечения трех силовых линий лишь в одной точке. Ее точные координаты

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| можно определить по следующим формулам: | | | | | | | | | |  | x3 |  | k3 | | |  |
|  | k2 | | k3 | y2 | y3 | | |  | x2 | |  | k2 |  | | |  |
|  |  | k1 | k2 | y1 |  | y2 |  |  |  | x1 | x2 | k1 |  | k2 |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| xr = |  |  |  |  |  |  |  | ; yr = |  |  |  |  |  |  |  |  |
|  |  |  | D | |  |  |  |  |  |  | D |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

39

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| D = x1 | | x2 | | y1 y2 | | |  | = | x2 | | y2 | 1 |  |
|  |  |  |  |  |  |  |  |  |  | x1 | y1 | 1 |  |
| x2 |  | x3 | y2 |  | y3 |  | x3 | y3 |  |  |
|  |  |  |  |  |  |  |  | 1 |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Определитель матрицы D 6= 0, если центры окружностей не коллине-

арны, тогда же, соответственно, существует решение приведенной системы

уравнений 9. В противном случае найти координаты пересечения прямых в

явном виде нельзя.

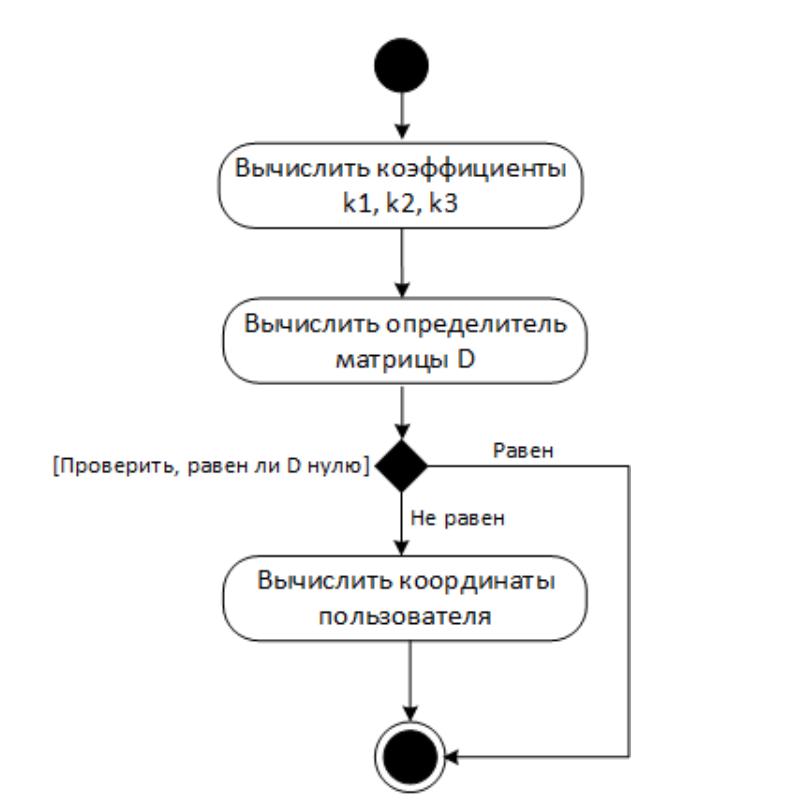


Рис. 6: Activity-диаграмма рассматриваемого алгоритма

1. Реализация алгоритма трилатерации, основанном на пересечении сфер
   * точки зрения геометрии проблема трилатерации - проблема нахожде-ния точки пересечения трех сфер в пространстве. Для того, чтобы эту точку найти, необходимо решить систему уравнений. Обычно для упрощения про-межуточных вычислений полагается, что центры этих сфер лежат на плоско-сти, определяемой уравнением z = 0. Кроме того, скажем, что центр первой сферы находится в начале координат, а центр второй лежит на оси x. Оче-видно, что данная формулировка задачи не уменьшает общности, и задача по сути осталась той же. С другой стороны, к такому виду может быть приве-ден любой набор из трех сфер. Решив уравнение в преобразованной системе

40

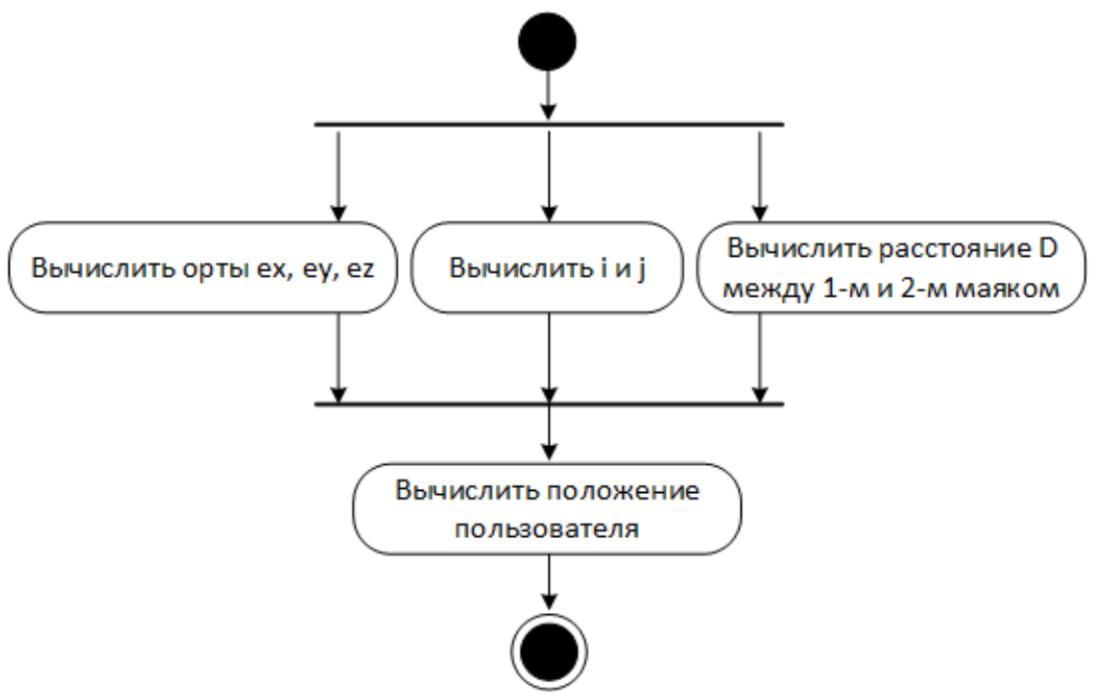


Рис. 7: Activity-диаграмма алгоритма пересечения сфер

координат, можно вернуться к начальной.

С первого взляда может показаться, что задача определения координаты устройства по нескольким маячкам схожа с задачей, которая решается спут-никами GPS, и, соответственно, мы тоже должны решать задачу о пересече-нии сфер. К тому же, маяки могут быть закреплены в помещении на разных высотах. Но на практике этим можно пренебречь, отбросив z-координату. Предпосылкой для этого является и то, что хотя маячки могут находить-ся на различной высоте, на одной и той же высоте находится пользователь. Таким образом, задача сводится к работе с окружностями.

1. Адаптивный геометрический алгоритм

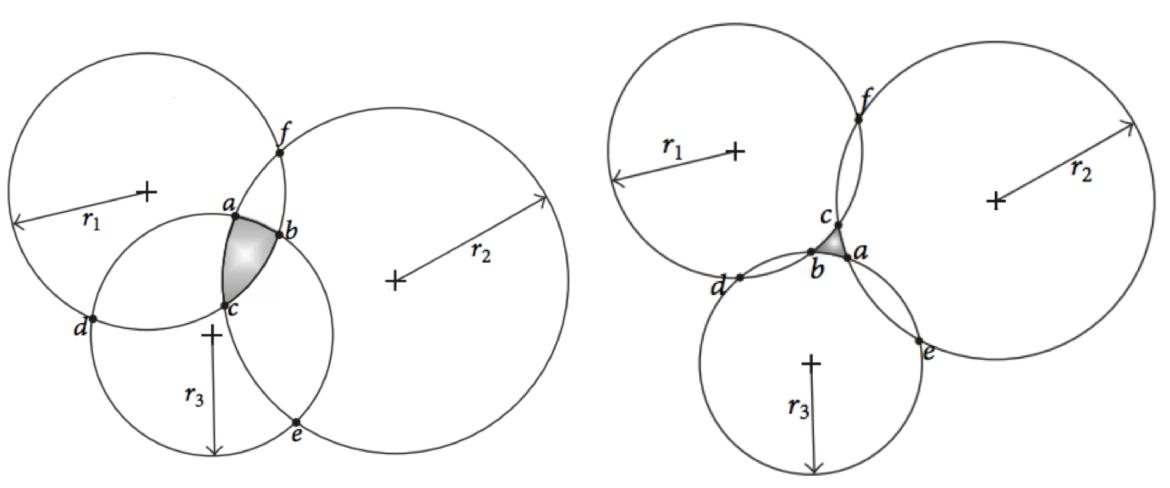
Прежде всего, следует объяяснить принцип работы геометрического ал-горитма.

Геометрический алгоритм опирается на взаимоположение окружностей и их радиусы, и результирующие точки пересечения имеют ключевое значение. На рисунке 8 показаны наиболее часто возникающие сценарии (точки, соот-ветствующие пересечениям окружностей, обозначены как a; b; c; и так далее).

В случае, изображенном на рисунке 8a, нас интересуют только три точ-ки из шести, так как они формируют область, принадлежащую всем трем окружностям.

На рисунке 8b более сложный сценарий, ведь общей области пересече-ния всех трех окружностей нет. Следовательно, мы должны воспользоваться

41



|  |  |  |
| --- | --- | --- |
| (a) | (b) |  |
|  |  |

Рис. 8: Наиболее часто встречающиеся сценарии при работе геометрического алгоритма

другой логикой для нахождения искомой области. Конкретнее, для каждй пары точек, образовавшейся в результате пересечения пары окружностей, нас удовлетворяет та точка, которая находится ближе к третьей окружности (например, из точек a и e будет выбрана a, так как она находится ближе к окружности с радиусом r1).

Наконец, искомым положением пользователя является центроид, вычис-лимый по формуле 10.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| x = | 1 | K | xl; | y = | 1 | K | yl; l = 1; 2; :::; K | (10) |  |
|  | Xl |  | X |  |
|  | K |  | K | |  |  |  |
|  | =1 |  | l=1 |  |  |  |
|  |  |  |  |  |  |  |  |

K в представленной выше формуле означает количество подходящих то-чек пересечений. Для случая с тремя окружностями K = 3.

В работе [37] представлена идея адаптивного геометрического алгорит-ма. В сущности, геометрический алгоритм, описанный выше, является част-ным случаем, а точнее - первой итерацией адаптивного геометрического ал-горитма.

Благодаря данной первой итерации получен набор точек пересечения. Суть алгоритма сводится к пропорциональному увеличению или уменьше-нию всех трех радиусов окружностей так, чтобы общая область пересечений (см. случай на рисунке 8a) была наименьшей. Конечный этап - нахождение положения пользователя - осуществляется по формуле 10. Алгоритм работы

42

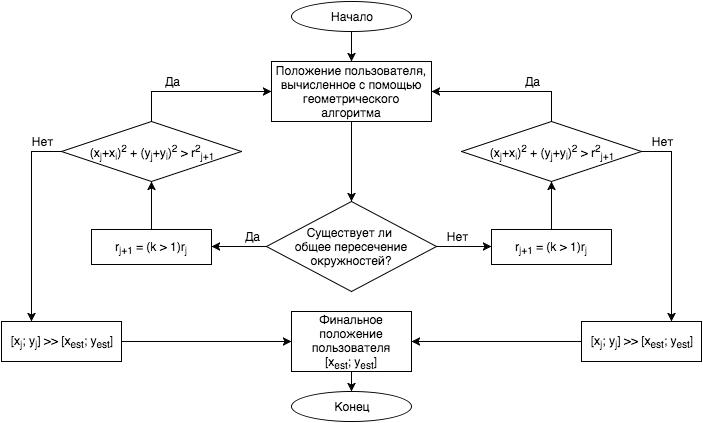


Рис. 9: Диаграмма потока данных адаптивного геометрического алгоритма

проиллюстрирован на рисунке 9.

В настоящей работе предложена модификация алгоритма для увеличения скорости его работы. Суть изменения сводится к следующему: для случая, по-казанного на рисунке 8a, вместо пропорционального уменьшения радиусов, вычисляются координаты центроида точек пересечения. Для случая, пока-занного на рисунке 8b, алгоритм функционирует, как описано выше.

Полный листинг конечной версии алгоритма представлен в приложении

3.

1. Основные классы библиотеки
2. BeaconLocation

Класс BeaconLocation – центральный класс библиотеки. Предполагает-ся, что пользователь начинает работу с инстанцирования этого класса. Во время создания BeaconLocation создает необходимые объекты CoreLocation

(CLBeaconRegion, CLLocationManager), а так же подписывается на обновле-ния данных, получаемых от маячков.

43

Кроме того, здесь же происходит создание классов Processor, который ответственен за вычисление новых координат пользователя, и Floor, кото-рый является хранилищем всех идендификаторов маячков, их координат и очередных показаний.

1. Floor

Класс Floor инкапсулирует маячки, расположенные на этаже. Также пре-доставляет набор методов, с помощью которых пользователь может добавить новые маячки, получить информацию об уже имеющихся и удалить указан-ные. Предполагается, что задающие их координаты измеряются в метрах, так как обновления значений расстояний до маячков тоже формируются в метрах самой библиотекой CoreLocation.

1. Processor

Класс Processor - ядро библиотеки. По умолчанию он подписывается на события об изменении вычисленных расстояний до маячков, и вычисляет на их основе новое положение пользователя. Пользователю предлагаются 3 гото-вых алгоритма, а также делегат, с помощью которого он может реализовать и использовать собственный метод. Кроме того, есть возможность использовать комбинацию из имеющихся алгоритмов для достижения лучшего результата.

Степень доверия результату каждого из алгоритмов может быть указана в параметрах метода

-(void)setAlgorithmsAndTrusts:(NSDictionary<NSNumber\*, NSNumber\*>\*)algorithms

Переданные в паре с каждым из алгоритмов числа нормируются к еди-нице. Пусть первый алгоритм вычислил положение пользователя как точку с координатами (1; 1). Второй используемый алгоритм - как точку (5; 5). По умолчанию, все алгоритмы эквивалентно важны, и общим результатом будет среднее арифметическое: точка (3; 3). Если второй алгоритм вдвое более ва-жен, чем первый, то результат будет вычислен как 0; 66 (5; 5) + 0; 33 (1; 1), то есть (3; 66; 3; 66).

44

1. Attractor и Spot

Для того, чтобы сделать приложение более отзывчивым во время при-ближения пользователя к некоторому объекту, в библиотеке предусмотрены классы Attractor и Spot.

Spot - класс, представляющий некоторую ¾горячую точку¿ в локацион-ном контексте приложения, другими словами, место, в котором мы ожидаем увидеть пользователя чаще, чем в другом (витрина в магазине, экспонат в музее).

Работа с ними осуществляется через класс Attractor. После того, как точ-ки добавлены, а положение пользователя вычислено, Attractor приближает положение пользователя к ближайшей ¾горячей точке¿, основываясь на опре-деленным пользователем коэффициенте (¾сила притяжения¿ - attractorPo-wer). Учтена и ¾мертвая зона¿, внутри которой Attractor перестает при-ближать положение пользователя к указанной отметке.

Все это позволяет создавать более дружественный интерфейс разрабаты-ваемого приложения.

1. Структура библиотеки

На рисунке 10 показана структура классов разработанной библиотеки. Классы AlgorithmPowerCenter, AlgorithmSphereIntersection и Algorithm-AGA являются Objective-C-обертками над одноименными С-классами, содер-жащими реализации алгоритмов.

Пользователь может начать работу с библиотекой двумя способами:

Вручную скопировав все файлы с исходными кодами библиотеки в соб-ственный проект. Следует отметить, что программисты, использующие как Objective-C, так и Swift в своих проектах, могут подключить раз-рабатываемую библиотеку.

Подключив библиотеку через менеджер управления зависимостей Cocoa-Pods.

После этого он может приступить к первоначальной настройке в рам-ках приложения: инстанцировать объект библиотеки, добавить необходимую

45

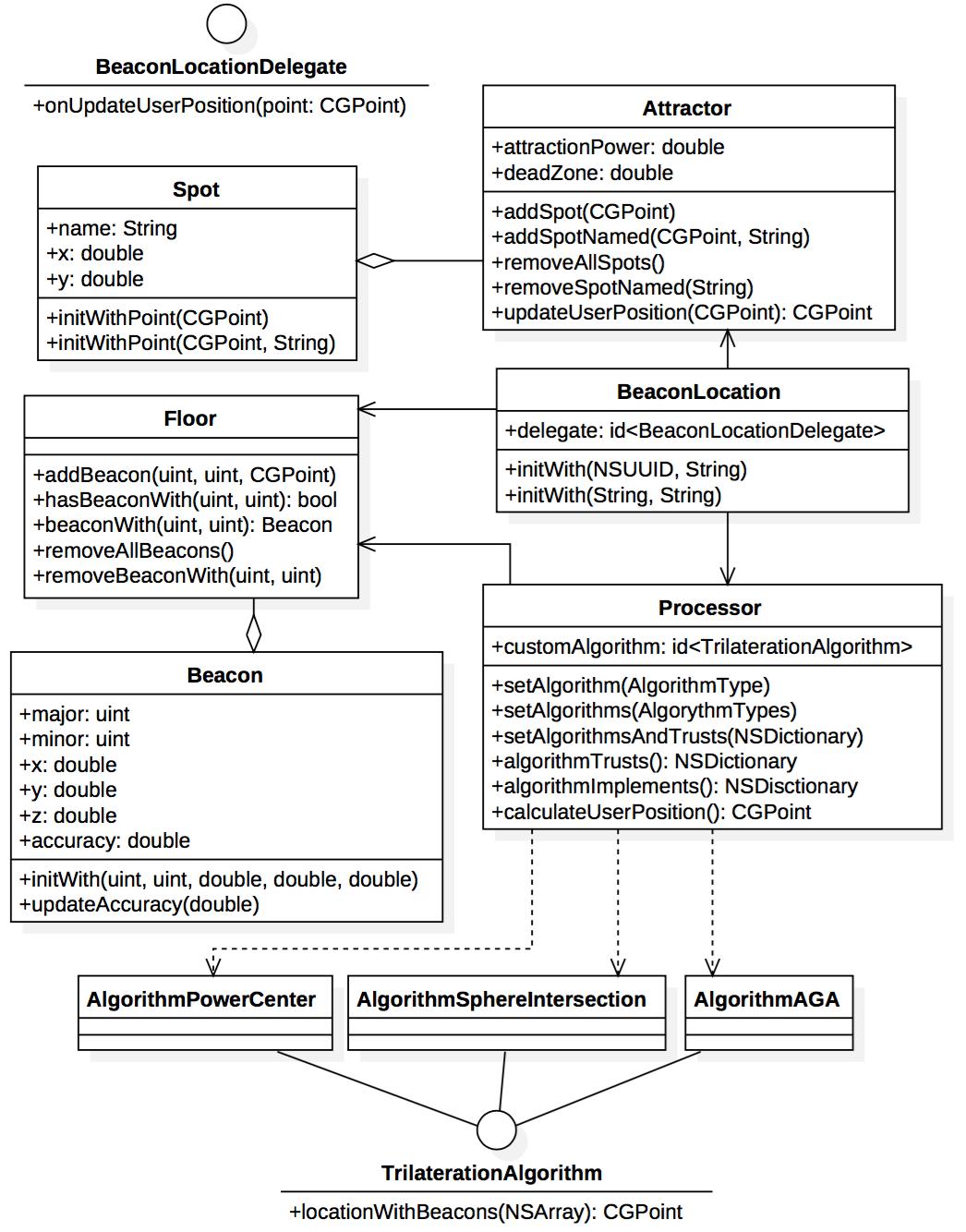


Рис. 10: Диаграмма классов разработанной библиотеки

информацию о маячках и подписавшись на обновление пользовательской ло-кации.

Данная процедура проиллюстрирована листингом в приложении 4.

1. Экспериментальные результаты применения синер-гетического метода

Предложенная архитектура библиотеки предполагает получение синерги-теческого эффекта от использования различных трилатерационных методов

46

для достижения наилучшей итоговой точности.

Кроме того, в работе маячков существуют некоторые особенности, кото-рые необходимо учитывать. Из этого следует вывод, что именно комбинация алгоритмов дает более надежный результат, ведь некоторые алгоритмы мо-гут показывать себя сильнее или слабее в некоторых сценариях.

К отрицательным сторонам используемых маячков можно отнести следу-ющие особенности:

Периодическое пропадание сигнала. Раз в некоторый промежу-ток времени (вплоть до 5с) величина сигнала не может быть вычислена

правильно, и будет отражено значение 1. Для нивелирования данного эффекта можно, например, принимать предыдущее значение, получен-ное от маячка, считая, что оно успело измениться незначительно.

Интерференция сигналов. Вследствие интерференции сигналов при неправильной установке маячков данные могут отличаться от реальных достаточно значительно. Следует принимать во внимание как принци-пы оптимального расположения маячков, описанные в текущей работе, так и плотность расположения маячков в помещениях. Даже для доста-точно больших комнат или залов не следует монтировать маячки вдоль их периметра чаще, чем одно устройство на 4м периметра.

Помехи и препятствия. К отображению неверных данных может и привести, конечно, наличие препятствия между устройством пользова-теля и маяком. В этом можно легко убедиться, стоя напротив маяка и, снимая показатели сигнала, повернуться спиной к источнику. В данном случае в качестве рекомендации, опять же, можно привести советы по оптимальному расположению BLE-устройств.

* + проведении ряда экспериментов отмечено, что на адаптивный геомет-рический алгоритм общий шум показателей сигналов влияет в меньшей сте-пени: с геометрической точки зрения, большие показатели - большие радиусы окружностей, и центроид точек пересечения, как результат, смещается незна-чительно.

47

Можно утверждать, что используемые методы надежны: каждый спосо-бен сформировать конечный результат, даже если полученные показания од-новременно малы или велики.

48

Заключение

В рамках данной работы была разработан готовый программный продукт: библиотека для indoor-локализации для платформы iOS. Согласно выдвину-тым требованиям:

1. Произведен детальный анализ различных алгоритмов трилатерации, выделены их сильные и слабые стороны;
2. В библиотеку включены несколько реализаций алгоритмов, выделен-ных в результате анализа, а так же предоставлена возможность исполь-зовать их комбинации;
3. Работа алгоритмов максимально оптимизирована как по скорости ра-боты, так и по объему занимаемой памяти;
4. Библиотека легко подключается в iOS-приложение, а ее первоначальная настройка занимает минимальный объем кода.

Однако даже на данном этапе можно выделить направления для дальней-ших улучшений.

Прежде всего, можно передавать показания акселерометра, магнитомет-ра и гироскопа в качестве управляющих воздействий в фильтр Калмана, и, как результат, во время движения пользователя в большей степени доверять показаниям ИНС, а во время его остановок - показаниям маяков.

Так как навигация внутри помещений может быть трактована как подза-дача нахождения положения пользователя в целом, возможна лучшая инте-грация с сервисами навигации GPS и ¾ГЛОНАСС¿.

49

Список литературы

|  |  |  |  |
| --- | --- | --- | --- |
| [1] Siewart Steﬀen. | Airfy | beacons - make your smart | home even |
| smarter. | URL: | https://www.indiegogo.com/projects/ | |
| airfy-beacon-make-your-smart-home-even-smarter/. | | | 2016. |
| (дата обращения: 10.05.16г.). | | |  |

1. Crunchtime! URL: http://www.crunchtime.com/. 2016. (дата об-ращения: 10.05.16г.).

[3] Mlb.com ballpark. URL: https://itunes.apple.com/app/

id513135722?mt=8. 2016. (дата обращения: 10.05.16г.).

1. Mingleton. URL: https://itunes.apple.com/ru/app/mingleton/ id807307276. 2015. (дата обращения: 03.05.16г.).
2. Знакомьтесь: биконы! Сессия-интенсив про электронные маячки, для бизнесменов и разработчиков. URL: http://2014.secr.ru/lang/ ru/program/submitted-presentations/. 2014. (дата обращения: 17.08.15г.).
3. Российские программисты не спали 24 часа - интервью с Джоном Ланном из paypal. URL: http://goo.gl/YHqQre. 2015. (дата обращения: 06.03.16г.).
4. Первый проект внедрения технологии ibeacon в пространствах музея появился в России. URL: http://www.retail-loyalty.org/news/. 2015. (дата обращения: 06.04.15г.).

|  |  |  |  |
| --- | --- | --- | --- |
| [8] Rite | aid deploys | proximity | beacons in more than 4,500 |
| us | stores. | URL: | http://www.zdnet.com/article/ |

rite-aid-deploys-proximity-beacons-in-more-than-4500-us-stores/. 2016. (дата обращения: 18.01.16г.).

1. Hillshire increases sales w/ ibeacon, 20x increase in purchase intent, 500x increase over average mobile ads. URL: http://9to5mac.com/2014/07/ 22/. 2014. (дата обращения: 22.07.15г.).

50

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| [10] | Inmarket | & | zatarain’s | launch | world | first | cpg |
|  | ibeacon | campaign. | | URL: | http://www.ibeacon.com/ | | |
|  | inmarket-zatarains-launch-world-first-cpg-ibeacon-campaign/. | | | | | | |
|  | 2014. (дата обращения: 04.04.15г.). | | | |  |  |  |
| [11] | Timberland, | kenneth | cole track | shoppers | who opt-in for | | deals. |
|  | URL: | http://adage.com/article/datadriven-marketing/ | | | | | |

timberland-tests-store-tracking-app-swirl-offers/243811/. 2013. (дата обращения: 08.05.16г.).

1. Virgin atlantic lights the way with apple’s ibeacon technology at heathrow. URL: https://blog.virgin-atlantic.com/t5/Our-Future/

Virgin-Atlantic-lights-the-way-with-Apple-s-iBeacon-technology/ ba-p/26359#.VzMaUxV96Rt. 2014. (дата обращения: 04.07.15г.).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| [13] Московские | аэропорты | могут | начать | использовать |
| ibeacon. | URL: | http://appleinsider.ru/sluxi/ | | |

moskovskie-aeroporty-mogut-nachat-ispolzovat-ibeacon.html.

2014. (дата обращения: 05.08.15г.).

1. Macy’s takes ibeacon technology nationwide, installing more than 4,000 devices. URL: http://marketingland.com/macys-100162. 2014. (дата обращения: 09.09.15г.).
2. Одно успешное внедрение ibeacon: 200 маячков для РИФ+КИБ 2014. URL: https://habrahabr.ru/post/225713/. 2014. (дата обращения: 03.07.15г.).

[16] How mobile is transforming the shopping experience in stores.

URL: https://www.thinkwithgoogle.com/research-studies/

mobile-in-store.html. 2015. (дата обращения: 12.11.15г.).

1. ¾Успешных кейсов в России - ноль¿ Эксперт о технологии ibeacon, ко-торая умеет продавать в оффлайне с помощью mobile. URL: http:// www.therunet.com/experts/4749-uspeshnyh-keysov-v-rossii-nol. 2015. (дата обращения: 08.09.15г.).

51

1. Навигация в помещениях с ibeacon и ИНС. URL: https://habrahabr. ru/post/245325/. 2014. (дата обращения: 10.12.15г.).
2. Indoor навигация с ibeacon в ios7. URL: https://habrahabr.ru/ company/touchinstinct/blog/195104/. 2013. (дата обращения: 04.03.15г.).
3. Gps tracking down to the centimeter. URL: http://phys.org/news/ 2016-02-gps-tracking-centimeter.html. 2016. (дата обращения: 21.01.16г.).
4. Secure beacons: Overview & options. URL: http://www.slideshare. net/localzco/beacon-security-overview. 2015. (дата обращения: 20.04.15г.).
5. Hacking the ces scavenger hunt. URL: http://makezine.com/2014/ 01/03/hacking-the-ces-scavenger-hunt/. 2014. (дата обращения: 04.04.15г.).

[23] Corebluetooth doesn’t let you see beacons. URL:

http://developer.radiusnetworks.com/2013/10/21/ corebluetooth-doesnt-let-you-see-ibeacons.html. 2013. (дата обращения: 22.03.15г.).

1. Эйкхофф П. Основы идентификации систем управления. М.: Мир, 1975. Vol. 680.
2. Дусеев ВР, Рудь МН, Мальчуков АН. ИССЛЕДОВАНИЕ ФИЛЬТРА

КАЛМАНА В ПРИЛОЖЕНИИ К ФИЛЬТРАЦИИ ДАННЫХ ГЛУБИ-НЫ С СЕНСОРА kinect.

1. Давид Худавердян. Фильтр Калмана. URL: https://habrahabr.ru/ post/166693/. 2013. (дата обращения: 15.02.15г.).
2. Фильтр Калмана Введение. URL: https://habrahabr.ru/post/ 140274/. 2012. (дата обращения: 15.02.15г.).

52

1. Han Guangjie, Choi Deokjai, Lim Wontaek. Reference node placement and selection algorithm based on trilateration for indoor sensor networks //

Wireless Communications and Mobile Computing. 2009. Vol. 9, no. 8. Pp. 1017–1027.

1. Bulusu Nirupama, Heidemann John, Estrin Deborah. Adaptive beacon placement // Distributed Computing Systems, 2001. 21st International Conference on. / IEEE. 2001. Pp. 489–498.
2. Mironovova Martina, Havlis H. Calculation of gdop coeﬃcient // CTU in Prague: Faculty of Electrical Engineering. 2011.
3. Comparison of anchor selection algorithms for improvement of position estimation during the wi-fi localization process in disaster scenario / Oleksandr Artemenko, Thierry Simon, Andreas Mitschele-Thiel et al. // Local Computer Networks (LCN), 2012 IEEE 37th Conference on / IEEE. 2012. Pp. 44–49.
4. Elbes Mohammed, Al-Fuqaha Ala, Anan Muhammad. A precise indoor localization approach based on particle filter and dynamic exclusion techniques // Network Protocols and Algorithms. 2013. Vol. 5, no. 2. Pp. 50–71.
5. Survey of wireless indoor positioning techniques and systems / Hui Liu, Houshang Darabi, Pat Banerjee, Jing Liu // Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on. 2007. Vol. 37, no. 6. Pp. 1067–1080.

[34] Бабаев Александр. ibeacon. Мифы и реальность. URL: https://

habrahabr.ru/post/278689/. 2016. (дата обращения: 04.03.16г.).

1. Robust monte carlo localization for mobile robots / Sebastian Thrun, Dieter Fox, Wolfram Burgard, Frank Dellaert // Artificial intelligence. 2001. Vol. 128, no. 1. Pp. 99–141.
2. Pierlot Vincent, Urbin-Choﬀray Maxime, Van Droogenbroeck Marc. A new three object triangulation algorithm based on the power center of three

53

circles // Research and Education in Robotics-EUROBOT 2011. Springer,

2011. Pp. 248–262.

1. Brida Peter, Machaj Juraj. A novel enhanced positioning trilateration algorithm implemented for medical implant in-body localization //

International Journal of Antennas and Propagation. 2013. Vol. 2013.

54

Приложения

Приложение 1. Реализация алгоритма, основанного на

поиске силового центра

double \* c a l c u l a t e U s e r P o s i t i o n P o w e r C e n t e r ( double \* xs , double \* ys , double \* accs ) {

double \* result = ( double \*) malloc ( sizeof ( double ) \* D I M E N S I O N S ) ;

// k [ n ] = ( B [ n ]. x ^2 + B [ n ]. y ^2 - dist [ n ]^2 ) /2

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| double | \* ks | = | ( double | | \*) malloc ( sizeof ( double ) | | | | \* M I N \_ B E A C O N S ) ; |
| for ( size\_t | | i | = | 0; i | < M I N \_ B E A C O N S ; ++ i ) | | | { |  |
| ks [ i ] = | | ( xs [ i ]\* xs [ i ] | | | | + | ys [ i ]\* ys [ i ] - | accs [ i ]\* accs [ i ]) /2.0; | |
| } |  |  |  |  |  |  |  |  |  |
| // D = | | x1 | - x2 | | y1 - y2 | | | = | ( x1 - x2 ) \*( y2 - y3 ) | - | ( y1 - y2 ) \*( x2 - x3 ) |

* | x2 - x3 y2 - y3 |

double d = ( xs [0] - xs [1]) \*( ys [1] - ys [2]) - ( ys [0] - ys [1]) \*( xs [1] - xs [2]) ;

if ( fabs ( d ) < 1e -7) { result [0] = -1; return result ;

}

// X = | k1 - k2 y1 - y2 | = ( k1 - k2 ) \*( y2 - y3 ) - ( y1 - y2 ) \*( k2 - k3 )

* | k2 - k3 y2 - y3 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| double | x | = | ( ks [0] - ks [1]) \*( ys [1] - ys [2]) | - | ( ys [0] - ys [1]) \*( ks [1] - ks |
|  | [2]) ; | | |  |  |
| // Y = | | | x1 | - x2 k1 - k2 | = ( x1 - x2 ) \*( k2 - k3 ) | - | ( k1 - k2 ) \*( x2 - x3 ) |

* | x2 - x3 k2 - k3 |

double y = ( xs [0] - xs [1]) \*( ks [1] - ks [2]) - ( ks [0] - ks [1]) \*( xs [1] - xs [2]) ;

result [0] = x / d ; result [1] = y / d ;

return result ;

}

55

Приложение 2. Реализация алгоритма, основанного на

пересечении сфер

double \* c a l c u l a t e U s e r P o s i t i o n S p h e r e I n t e r s e c t i o n ( double \* xs , double \* ys ,

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| double \* accs ) { | | | | |  |  |  |  |  |  |  |  |  |  |  |
| double | \* result = ( double | | | | | | \*) malloc ( sizeof ( double ) | | | | | \* | D I M E N S I O N S ) ; | |  |
| double | temp | | = | ( xs [1] | - | xs [0]) \*( xs [1] - xs [0]) | | | | | + | ( ys [1] | | - ys [0]) \*( ys [1] | - |
|  | ys [0]) ; | | | |  |  |  |  |  |  |  |  |  |  |  |
| double | exx | = | ( xs [1] | | - | xs [0]) | | / | sqrt ( temp ) ; | |  |  |  |  |  |
| double | exy | = | ( ys [1] | | - | ys [0]) | | / | sqrt ( temp ) ; | |  |  |  |  |  |
| double | p3p1x | | = | xs [2] | - | xs [0]; | | |  |  |  |  |  |  |  |
| double | p3p1y | | = | ys [2] | - | xs [0]; | | |  |  |  |  |  |  |  |
| double ival = exx \* p3p1x + exy \* p3p1y ; | | | | | | | | | | |  |  |  |  |  |
| double | p3p1i | | = | ( xs [2] | | - | xs [0] | | - | exx ) \*( xs [2] - | xs [0] | | - | exx ) + ( ys [2] - | ys |
|  | [0] | - | exy ) \*( ys [2] | | | | - ys [0] | | | - exy ) ; |  |  |  |  |  |
| double | eyx | = | ( xs [2] | | - | xs [0] | | - | exx ) / sqrt ( p3p1i ) ; | | | |  |  |  |
| double | eyy | = | ( ys [2] | | - | ys [0] | | - | exy ) / sqrt ( p3p1i ) ; | | | |  |  |  |
| double d = sqrt ( temp ) ; | | | | | |  |  |  |  |  |  |  |  |  |  |
| double jval = ( eyx \* p3p1x ) + ( eyy \* p3p1y ) ; | | | | | | | | | | |  |  |  |  |  |
| double | xval | | = ( accs [0]\* accs [0] | | | | | | - accs [1]\* accs [1] | | | + | d \* d ) / (2\* d ) ; | |  |
| double | yval | | = ( accs [0]\* accs [0] | | | | | | - accs [2]\* accs [2] | | | + | ival \* ival + jval \* jval | | |
|  | ) /(2\* jval ) - ( ival / jval ) \* xval ; | | | | | | | | | |  |  |  |  |  |

result [0] = xs [0] + exx \* xval + eyx \* yval ; result [1] = ys [0] + exy \* xval + eyy \* yval ;

return result ;

}

Приложение 3. Реализация адаптивного геометрического

алгоритма

|  |  |  |
| --- | --- | --- |
| # define | M A X \_ I T E R A T I O N S | 500 |
| # define | Points | struct point \* |
| # define | e n o u g h P o i n t s | 7 |
| # define | step | 0.1 f |

56

/\* \*

\* Struct r e p r e s e n t s a beacon or a point \*/

struct point { double x ; double y ; double r ;

};

|  |  |  |  |
| --- | --- | --- | --- |
| /\* \* |  |  |  |
| \* | Checks | whether | the point belongs to all three circles |
| \* |  |  |  |
| \* | @param | x | x |
| \* | @param | y | y |

* @param points Beacons
* @return 1 if true , 0 o th erw is e

\*/

int i s P o i n t B e l o n g T o A l l C i r c l e s ( double x , double y , Points points ) {

int belongs = 1;

for ( size\_t i = 0; i < M I N \_ B E A C O N S ; ++ i ) {

double dist = g e t D i s t a n c e (x , y , points [ i ]. x , points [ i ]. y ) ;

if ( dist > ( points [ i ]. r + 1e -2) ) {

belongs = 0;

break ;

}

}

return belongs ;

}

/\* \*

\* Returns the centroid c o o r d i n a t e of given points

\*

\* @param points Array of points

\* @param size Its size

\*

\* @return Centroid c o o r d i n a t e s

\*/

Points g etC en te r ( Points points , size\_t size ) {

Points center = ( Points ) malloc ( sizeof ( struct point ) ) ;

center - > x = 0;

center - > y = 0;

for ( size\_t i = 0; i < size ; ++ i ) {

center - > x += points [ i ]. x ;

57

center - > y += points [ i ]. y ;

}

center - > x /= size ; center - > y /= size ;

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | return | | center ; | | |  |  |  |  |  |  |  |  |  |  |
| } |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| /\* \* |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| \* C a l c u l a t e s | | | | | all | circle - circle i n t e r s e c t i o n s and return | | | | | | | | | an array of |
|  |  | re su lt in g point | | | | | | |  |  |  |  |  |  |  |
| \* |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| \* | @param | | pointA | | | Circle | | | A |  |  |  |  |  |  |
| \* | @param | | pointB | | | Circle | | | B |  |  |  |  |  |  |
| \* | @param | | cnt | |  | R ef er en ce | | | | to | the r es ul ti ng array ’s | | | size |  |
| \* |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| \* @return Array of points | | | | | | | | | |  |  |  |  |  |  |
| \*/ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Points | | g e t C i r c l e C i r c l e I n t e r s e c t i o n ( Points | | | | | | | | | | pointA , | Points | | pointB , int \* cnt ) { |
|  | \* cnt = | | 0; |  |  |  |  |  |  |  |  |  |  |  |  |
|  | double | | r1 | = pointA - > r ; | | | | | |  |  |  |  |  |  |
|  | double | | r2 | = pointB - > r ; | | | | | |  |  |  |  |  |  |
|  | double p1x = pointA - > x ; | | | | | | | | |  |  |  |  |  |  |
|  | double p1y = pointA - > y ; | | | | | | | | |  |  |  |  |  |  |
|  | double p2x = pointB - > x ; | | | | | | | | |  |  |  |  |  |  |
|  | double p2y = pointB - > y ; | | | | | | | | |  |  |  |  |  |  |
|  | double | | d = g e t D i s t a n c e ( p1x , p1y , p2x , p2y ) ; | | | | | | | | | |  |  |  |
|  | // | if | too | far | | away , | | | or self | | con ta in ed | - can ’t | be | done |  |
|  | if | (( d >= | | ( r1 + r2 ) ) || ( d <= fabs ( r1 - r2 ) ) ) { | | | | | | | | | |  |  |
|  |  | return | | | 0; |  |  |  |  |  |  |  |  |  |  |
|  | } |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | double a = ( r1 \* r1 - r2 \* r2 + d \* d ) /(2\* d ) ; | | | | | | | | | | | |  |  |  |
|  | double h = sqrt ( r1 \* r1 - a \* a ) ; | | | | | | | | | | |  |  |  |  |
|  | double | | x0 | = | p1x | | + | a \*( p2x | | - | p1x ) / d ; |  |  |  |  |
|  | double | | y0 | = | p1y | | + | a \*( p2y | | - | p1y ) / d ; |  |  |  |  |
|  | double | | rx | = -( p2y - p1y ) \*( h / d ) ; | | | | | | | |  |  |  |  |
|  | double | | ry | = -( p2x - p1x ) \*( h / d ) ; | | | | | | | |  |  |  |  |
|  | \* cnt = | | 2; |  |  |  |  |  |  |  |  |  |  |  |  |
|  | Points | | result | | | = | ( Points | | | ) malloc (\* cnt | | \* sizeof ( struct | | | point ) ) ; |
|  | result [0]. x | | | | = | x0 | + | rx ; | |  |  |  |  |  |  |
|  | result [0]. y | | | | = | y0 | - | ry ; | |  |  |  |  |  |  |
|  | result [1]. x | | | | = | x0 | - | rx ; | |  |  |  |  |  |  |
|  | result [1]. y | | | | = | y0 | + | ry ; | |  |  |  |  |  |  |

58

return result ;

}

/\* \*

\* Returns an array of circle i n t e r s e c t i o n s

\*

* @param points Beacons

\* @param size Size of the array

\* @param cnt R ef er en ce to the size of the res ul ti ng array

\*

\* @return Array of points

\*/

Points g e t I n t e r s e c t i o n P o i n t s ( Points points , size\_t size , int \* cnt ) {

size\_t e = 20;

Points result = ( Points ) malloc ( e \* sizeof ( struct point ) ) ;

size\_t i n d e x T o A d d = 0;

for ( size\_t i = 0; i < size ; ++ i ) {

for ( size\_t j = 1; j < size ; ++ j ) {

int cnt = 0;

Points i n t e r s e c t s = g e t C i r c l e C i r c l e I n t e r s e c t i o n (& points [ i ] , &

points [ j ] , & cnt ) ;

if ( cnt > 0) {

for ( size\_t k = 0; k < cnt ; ++ k ) {

result [ i n d e x T o A d d ++] = i n t e r s e c t s [ k ];

}

}

}

}

return result ;

}

/\* \*

\* Selects only common points from the circles i n t e r s e c t i o n

\*

\* @param points I n t e r s e c t i o n points

\* @param size Size of that array \* @param beacons The given beacons and

their a c c u r a c i e s

\* @param cnt R ef er en ce to the r es ul ti ng array ’s count

\*

\* @return Array of points

\*/

Points g e t C o m m o n P o i n t s ( Points points , size\_t size , Points beacons , int \* cnt )

{

Points result = ( Points ) malloc ( sizeof ( struct point ) \* e n o u g h P o i n t s ) ;

59

int k = 0;

for ( size\_t i = 0; i < size ; ++ i ) {

if (! i s P o i n t B e l o n g T o A l l C i r c l e s ( points [ i ]. x , points [ i ]. y , beacons ) ) {

continue ;

}

result [ k ]. x = points [ i ]. x ;

result [ k ]. y = points [ i ]. y ;

result [ k ]. r = points [ i ]. r ;

++ k ;

}

\* cnt = k ;

return result ;

}

/\* \*

\* Mutates three arrays of c o o r d i n a t e s into one array of points

\*

\* @param xs xs

\* @param ys ys

* @param accs accs
* @return Array of points

\*/

Points c r e a t e P o i n t s ( double \* xs , double \* ys , double \* accs ) {

Points points = ( Points ) malloc ( M I N \_ B E A C O N S \* sizeof ( struct point ) ) ;

for ( size\_t i = 0; i < M I N \_ B E A C O N S ; ++ i ) {

points [ i ]. x = xs [ i ];

points [ i ]. y = ys [ i ];

points [ i ]. r = accs [ i ];

}

return points ;

}

/\* \*

\* Enlarges the beacons ’ a c c u r a c i e s p r o p o r t i o n a l l y

\*

\* @param beacons Array of beacons

\*/

void e n l a r g e A c c u r a c i e s ( Points beacons ) {

for ( size\_t i = 0; i < M I N \_ B E A C O N S ; ++ i ) { beacons [ i ]. r \*= (1 + step ) ;

}

}

60

// \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*

double \* c a l c u l a t e U s e r P o s i t i o n E p t a ( double \* xs , double \* ys , double \* accs ) {

double \* result = ( double \*) malloc ( sizeof ( double ) \* D I M E N S I O N S ) ;

Points beacons = c r e a t e P o i n t s ( xs , ys , accs ) ;

int i t e r a t i o n s = 0; while (1) {

if (++ i t e r a t i o n s > M A X \_ I T E R A T I O N S ) { return result ;

}

int i n t e r s e c t i o n C o u n t = 0;

Points i n t e r s e c t i o n s = g e t I n t e r s e c t i o n P o i n t s ( beacons , MIN\_BEACONS , &

i n t e r s e c t i o n C o u n t ) ;

if ( i n t e r s e c t i o n C o u n t == 0) { return result ;

}

int c o m m o n C o u n t = 0;

Points common = g e t C o m m o n P o i n t s ( intersections , intersectionCount , beacons , & c o m m o n C o u n t ) ;

if ( c o m m o n C o u n t == 2 || c o m m o n C o u n t == 3) {

Points center = g et Ce nt er ( common , c o m m o n C o u n t ) ; result [0] = center [0]. x ;

result [1] = center [0]. y ; return result ;

}

e n l a r g e A c c u r a c i e s ( beacons ) ;

}

return result ;

}

Приложение 4. Начальная настройка подключенной биб-

лиотеки

# import < B e a c o n L o c a t i o n / B e a c o n L o c a t i o n .h > @import C o r e G r a p h i c s ;

// ...

@p ro pe rt y ( nonatomic , strong ) B e a c o n L o c a t i o n \* library ;

61

// ...

@ i n t e r f a c e MyClass : NSObject < B e a c o n L o c a t i o n D e l e g a t e >

// ...

- ( void ) init {

self = [ super init ];

if ( self ) {

// init

\_library = [[ B e a c o n L o c a t i o n alloc ] i n i t W i t h U U I D S t r i n g : @ "

12345678 -1234 -0000 -4321 -876543210000 "

identifier :@"My region "];

// beacons

[ \_library . floor addBeaconWithMajor :0 minor :0 inPosition : CGPointMake (1,

2) ];

[ \_library . floor addBeaconWithMajor :0 minor :1 inPosition : CGPointMake (2,

3) ];

[ \_library . floor addBeaconWithMajor :0 minor :2 inPosition : CGPointMake (1,

3) ];

// method

* \_library . processor setAlgorithm : AlgorithmTypeSphereIntersection ];
  + delegate

\_library . delegate = self ;

}

return self ;

}

* ( void ) onUpdateUserPostion :( CGPoint ) position {

// do fancy stuff

}

62