Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Институт естественных и точных наук
Факультет математики, механики и компьютерных технологий
Кафедра прикладной математики и программирования
Направление подготовки Программная инженерия

РАБОТА ПРОВЕРЕНА
Рецензент, к.т.н. доцент кафедры

_____/
«____»_____2017 г.

ДОПУСТИТЬ К ЗАЩИТЕ
Заведующий кафедрой, д.ф.-м.н., доцент

_____/А.А.Замышляева
«____»_____2017 г.

Определение местоположения Android устройств в зонах отсутствия сигнала GPS

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ
ЮУрГУ–090404.2017.164.ПЗ ВКР

Руководитель работы, к.ф.-м.н., доцент
_____/С.М. Елсаков
«____»_____2017 г.
Автор работы
студент группы ЕТ-223
_____/ П.М. Порватов
«____»_____2017 г.
Нормоконтролер, к.э.н., доцент
_____/ Д.А. Дрозин
«____»_____2017 г.

Челябинск, 2017

АННОТАЦИЯ

Порватов П.М. Определение местоположения Android устройств в зонах отсутствия сигнала GPS.– Челябинск: ЮУрГУ, ЕТ-223, 41 с., 11 ил., 2 табл., библиогр. список – 50 наим., 3 прил.

В работе сделан обзор сервисов и программ для android устройств, которые определяют местоположения объекта. Приведены существенные функции программы геолокации. Составлены требования для программы и для android устройства. Разработана математическая модель определения местоположения. Приведен алгоритм фильтрации ошибок данных от датчиков (линейный фильтр Калмана). Математическая модель адаптирована для использования в фильтре Калмана. Сделана оценка эффективности работы фильтра Калмана. Разработана программа определения местоположения. Приведен пример работы разработанной программы.

Оглавление

ВВЕДЕНИЕ

Марка или модель автомобиля особо привлекают угонщиков. Самая распространенная причина этого вида краж – выгода от продажи автомобиля в разобранном виде. Около 80% всех хищений автотранспорта случается в ночное время на неохраняемых случайных стоянках. Оставляя машину у подъезда, владельцы ежевечерне рискуют.

Защита автомобиля состоит из заурядной звуковой сигнализации или слабой противоугонной системы. Если преступник знает как работает система защиты автомобиля, то благодаря определенным приспособлениям может за некоторое время взломать защиту автомобиля и угнать его.

Зачастую разработчики противоугонных систем предусматриваю возможность открыть машину без специального ключа и эта информация со временем попадает в руки преступников. В такой ситуации от угона уже ни чего не спасет.

Что же можно сделать, чтобы в случае угона автомобиля были хоть какие-то шансы его найти? Для этого существует множество GPS-трекеров и приложений для android коммуникаторов. Они отслеживают благодаря GPS и точкам связи (сотовая и Wi-Fi) координат автомобиля и передают эту информацию владельцу авто. Многие такие решения являются платными и не всегда доступны обычному пользователю авто.

Чтобы обезопасить себя от подобного характера преступлений иногда не достаточно использовать стандартные средства защиты, необходимо использовать комплекс средств защиты. Наша жизнь окружена множеством устройств, которыми можно воспользоваться для повышения уровня защиты. К таким устройствам можно отнести и android устройства.

В android устройства производители электроники интегрируют множество различных датчиков, таких как : GPS приемник, акселерометр, линейный акселерометр, гироскоп, магнитометр, барометр и множество других датчиков. Но не все датчики могут быть использованы для определения местоположения, ниже описаны некоторые датчики, которые можно будет использовать в инерционной системе навигации.

Акселерометр – это прибор, предназначенный для измерения кажущегося ускорения. Кажущееся ускорение – это разница между гравитационным и истинным ускорениями объекта.

Акселерометр, при изменении внутреннего состояния (то есть изменение ускорения мобильного устройства), сообщает мобильному устройству три значения: кажущееся ускорение по оси OX, кажущееся ускорение по оси OY, кажущееся ускорение по оси OZ.

Вычислять ускорение (дальнейшем будем использовать не просто ускорение, а линейное ускорение) из кажущегося ускорения нет необходимости, если внутри мобильного устройства интегрирован линейный акселерометр, который и предоставляем данные об линейном ускорении.

Гироскоп (трех осевой) – это прибор, измеряющий проекцию угловой скорости на три перпендикулярные друг другу оси.

Гироскоп, при изменении внутреннего состояния (то есть изменение углового ускорения мобильного устройства), сообщает мобильному устройству три значения: угловое ускорение по оси OX, угловое ускорение по оси OY, угловое ускорение по оси OZ.

Датчик магнитного поля (компас) или магнитометр – определяет текущие показания интенсивности внешнего магнитного поля. Датчик магнитного поля сообщает мобильному устройству три значения: значение магнитного поля по оси OX, значение магнитного поля по оси OY, значение магнитного поля по оси OZ.

GPS – глобальная система определения координат, предоставляет данные о времени, позволяет определить местоположение во всемирной системе координат WGS 84 в том случае, если доступен сигнал, который отправляют спутники системы навигации. Для получения сигнала необходим GPS приемник, интегрированный на android устройство.

Все перечисленные датчики мобильного устройства могут совместно предоставить возможность определить местоположение объекта в том случае, если по каким-то причинам пропал GPS сигнал. GPS приемник необходим для того, чтобы задать начальную точку местоположения, от которой, по данным от инерционных датчиков, можно будет вычислить следующую точку местоположения. Для передачи данных необходим беспроводной доступ в интернет. Все перечисленные датчики обладают погрешностью измерения данных.

Цель работы – разработать приложение, в котором будет реализован фильтр коррекции входных данных от ошибок и алгоритм определения местоположения по данным от инерционных датчиков и GPS.

Данное приложение позволит повысить вероятность вернуть угнанный автомобиль. Также можно отслеживать перемещение транспорта в местах, где GPS сигнал не проникает (туннели, крытые автостоянки) или GPS сигнал блокируется устройствами, предназначенные для глушения GPS сигнала. Данное приложение может пригодиться организациям, доход которых напрямую зависит от имеющегося автопарка.

Для достижения поставленной цели необходимо провести анализ программ определения местоположения, разработать математическую модель, реализовать фильтрацию ошибок данных от датчиков android устройства.

ГЛАВА 1. ГЕОЛОКАЦИЯ С УЧЕТОМ РЕАЛИЗАЦИИ ФУНКЦИЙ ПРОГРАММ

1.1. Обзор android приложений определения местоположения.

1.1.1. «ГдеМои»

ГдеМои – сервис геолокации подвижных объектов: автомобилей, людей, грузов и другого имущества. Сервис предоставляется компаниям и физическим лицам – по всей России, а также за ее пределами. Услуга основана на технологиях GPS / ГЛОНАСС / LBS, при этом используются как GPS-трекеры, так и мобильные приложения.

Местонахождение определяется по сигналам навигационных систем GPS/ГЛОНАСС с высочайшей точностью – до 5 метров.

Место отображается меткой на масштабируемой электронной карте. Вам также будут известны: ближайший адрес (город, улица, номер дома), скорость и направление движения, координаты и другая информация.

Когда сигналы спутниковой навигации недоступны, например, в метро или в глубине зданий, местонахождение определяется с меньшей точностью, по сигналам базовых станций GSM и WiFi-сетям.

Маршруты перемещения просматриваются за любой интересующий период в прошлом. Если в приложении выбран режим непрерывного слежения, на карте отражается подробный маршрут движения, места стоянок и события. Для каждой поездки указывается длина и продолжительность.

В режиме интервального слежения заряд аккумулятора расходуется более экономично, но история представляет собой не линию на карте, а последовательность пронумерованных меток. При подключении внешнего питания может автоматически включаться непрерывный режим.

Система мониторинга хранит подробную историю поездок вплоть до 3-х лет. В любой момент вы можете отобразить поездку на карте или построить отчет по передвижениям за интересующий период.

Для наглядности весь маршрут движения за выбранный период разбивается на отдельные поездки, разделенные стоянками. При этом для каждой поездки указывается:

- адрес, дата и время начала;
- адрес, дата и время окончания;
- общая длина и продолжительность.

Система позволяет отобразить список поездок за разные периоды или одновременно для нескольких выбранных объектов наблюдения (например, нескольких автомобилей). Выбор схемы раскрашивания отдельных поездок позволяет наглядно сравнить поездки между собой, не опасаясь запутаться. При необходимости можно отключить разделение на поездки, тогда система не будет учитывать стоянки.

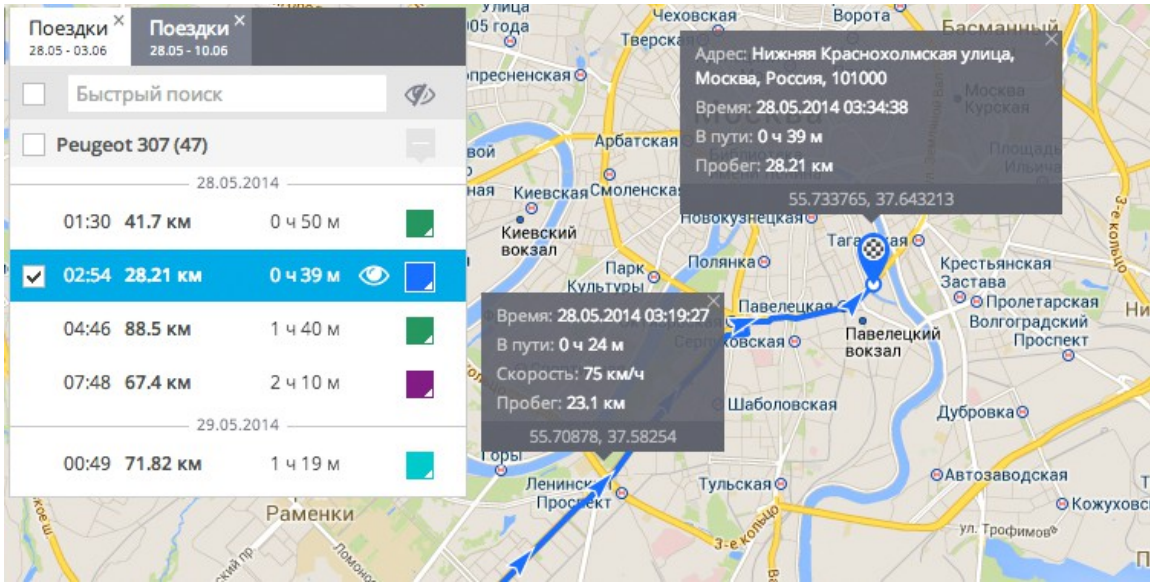

Рис. 1.1.Пример работы сервиса ГдеМои

Кликнув на любую точку поездки можно получить дополнительную информацию о параметрах движения на этом участке, таких как скорость, время и пробег с начала движения (рис. 1.1).

Система ГдеМои позволяет контролировать широкий спектр событий на основании местонахождения наблюдаемых объектов. Параметры этих событий и способы оповещения гибко настраиваются пользователем.

Типы регистрируемых событий можно разделить на две группы – те, которые не зависят от модели GPS-трекера, и те, которые определяются специфическими функциями устройства и/или схемой подключения. К первому типу относятся такие события как вхождение в гео-зону, ко второму, например, срабатывание датчика ДТП.

Типов поддерживаемых событий очень много. Но благодаря удобному помощнику создания контролирующих правил, их настраивать очень легко. Вот далеко не полный список событий, для примера:

- движение (вхождение в гео-зону, сход с маршрута, превышение скорости, начало стоянки, возобновление движения);
- безопасность (нажатие тревожной кнопки, сработка тревоги автосигнализации, отключение аккумулятора автомобиля, потеря GSM-связи, датчик ДТП, сенсор падения или удара, эвакуация);
- контроль техники (включение зажигания, спецмеханизмов, нажатие кнопок, выход температуры за установленные границы);
- качество вождения(резкие маневры, длительная стоянка с работающим двигателем);
- работа GPS-устройства (низкий заряд аккумулятора, отключение от внешнего питания, извлечение SIM-карты, отключение антенны);
- и многие другие...

В момент фиксации события система незамедлительно проинформирует пользователя – сообщением на экране, на телефон (SMS-уведомлением или автоматическим звонком) или Email. Можно указать любое число контактов.

Каждое сообщение содержит:

- наименование объекта;
- дату и время наступления события;
- настраиваемый пользователем текст оповещения;
- адрес места и его географические координаты;
- ссылку на карту;
- контроль с учетом места и времени.

В отношении любого типа регистрируемых событий можно ограничить место и время контроля. Для этого в правиле к событию уточняется гео-зона и расписание.

Вывод:

Программное обеспечение, в том числе и для операционных систем android, обладает множеством возможностей, но программное обеспечение является платным.

1.1.2. «GPS-TRACEORANGE»

Сервис «GPS-TRACE ORANGE» (рис. 1.2) предоставляет функцию геолокации подвижных объектов. Сервис позволяет отслеживать объект в реальном времени, история передвижений хранится в течение 30 дней, поэтому в любой момент можно отобразить нужный трек на карте. Можно отмечать интересные места и создавать геозоны (области, которые для тебя важны). Получай уведомления о превышении скорости, пересечении объектов геозоны, возможность отправлять через сервис сообщения и фотографии.Делиться своими треками и любимыми местами с друзьями в социальных сетях и открывай им доступ к онлайн наблюдению за твоим передвижением.

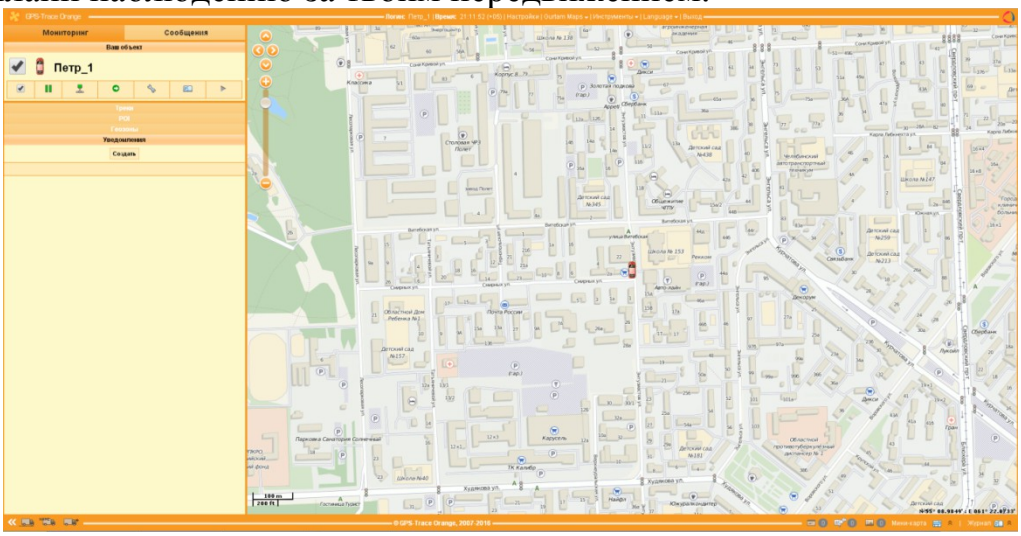

Рис. 1.2.Страница отслеживания на сервисе «GPS-TRACE ORANGE»

Для того, чтобы воспользоваться перечисленным функционалом выше нужно зарегистрироваться на сайте сервиса«GPS-TRACEORANGE» и скачать на android

– комуникатор приложение «GPSTagOrang», затем ввести логин и пароль, которые использовали для регистрации на сайте сервиса в приложении и можно пользоваться.

Стартовое окно приложеия изображено на рисунке 1.3.

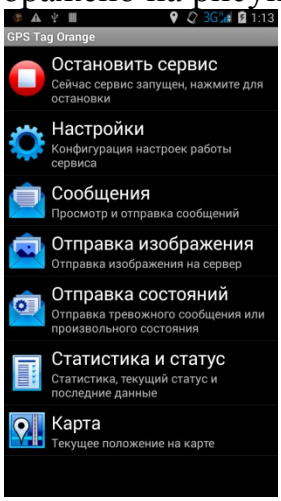

Рис. 1.3.Стартовое окно приложеия «GPSTagOrang»

Стартовое окно содержит семь кнопок:

- «Запустить/Остановить сервис»;
- «Настройки»;
- «Сообщения»;
- «Отправка изображения»;
- «Отправка состояний»;
- «Статистика и статусы»;
- «Карта».

Какие настройки позволяет приложение настраивать:

- работать постоянно или по таймауту (таймаут можно задать вручную от 5 и более минут);
- разрешить или запретить отправлять данные в роуминге;
- автозапуск сервиса (автоматически запускает серви после перезагрузки);
- работать при зарядке (запускать сервис при подключении зарядного устройства и останавливать при отключении);
- работа по расписанию;
- фильтрация сообщений (отправка сообщений через определенный интервал времени, преодоления минимального расстояния, изменения курса, перепада скорости движения, при превышении установленной погрешности определения местонахождения и превышении максимальной скорости);
- уведомления о событиях (сопровождение новых событий звуковым сигналом, вибрацией, светом);
- отправляемые сообщения могут включать в себя дополнительные параметры (имя провайдера, точность позиционных сообщений, уровень заряда батареи, подлючено ли устройство к источнику питания и состояние обьекта);

- возможность добавлять новые состояния обьекта;
- смена пользователя.

Настроек у приложения куда больше, чем было перечисленно выше, были выделены основные, которые имеют важность.

Особенно можно выделить просмотр текущего статуса приложения (рис. 1.4), который позволяет опеделить, из-за каких причин могут быть не получены геолокационные данные, отправлены сообщения или получены. Текущий статус программы также отображает информацию об обьекте.

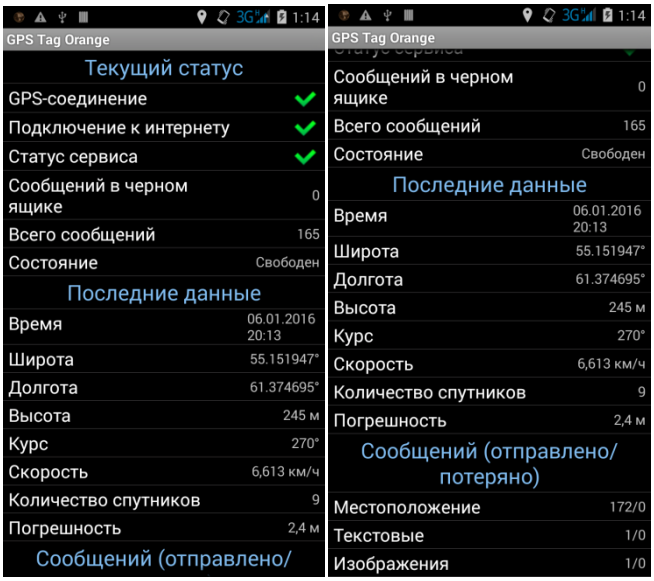

Рис. 1.4.Окно текущего статуса приложения «GPSTagOrang»

Сервис «GPS-TRACEORANGE» будетфункционировать только в том случае, если приложению «GPSTagOrang» есть доступ к интернету. Сервис не предусматнривает возможность следить за объектом с другого android-устройства.

Приложение «GPS Tag Orang» определяет геоданные при помощи GPS-спутников и беспроводных сетей. При этом настройки приложения позволяют выбрать источник местоположения: только GPS, только беспроводные сети, совместно GPS и беспроводные сети.

Выводы:

Следить за объектами придетя только с сайта сервиса «GPS-TRACEORANGE» и в случае падения сервиса, ни каких других запасных возможностей следить за обьектом нет.

Но сервис предусматривает оповещение через СМС или emailв случае появления определенных событий.

Точность определения местоположения беспроводными сетями прямо зависит от плотности застройки местности, где находится объект. Из-за множества помех на пути движения сигнала, ошибка может увеличиваться в несколько раз. И не все беспроводные сети (WiFi) могут определять местоположение устройства.

1.1.3. «WheresmyDroid»

Приложение «WheresmyDroid» можетпомочь найти телефон дома, в лесу и в других местах. Для этого нужно буде только отправить смс на номер телефона с кодовым словом и телефон начинает звенеть, вибрировать, независимо от выбранного в данный момент профиля и настроек громкости.

Полезной функцией программы «Wheres my Droid» является возможность через смс получить координаты своего телефона (Конечно, при условии включенного GPS). Для данной команды ключевое слово задается отдельно.

На рисунке 1.5 отображено главное окно приложения.

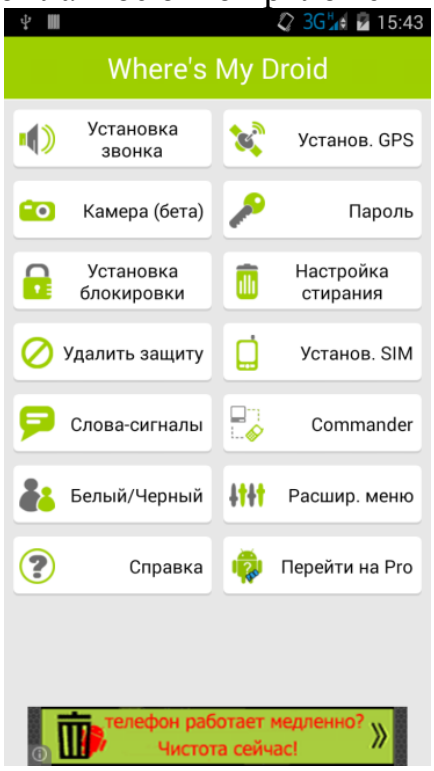

Рис. 1.5.Главное окно приложения«WheresmyDroid»

Данное бесплатное приложение имеет рекламу. И судя по всему программа после установки сразу работает.

Функции, доступные пользователю приложения:

- получение геолокации GPS;
- сообщение о геолокации при низком заряде батареи;
- звонок/вибрация телефона;
- текстовые сообщения для активации компонентов;
- активация компонентов с помощью веб-сайта Commander;
- защита с помощью кода доступа в целях предотвращения неавторизованных изменений приложения;
- уведомление о смене SIM-карты или номера телефона;
- режим невидимки скрывает входящие сообщения с помощью слова-сигнала;
- белый/черный списки для контроля над тем, кто может использовать приложение с помощью текстового сообщения.

Все перечисленные функции можно настраивать, но если приобрести профессиональную версию приложения, то добавятся следующие функции:

- возможность делать снимки с помощью камеры в устройстве;

- удаленная блокировка устройства;
- удаленное стирание данных с телефона и карты SD;
- предотвращение удаления;
- возможность скрытия ярлыка приложения.

Если скачали и установили бесплатную версию приложения «WheresmyDroid», то иногда будет появляться предложение о покупке профессиональной версии не выходя из приложения.

Наблюдение за объектом реализовано через сайт сервиса «WheresmyDroid» (рис. 1.6). Как можно видеть, сервис позволяет определить место расположения устройства на карте, отправить сообщение для обнаружения устройства по звуку и свету вспышки, при этом играет специфическая мелодия и мерцает вспышка (устройство находилось в беззвучном режиме). Существует возможность с сайта сделать фотографию и ее просмотреть, но так как я пользуюсь бесплатной версией приложения, данная функция у меня не работает. Также можно заблокировать телефон удаленно, но я не решился это проверять (а вдруг не получится разблокировать). И зачистить SD-карту или все устройство, дабы злоумышленник не получил доступ к конфиденциальной информации.

Рис. 1.6. страница сайта управления сервисом «WheresmyDroid»

Обмен данными происходил сайта сервиса и приложения через интернет. Также отправляя смс сообщения с кодовым словом с другого аппарата, приложение выполняет соответствующую кодовому слову функцию.

Выводы:

Можно через смс получить координаты устройства, но нет приложения, которое могло бы показать расположение этого устройства на карте, следить же за объектом возможно только через сайт сервиса.

Отсутствует определение местоположения устройства без GPSсигнала.

1.2.Выявление существенных функций приложения «android tracker».

Функций приложения «android tracker»:

- получение геолокационных данных с GPS и рядом находящихся вышек сотовой связи и Wi-Fi точек;
- Вычисление геолокационных данных по данным от датчиков устройства;
- обмен сообщениями через интернет.

1.3. Обоснование выбора средств разработки и программных средств для реализации приложения.

1.3.1.Язык программирования JAVA

Java – это объектно-ориентированный язык программирования. Программы на Java транслируются в байт-код, выполняемый виртуальной машиной Java, которая обрабатывает байтовый код и передает инструкции оборудованию как интерпретатор. Достоинство подобного способа выполнения программ заключается в полной независимости байт-кода от операционной системы и оборудования, что позволяет выполнять Java-приложения на любом устройстве, для которого существует соответствующая виртуальная машина. Другой важной особенностью Java является гибкая система безопасности благодаря тому, что исполнение программы полностью контролируется виртуальной машиной. Любые операции, которые превышают установленные полномочия программы (например, попытка несанкционированного доступа к данным или соединения с другим компьютером) вызывают немедленное прерывание.

1.3.2. СредаразработкиAndroidStudio

В основе Android Studio лежит интеллектуальный редактор исходного кода, предлагающий такие возможности, как расширенное автозавершение кода, его реструктуризация и анализ.

Android Software Development Kit (SDK) содержит множество инструментов и утилит для создания и тестирования приложений.

С помощью SDK Manager можно установить Android API любой версии, а также проверить репозиторий на наличие доступных, но еще не установленных пакетов и архивов.

AndroidStudio теперь является официальной средой разработки для Android.

1.4. Выводы

Сервис геолокации подвижных объектов "ГдеМои" обладает множеством функций, большая часть из которых относится к реализации серверной части

данного сервиса. Те функции, которые можно отнести к клиентской части сервиса "ГдеМои", это определение местоположение через GPS / ГЛОНАСС с высокой точности и, если нет сигнала GPS / ГЛОНАСС, то определение местоположение от базовых станций GSM и WiFi-сетей, точность определения местоположения от базовых станций GSM и WiFi-сетей зависит от количества помех, стоящих на пути прохождения сигнала, погрешность может определяться от нескольких метров, до нескольких сотен метров. Что не так хорошо, если объект находится в городе с плотной застройкой территории высокими зданиями. Не все WiFi-сети имеют возможность определять местоположения. Так как многие WiFi-модемы не имеют возможности привязки к определенному месту положению.

Технологии определения местоположения по GPS и ГЛОНАСС похожи. Не все android устройства имеют возможность определять местоположение по ГЛОНАСС, так как данная технология появилась намного позже, чем GPS и многие android устройства не были укомплектованы ГЛОНАСС приемниками.

У сервиса "ГдеМои" отсутствует возможность определения местоположения по данным от инерционных датчиков (акселерометр, гироскоп и т.д.), которые могут увеличить точность или, если не может определить местоположение от GPS / ГЛОНАСС и базовых станций GSM или WiFi-сетей (если объект движется, а сигнал GSM слабый, то определение местоположения по GSM маловероятно), определить местоположение с точностью до некоторой погрешности.

Сервис «GPS-TRACE ORANGE» предлагает множество функций, которые реализованы на серверной стороне данного сервиса, которые также имеются у сервиса "ГдеМои". Определение местоположения происходит при помощи GPS и беспроводных сетей. Но отсутсвует определение местоположения или уточнение его по данным от инерционных датчиков android устройства.

Сервис «WheresmyDroid» поможет отследить android устройство. Приложение для android устройства может определять местоположение только по данным от GPS сигнала. Возможность делать снимки и отправлять их также повышает возможность поймать злоумышлиника или определить по фотографии местоположение. Но у приложения отсутсвует определение местоположения или уточнение его по данным от инерционных датчиков android устройства.

Реализация того множества функций, которые предоставляют сервисы "ГдеМои", «GPS-TRACE ORANGE», «WheresmyDroid» нет необходимости, так как можно установить бесплатные приложения от «GPS-TRACE ORANGE» и «WheresmyDroid» и указанные функции появятся, но также можно увеличить шанс определения местоположения, если разработать на android устройство приложение, которое сможет вычислить геолокацию по данным от инерционных датчиков android устройства.

ГЛАВА 2. АЛГОРИТМ ОПРЕДЕЛЕНИЯ МЕСТОПОЛОЖЕНИЯ ANDROID УСТРОЙСТВА

2.1.Требования

Существует множество разных датчиков, которые интегрированы в android устройство производителем электроники. Их задача упростить или улучшить жизнь пользователя мобильного устройства.

Магнитометр сообщает текущие показания интенсивности внешнего магнитного поля, а акселерометр измеряет кажущееся ускорение объекта. Связка датчиков магнитометр и акселерометр могут определить ориентацию устройства в пространстве. Магнитометр является самым не точным датчиком, наша жизнь окружена железными предметами и магнитными полями электрических изделий, которые вносят ошибку в измеренные данные магнитометром. Из-за этого решено магнитометр не использовать.

Кажущееся ускорение, которое сообщает акселерометр, содержит влияние гравитации, тогда придется компенсировать значения гравитации из данных акселерометра и получим линейное ускорение. Чтобы не вносить ошибки, которые появятся при вычислении линейного ускорения из кажущегося ускорения, будет использовать линейный акселерометр для определения пройденного пути.

Угол поворота можно определить как по данным от линейного акселерометра, так и по данным от гироскопа. Данные от линейного акселерометра более зашумлены по сравнению с данными от гироскопа. Для определения угла поворота будет использоваться гироскоп.

На данный момент самой распространенной операционной системой android является Lollipop (см. таблицу 2.1).

Таблица 2.1

Версия операционной системы android	Кодовое имя операционной системы android	Распределение операционной системы android относительно общего числа экземпляров, %
2.2	Froyo	0,1
2.3.3-2.3.7	Gingerbread	1,2
4.0.3-4.0.4	Ice Cream Sandwich	1,2
4.1.x-4.3	Jelly Bean	12,8
4.4	KitKat	24,0
5.0-5.1	Lollipop	34,0
6.0	Marshmallow	26,3
7.0	Nougat	0,4

Задача:

Необходимо разработать приложение (программу) для android смартфона, которое определяет местоположение мобильного устройства в зонах, где нет

сигнала GPS и отсутствует беспроводной интернет, используя интегрированные датчики в android смартфоне: GPS, линейный акселерометр, гироскоп.

Требования к приложению (программе):

Приложение должно получать данные от GPS, линейного акселерометра и гироскопа.

Приложение должно определять начальное местоположение по GPS и последующее местоположение по данным от линейного акселерометра и гироскопа.

В приложении должен быть реализован алгоритм фильтрации данных от ошибок измерения

Приложение должно отправлять данные о местоположении устройства на сервер.

Требования к устройству:

Для работы приложения (программы) необходимо иметь android смартфона с версией операционной системы android 5.0-5.1 и интегрированные на устройство датчиками: GPS, линейный акселерометр, гироскоп.

2.2.Математическая модель определения местоположения

Для нахождения положения мобильного устройства будет использован метод инерциальной навигации. Физические принципы инерциальной навигации неразрывно связаны с решением основной задачи динамики: когда на тело действует сила, с помощью параметров и существующих данных нужно определить местоположение тела в определенный момент.

При внешнем воздействии на мобильное устройство с датчиков линейного акселерометра и гироскопа будут поступать данные, которые характеризуют это воздействие относительно локальной системы координат. Локальная система координат зафиксирована относительно мобильного устройства и является прямоугольной декартовой системы координат XYZ с углами поворота (крен, тангаж и азимут) вокруг осей OX, OY и OZ соответственно по часовой стрелке (рис.2.4) относительно подобной глобальной прямоугольной декартовой системы координат, цент которой находится на нулевой широте (экватор) и нулевой долготе (проходит через Гринвичскую лабораторию) Земли, где ось X направлена на север, ось Y на восток, а ось Z вверх (рис.2.5).

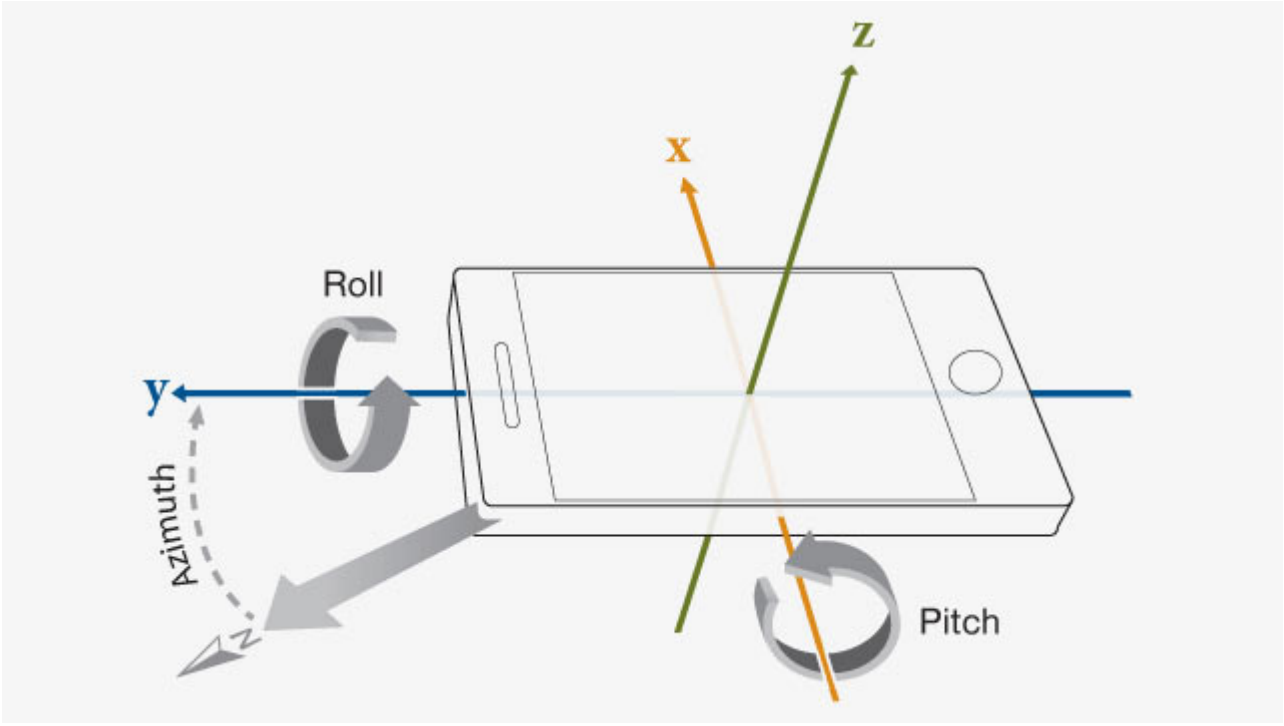

Рис. 2.4. Локальная система координат мобильного устройства

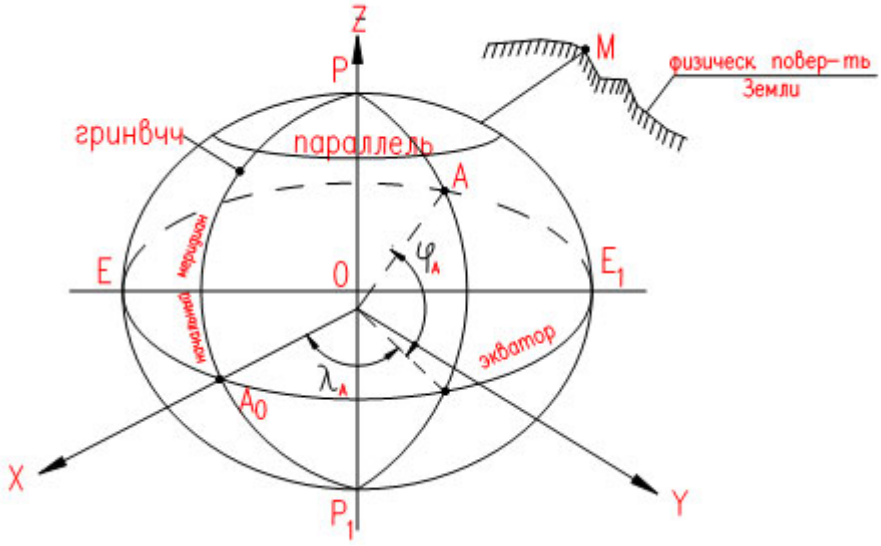

Рис. 2.5. Глобальная система координат

Данные от линейного акселерометра сообщают только об ускорении объекта вдоль осей OX, OY и OZ мобильного устройства, чтобы узнать его скорость, необходимо полученное ускорение умножить на период времени, за которое происходит ускорение объекта. Так как в определенный момент времени узнать скорость объекта невозможно, то необходимо будет интегрировать ускорение по промежутку времени, чтобы получить скорость:

$$V_{x,k} = V_{x,k-1} + a_{x,k} \cdot dt + \xi_{1,k} \quad , \tag{1}$$

$$V_{y,k} = V_{y,k-1} + a_{y,k} \cdot dt + \xi_{2,k} \quad , \tag{2}$$

$$V_{z,k} = V_{z,k-1} + a_{z,k} \cdot dt + \xi_{3,k} \quad , \tag{3}$$

где $V_{x,k}, V_{y,k}, V_{z,k}$ – скорость устройства м/с в k-й момент времени по осям OX, OY, OZ;

$V_{x,k-1}, V_{y,k-1}, V_{z,k-1}$ – скорость устройства м/с в k-1-й момент времени по осям OX, OY, OZ;

$a_{x,k}, a_{y,k}, a_{z,k}$ – ускорение устройства м/с^2 в k-й момент времени по осям OX, OY, OZ;

dt – время, за которое происходит ускорение объекта или разница между k-м моментом времени и k-1-м;

$\xi_{1,k}, \xi_{2,k}, \xi_{3,k}$ – ошибки скорости устройства м/с в k-й момент времени по осям OX, OY, OZ.

Аналогично будет вычислен пройденный путь объектом, необходимо будет интегрировать скорость по промежутку времени:

$$S_{x,k} = S_{x,k-1} + V_{x,k} \cdot dt + \xi_{4,k} \quad , \tag{4}$$

$$S_{y,k} = S_{y,k-1} + V_{y,k} \cdot dt + \xi_{5,k} \quad , \tag{5}$$

$$S_{z,k} = S_{z,k-1} + V_{z,k} \cdot dt + \xi_{6,k} \quad , \tag{6}$$

где $S_{x,k}, S_{y,k}, S_{z,k}$ – расстояние до точки относительно центра координат по осям OX, OY, OZ в k-й момент времени;

$S_{x,k-1}, S_{y,k-1}, S_{z,k-1}$ – расстояние до точки относительно центра координат по осям OX, OY, OZ в k-1-й момент времени;

$\xi_{4,k}, \xi_{5,k}, \xi_{6,k}$ – ошибки расстояния в k-й момент времени по осям OX, OY, OZ в метрах.

Данные от гироскопа сообщают угловую скорость устройства вокруг осей OX, OY и OZ. Для того чтобы получить угол поворота по данным от гироскопа, необходимо интегрировать угловую скорость по промежутку времени:

$$\alpha_k = \alpha_{k-1} + \omega_{x,k} \cdot dt + \xi_{7,k} \quad , \tag{7}$$

$$\beta_k = \beta_{k-1} + \omega_{y,k} \cdot dt + \xi_{8,k} \quad , \tag{8}$$

$$\gamma_k = \gamma_{k-1} + \omega_{z,k} \cdot dt + \xi_{9,k} \quad , \tag{9}$$

где $\omega_{x,k}, \omega_{y,k}, \omega_{z,k}$ – угловая скорость устройства в k-й момент времени вокруг осей OX, OY, OZ в рад/с;

$\alpha_{k-1}, \beta_{k-1}, \gamma_{k-1}$ – углы поворота (соответственно крен, тангаж и азимут) вокруг осей OX, OY, OZ в радианах в k-1-й момент времени;

$\alpha_k, \beta_k, \gamma_k$ – углы поворота вокруг осей OX, OY, OZ в градусах в k-й момент времени;

$\xi_{7,k}, \xi_{8,k}, \xi_{9,k}$ – ошибки углов поворота вокруг осей OX, OY, OZ в радианах в k-й момент времени.

Формулы (4, 5, 6) определяют местоположение мобильного устройства относительно локальной системы координат. Чтобы определить местоположение мобильного устройства относительно глобальной системы координат,

необходимо будет учесть углы поворота устройства, которые можно вычислить, используя формулы (7, 8, 9). Для преобразования пройденного пути мобильного устройства, из локальной системы координат в глобальную систему координат, будем использовать матрицы поворота вокруг осей OX, OY и OZ с отрицательными углами поворота $\alpha_k, \beta_k, \gamma_k$:

$$R_x = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(-\alpha_k) & -\sin(-\alpha_k) \\ 0 & \sin(-\alpha_k) & \cos(-\alpha_k) \end{pmatrix}, \quad (10)$$

$$R_y = \begin{pmatrix} \cos(-\beta_k) & 0 & -\sin(-\beta_k) \\ 0 & 1 & 0 \\ \sin(-\beta_k) & 0 & \cos(-\beta_k) \end{pmatrix}, \quad (11)$$

$$R_z = \begin{pmatrix} \cos(-\gamma_k) & -\sin(-\gamma_k) & 0 \\ \sin(-\gamma_k) & \cos(-\gamma_k) & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad (12)$$

$$\hat{\bar{S}} = R_x \cdot R_y \cdot R_z \cdot \bar{S}, \quad (13)$$

где R_x, R_y, R_z – матрицы поворота вокруг осей OX, OY и OZ;

$\bar{S}$ – вектор пройденного пути, который состоит из $S_{x,k}, S_{y,k}, S_{z,k}$ в локальной системе координат мобильного устройства;

$\hat{\bar{S}}$ – вектор пройденного пути, который состоит из $\hat{\bar{S}}_{x,k}, \hat{\bar{S}}_{y,k}, \hat{\bar{S}}_{z,k}$ в глобальной системе координат.

Данные, которые поступают от GPS, определяют местоположение мобильного устройства во всемирной системе координат WGS 84. Чтобы перейти из всемирной системе координат WGS 84 в глобальную систему координат, будем использовать следующие преобразования:

$$lat_k = \frac{\hat{\bar{S}}_{x,k}}{111197} + \xi_{10,k}, \quad (14)$$

$$lon_k = \frac{\hat{\bar{S}}_{y,k}}{111197 * cos\left(\frac{\hat{\bar{S}}_{x,k}}{111197}\right)} + \xi_{11,k}, \quad (15)$$

$$alt_k = \hat{\bar{S}}_{z,k} + \xi_{12,k}, \quad (16)$$

где lat_k, lon_k, alt_k – широта, долгота и высота устройства, полученные от GPS в k-й момент времени;

$\xi_{10,k}$, $\xi_{11,k}, \xi_{12,k}$ – ошибки широты, долготы и высоты, которые можно получить от GPS в k-й момент времени.

Также от GPS можно получить азимут угла поворота в глобальной системе координат:

$$A z_k = \gamma_k + \xi_{13,k} \quad , \tag{17}$$

где $A z_k$ – азимут угла поворота, полученный от датчика GPS в к-й момент времени в радианах;

γ_k – азимут, подсчитанный по данным от гироскопа за k периодов времени в радианах;

$\xi_{13,k}$ – ошибка азимута, полученного от датчика GPS в k-й момент времени в радианах.

Широта, долгота и высота, полученные от датчика GPS и переведенные в глобальную систему координат, будут считаться начальной точкой отсчета для вычисления следующей точки по данным от линейного акселерометра. В этот же момент азимут, полученный от датчика GPS, будет считаться начальным углом для дальнейшего подсчета угла поворота.

2.3.Описание линейного фильтра Калмана

Нашу жизнь окружает множество данных, которые необходимо каким либо способом оценить и отфильтровать. Фильтр Калмана не является самым простым фильтром данных, существует множество других фильтров, которые являются более простыми в реализации, но также менее эффективными. Суть фильтра Калмана заключается в том, что он рекурсивно способен оценивать вектор состояния динамической системы, опираясь на зашумленные измерения.

Допустим имеем динамическую систему, эволюцию которой можно записать в виде системы линейных уравнений, которая будет принимать множество входных параметров и возвращать тоже множество параметров, но только предсказанных. Такое множество называют вектором состояния и обозначают символом x. Систему линейных уравнений необходимо записать в виде матрицы F. Предположим , что вектор состояния системы x имеет вектор ошибки W, который характеризуется матрицей ковариации случайного воздействия. Также может существовать управляющее воздействие, которое состоит из матрицы управления B и вектора управляющего воздействия u. Данные привязаны к определенному моменту времени, то у векторов x, u и матрицы W появится нижний индекс, характеризующий момент времени. Тогда процесс предсказания состояния системы можно записать выражением:

$$x_k = F \cdot x_{k-1} + B \cdot u_{k-1} + W_{k-1} \quad , \tag{18}$$

где x_k – предсказанный вектор состояния в момент времени k;

x_{k-1} – вектор состояния в момент времени k−1;

F – матрица эволюции физического процесса, которая воздействует на вектор состояния x_{k-1} ;

u_{k-1} – вектор управляющего воздействия в момент времени k−1;

B – матрица управления, которая применяется к вектору управляющих воздействий u_{k-1} ;

W_{k-1} – вектор ошибки вектора состояния в момент времени k−1, который характеризуется матрицей ковариации случайного воздействия.

Если отсутствует управляющее воздействие на физический процесс, то формула будет упрощена:

$$x_k = F \cdot x_{k-1} + W_{k-1} \quad . \tag{19}$$

Вектор W_{k-1} вычисляется из ковариационной матрицы Q и константы σ , являющейся среднеквадратичной ошибкой покоя (белый шум):

$$W_{k-1} = \sigma \cdot Q \tag{20}$$

Необходимо определить ковариационную матрицу ошибки для физической модели. Например ошибка ускорения пропорциональна промежутку времени dt :

$$w_a = \sigma_a \cdot dt^2 \tag{21}$$

где:

w_a – ячейка матрицы W_{k-1} , находящаяся по диагонали матрицы ковариации и соответствующей ускорению a_{k-1} устройства;

σ_a – среднеквадратичная ошибка ускорения.

Для нахождения среднеквадратичной ошибки покоя необходимо будет произвести замеры параметров, характеризующие данную физическую модель (ускорение акселерометра, угловое ускорение гироскопа) и вычислить по следующей формуле:

$$\sigma = \frac{\sum_{i=1}^{n} \left(y_i - \acute{y} \right)^2}{n} \quad , \tag{22}$$

где y_i – i-ое измеренное значение;

$\acute{y}$ – среднеарифметическое значение за n измерений;

n – количество измерений.

На этапе коррекции по данным измерения (полученного с некоторой погрешностью), результат экстраполяции уточняется. Благодаря пошаговой природе алгоритма, он может в реальном времени отслеживать состояние объекта (без заглядывания вперед, используя только текущие замеры и информацию о предыдущем состоянии и его неопределенности).

$$z_k = H \cdot x_k + V_k \quad , \tag{23}$$

где z_k – вектор измерений, который состоит из входных данных в момент времени k;

H – матрица, определяющая отношение между вектором с измерениями z_k и вектором состояния x_k системы;

V_k – вектор ошибки, характеризующий ошибку измерений.

Также для выполнения фильтра Калмана потребуется матрица R, которая является матрицей ошибки измерения, она может быть определена испытанием измерительных приборов и определение погрешности их измерения.

Весь алгоритм фильтра Калмана будет выглядеть следующим образом:

1) Предсказание состояния системы:

$$x_k = F \cdot x_{k-1} \quad . \tag{24}$$

2) Предсказание ошибки ковариации:

$$P_k = F \cdot P_{k-1} \cdot F^T + Q \quad , \tag{25}$$

где P_{k-1} – предсказание ошибки в прошлый момент времени;

P_k – предсказание ошибки.

3) Вычисление усиления Калмана:

$$K_k = P_k \cdot H^T \cdot \left(H \cdot P_k \cdot H^T + R \right)^{-1} \quad , \tag{26}$$

где K_k – усиление Калмана.

4) Обновление оценки вектора состояния с учетом измерения z_k :

$$\widehat{x_k} = x_k + K_k \cdot (z_k - H \cdot x_k) \quad , \tag{27}$$

где $\widehat{x_k}$ – обновленный вектор состояния системы в момент времени k, скорректированное с учетом ошибки между x_k и z_k .

5) Обновление ошибки ковариации $\widehat{P_k}$:

$$\widehat{P_k} = (I - K_k \cdot H) \cdot P_k \quad , \tag{28}$$

где I – матрица инцидентности.

2.4.Адаптирование математической модели под структуру линейного фильтра Калмана

На этапе экстраполяции в фильтре Калмана матрица F будет описывать эволюцию вектора состояния x определения местоположения в глобальной системе координат.

Вектор состояния x математической модели будет состоять из:

S_y, S_x, S_z – местоположение устройства в глобальной системе координат в метрах;

V_x, V_y, V_z – скорость устройства м/с в k-й момент времени по осям OX, OY, OZ в глобальной системе координат;

a_x, a_y, a_z – ускорение устройства м/с^2 по осям OX, OY, OZ в глобальной системе координат;

α, β, γ – углы поворота вокруг осей OX, OY, OZ в радианах в глобальной системе координат;

$\omega_x, \omega_y, \omega_z$ – угловая скорость устройства вокруг осей OX, OY, OZ в рад/с радианах в глобальной системе координат.

Вектор состояния x математической модели будет выглядеть следующим образом:

$$x=(S_y,S_x,S_z,V_x,V_y,V_z,a_x,a_y,a_z,\alpha,\beta,\gamma,\omega_x,\omega_y,\omega_z)^T \quad . \tag{29}$$

Для $S_{y,k},S_{x,k},S_{z,k}$ выражения должны выглядеть следующим образом:

$$S_{x,k}=S_{x,k-1}+V_{x,k-1}\cdot dt+a_{x,k-1}\cdot\frac{dt^2}{2} \quad , \tag{30}$$

$$S_{y,k}=S_{y,k-1}+V_{y,k-1}\cdot dt+a_{y,k-1}\cdot\frac{dt^2}{2} \quad , \tag{31}$$

$$S_{z,k}=S_{z,k-1}+V_{z,k-1}\cdot dt+a_{z,k-1}\cdot\frac{dt^2}{2} \quad , \tag{32}$$

где $S_{y,k-1},S_{x,k-1},S_{z,k-1}$ – местоположение устройства в глобальной системе координат в метрах в k-1-й момент времени;

$V_{x,k-1},V_{y,k-1},V_{z,k-1}$ – скорость устройства м/с в k-1-й момент времени по осям OX, OY, OZ в глобальной системе координат;

$a_{x,k-1},a_{y,k-1},a_{z,k-1}$ – ускорение устройства м/с^2 по осям OX, OY, OZ в глобальной системе координат в k-1-й момент времени.

Но так как выражения $a_{x,k-1}\cdot\frac{dt^2}{2}$, $a_{y,k-1}\cdot\frac{dt^2}{2}$ и $a_{z,k-1}\cdot\frac{dt^2}{2}$ принимают слишком маленькое значение, то решено их убрать из выражений (32, 33, 34).

Этап экстраполяции в фильтре Калмана будет выглядеть следующим образом:

$$\begin{aligned}
S_{x,k}&=S_{x,k-1}+V_{x,k-1}\cdot dt+\xi_{1,k-1}\\
S_{y,k}&=S_{y,k-1}+V_{y,k-1}\cdot dt+\xi_{2,k-1}\\
S_{z,k}&=S_{z,k-1}+V_{z,k-1}\cdot dt+\xi_{3,k-1}\\
V_{x,k}&=V_{x,k-1}+a_{x,k-1}\cdot dt+\xi_{4,k-1}\\
V_{y,k}&=V_{y,k-1}+a_{y,k-1}\cdot dt+\xi_{5,k-1}\\
V_{z,k}&=V_{z,k-1}+a_{z,k-1}\cdot dt+\xi_{6,k-1}\\
a_{x,k}&=a_{x,k-1}+\xi_{7,k-1}\\
a_{y,k}&=a_{y,k-1}+\xi_{8,k-1}\\
a_{z,k}&=a_{z,k-1}+\xi_{9,k-1}\\
\alpha_k&=\alpha_{k-1}+\omega_{x,k-1}\cdot dt+\xi_{10,k-1}\\
\beta_k&=\beta_{k-1}+\omega_{y,k-1}\cdot dt+\xi_{11,k-1}\\
\gamma_k&=\gamma_{k-1}+\omega_{z,k-1}\cdot dt+\xi_{12,k-1}\\
\omega_{x,k}&=\omega_{x,k-1}+\xi_{13,k-1}\\
\omega_{y,k}&=\omega_{y,k-1}+\xi_{14,k-1}\\
\omega_{z,k}&=\omega_{x,k-1}+\xi_{15,k-1}
\end{aligned} \quad , \tag{33}$$

где $\xi_{1,k-1},\xi_{2,k-1},\xi_{3,k-1}$ – ошибки предсказания координат устройства соответственно для осей OX, OY и OZ в метрах в k-1-й момент времени;

$\xi_{4,k-1}, \xi_{5,k-1}, \xi_{6,k-1}$ – ошибки предсказания скоростей устройства для осей OX, OY и OZ в м/с в k-1-й момент времени;

$\xi_{7,k-1}, \xi_{8,k-1}, \xi_{9,k-1}$ – ошибки предсказания ускорения устройства для осей OX, OY и OZ в м/с^2 в k-1-й момент времени;

$\xi_{10,k-1}, \xi_{11,k-1}, \xi_{12,k-1}$ – ошибки предсказания углов направления движения устройства для осей OX, OY и OZ в радианах в k-1-й момент времени;

$\xi_{13,k-1}, \xi_{14,k-1}, \xi_{15,k-1}$ – ошибки предсказания углового ускорения устройства соответственно для осей OX, OY и OZ в рад/с в k-1-й момент времени;

$S_{y,k}, S_{x,k}, S_{z,k}, V_{x,k}, V_{y,k}, V_{z,k}, a_{x,k}, a_{y,k}, a_{z,k}, \alpha_k, \beta_k, \gamma_k, \omega_{x,k}, \omega_{y,k}, \omega_{z,k}$ – значения вектора состояния x_k в k-й момент времени;

$S_{y,k-1}, S_{x,k-1}, S_{z,k-1}, V_{x,k-1}, V_{y,k-1}, V_{z,k-1}, a_{x,k-1}, a_{y,k-1}, a_{z,k-1}, \alpha_{k-1}, \beta_{k-1}, \gamma_{k-1}, \omega_{x,k-1}, \omega_{y,k-1}, \omega_{z,k-1}$ – значения вектора состояния x_{k-1} в k-1-й момент времени.

Выражение предсказания состояния $x_k = F \cdot x_{k-1}$ будет выглядеть:

$$\begin{matrix} S_y \\ S_x \\ S_z \\ V_x \\ V_y \\ V_z \\ a_x \\ a_y \\ a_z \\ \alpha \\ \beta \\ \gamma \\ \omega_x \\ \omega_y \\ \omega_z \end{matrix} = \begin{pmatrix} 1 & 0 & 0 & dt & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & dt & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & dt & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & dt & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & dt & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & dt & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & dt & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & dt & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & dt \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{matrix} S_y \\ S_x \\ S_z \\ V_x \\ V_y \\ V_z \\ a_x \\ a_y \\ a_z \\ \alpha \\ \beta \\ \gamma \\ \omega_x \\ \omega_y \\ \omega_z \end{matrix}, \quad (34)$$

где dt – это промежуток времени между приходящими данными от датчиков GPS, линейного акселерометра и гироскопа.

Необходимо определить ковариационную матрицу ошибки Q для физической модели F. Так ошибка ускорения пропорциональна промежутку времени dt. Тогда матрица ковариации будет выглядеть следующим образом:

$$q_a = \begin{pmatrix} \sigma_{a,x} \cdot dt^2 & 0 & 0 \\ 0 & \sigma_{a,y} \cdot dt^2 & 0 \\ 0 & 0 & \sigma_{a,z} \cdot dt^2 \end{pmatrix}, \quad (35)$$

$$q_g = \begin{pmatrix} \sigma_{g,x} \cdot dt^2 & 0 & 0 \\ 0 & \sigma_{g,y} \cdot dt^2 & 0 \\ 0 & 0 & \sigma_{g,z} \cdot dt^2 \end{pmatrix}, \quad (36)$$

$$O = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad (37)$$

$$Q = \begin{pmatrix} q_a & O & O & O & O \\ O & q_a & O & O & O \\ O & O & q_a & O & O \\ O & O & O & q_g & O \\ O & O & O & O & q_g \end{pmatrix}, \quad (38)$$

где $\sigma_{a,x}, \sigma_{a,y}, \sigma_{a,z}$ – среднеквадратичная ошибка линейного ускорения, полученного от линейного акселерометра в состоянии покоя;

$\sigma_{g,x}, \sigma_{g,y}, \sigma_{g,z}$ – среднеквадратичная ошибка углового ускорения, полученного от гироскопа в состоянии покоя;

O – нулевая матрица размером 3x3.

Так как данные от датчиков приходят в разное время, то в данном фильтре Калмана будет три матрицы наблюдения: матрица наблюдения для GPS, матрица наблюдения для гироскопа и матрица наблюдения для акселерометра. По аналогии с матрицами наблюдения появятся три входных вектора Z и три матрицы ковариации входных данных R.

Этап коррекции в фильтре Калмана по данным от GPS будет выглядеть следующим образом:

$$\begin{cases} lat_k = \dfrac{S_{x,k}}{111197} + \xi_{16,k} \\ lon_k = \dfrac{S_{y,k}}{111197 \cdot cos\left(\dfrac{S_{x,k}}{111197}\right)} + \xi_{17,k} \\ alt_k = S_{z,k} + \xi_{18,k} \\ ¿Az_k = \gamma_k + \xi_{19,k} \end{cases}, \quad (39)$$

где lat_k, lon_k, alt_k, Az_k – широта, долгота, высота и азимут которые являются входные данные от GPS в k-й момент времени;

$S_{x,k}, S_{y,k}, S_{z,k}$ – координаты местоположения устройства в k-й момент времени в метрах, взятые из вектора состояния x_k;

γ_k – азимут угла поворота устройства в глобальной системе координат в радианах;

$\xi_{16,k}, \xi_{17,k}, \xi_{18,k}, \xi_{19,k}$ – ошибки измерения входных данных от GPS в k-й момент времени в метрах.

Матрица наблюдения H для GPS:

$$H=\begin{pmatrix} \frac{1}{111197} & 0 & 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0 & \frac{1}{111197*cos\left(\frac{S_x}{111197}\right)} & 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0 & 0 & 1&0&0&0&0&0&0&0&0&0&0&0 \\ 0 & 0 & 0&0&0&0&0&0&0&0&1&0&0&0 \end{pmatrix}, \quad (40)$$

где S_x – местоположение устройства по оси OX в глобальной системе координат в метрах, взятое из вектора состояния x.

Вектор измерений для GPS:

$$Z=(lat, lon, alt, Az)^T, \quad (41)$$

где lat, lon, alt, Az – широта, долгота, высота и азимут устройства, полученные от GPS на данный момент.

$$R=\begin{pmatrix} \delta_{lat}\cdot dt^2 & 0 & 0 & 0 \\ 0 & \delta_{lon}\cdot dt^2 & 0 & 0 \\ 0 & 0 & \delta_{alt}\cdot dt^2 & 0 \\ 0 & 0 & 0 & \delta_{Az}\cdot dt^2 \end{pmatrix}, \quad (42)$$

где $\delta_{lat}, \delta_{lon}, \delta_{alt}, \delta_{Az}$ – среднеквадратичная ошибка данных lat, lon, alt, Az, которые можно получить через класс LocationManager в библиотеке для андройд приложений android.location.LocationManager.

Этап коррекции в фильтре Калмана по данным от гироскопа:

$$\begin{cases} g_{x,k}=\omega_{x,k}+\xi_{20,k} \\ g_{y,k}=\omega_{y,k}+\xi_{21,k} \\ g_{z,k}=\omega_{z,k}+\xi_{22,k} \end{cases}, \quad (43)$$

где $g_{x,k}, g_{y,k}, g_{z,k}$ – входные данные от гироскопа в k-й момент времени, каждое из которых обозначает угловое ускорение устройства вокруг осей OX, OY и OZ в рад/с;

$\omega_{x,k}, \omega_{y,k}, \omega_{z,k}$ – угловое ускорение устройства вокруг осей OX, OY и OZ в рад/с в k-й момент времени, взятые из вектора состояния x_k;

$\xi_{20,k}, \xi_{21,k}, \xi_{22,k}$ – ошибки измерения входных данных от гироскопа в k-й момент времени.

Матрица наблюдения H для гироскопа:

$$H=\begin{pmatrix} 0&0&0&0&0&0&0&0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&1 \end{pmatrix}. \quad (44)$$

Вектор измерений Z для гироскопа:

$$Z=(g_x, g_y, g_z)^T, \quad (45)$$

где g_x, g_y, g_z – угловые ускорения устройства вокруг осей OX, OY и OZ в рад/с на текущий момент.

Матрица ковариации R входных данных от гироскопа:

$$R = \begin{pmatrix} \delta_{g,x} \cdot dt^2 & 0 & 0 \\ 0 & \delta_{g,y} \cdot dt^2 & 0 \\ 0 & 0 & \delta_{g,z} \cdot dt^2 \end{pmatrix}, \quad (46)$$

где $\delta_{g,x}, \delta_{g,y}, \delta_{g,z}$ – среднеквадратичная ошибка угловых ускорений g_x, g_y, g_z устройства в рад/с, которая определяется в движении устройства.

Этап коррекции в фильтре Калмана по данным от линейного акселерометра:

$$R_x = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(-\alpha_k) & -\sin(-\alpha_k) \\ 0 & \sin(-\alpha_k) & \cos(-\alpha_k) \end{pmatrix}, \quad (47)$$

$$R_y = \begin{pmatrix} \cos(-\beta_k) & 0 & -\sin(-\beta_k) \\ 0 & 1 & 0 \\ \sin(-\beta_k) & 0 & \cos(-\beta_k) \end{pmatrix}, \quad (48)$$

$$R_z = \begin{pmatrix} \cos(-\gamma_k) & -\sin(-\gamma_k) & 0 \\ \sin(-\gamma_k) & \cos(-\gamma_k) & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad (49)$$

$$\begin{pmatrix} acc_{x,k} \\ acc_{y,k} \\ acc_{z,k} \end{pmatrix} = Rx \cdot Ry \cdot Rz \cdot \begin{pmatrix} a_{x,k} \\ a_{x,k} \\ a_{x,k} \end{pmatrix} + \begin{pmatrix} \xi_{23,k} \\ \xi_{24,k} \\ \xi_{25,k} \end{pmatrix}, \quad (50)$$

где $\alpha_k, \beta_k, \gamma_k$ – углы поворота устройства вокруг осей OX, OY и OZ в радианах в k-й момент времени, взятые из вектора состояния x_k ;

$acc_{x,k}, acc_{y,k}, acc_{z,k}$ –линейное ускорение устройства вдоль осей OX, OY и OZ в м/с^2 в k-й момент времени, которые являются входными данными от линейного акселерометра;

$a_{x,k}, a_{y,k}, a_{z,k}$ – линейное ускорение вдоль осей OX, OY и OZ в м/с^2 в k-й момент времени, взятые из вектора состояния x_k ;

$\xi_{23,k}, \xi_{24,k}, \xi_{25,k}$ – ошибки измерения входных данных от линейного акселерометра в k-й момент времени в м/с^2.

Выражение (43) необходимо записано в линейном виде. Так как линейное представление выражение (43) будет большим, то его линейное представление опустим.

Матрица ковариации R входных данных от линейного акселерометра:

$$R=\begin{pmatrix} \delta_{acc,x}\cdot dt^2 & 0 & 0 \\ 0 & \delta_{acc,y}\cdot dt^2 & 0 \\ 0 & 0 & \delta_{acc,z}\cdot dt^2 \end{pmatrix}, \quad (51)$$

где $\delta_{acc,x}, \delta_{acc,y}, \delta_{acc,z}$ – среднеквадратичная ошибка линейного ускорения $a_{x,k}, a_{y,k}, a_{z,k}$ устройства в м/с^2, которая определяется в движении устройства.

2.4.Схема алгоритма фильтра Калмана

Алгоритм Калмана состоит из двух часте: этапа предсказания и этапа коррекции. На этапе коррекции необходимо задать соответствующие матрицы наблюдения H и ковариации R, а также вектор наблюдения Z для датчика, данные от которого поступили. Схема алгоритма фильтра Калмана проиллюстрирована на рисунке 2.3.

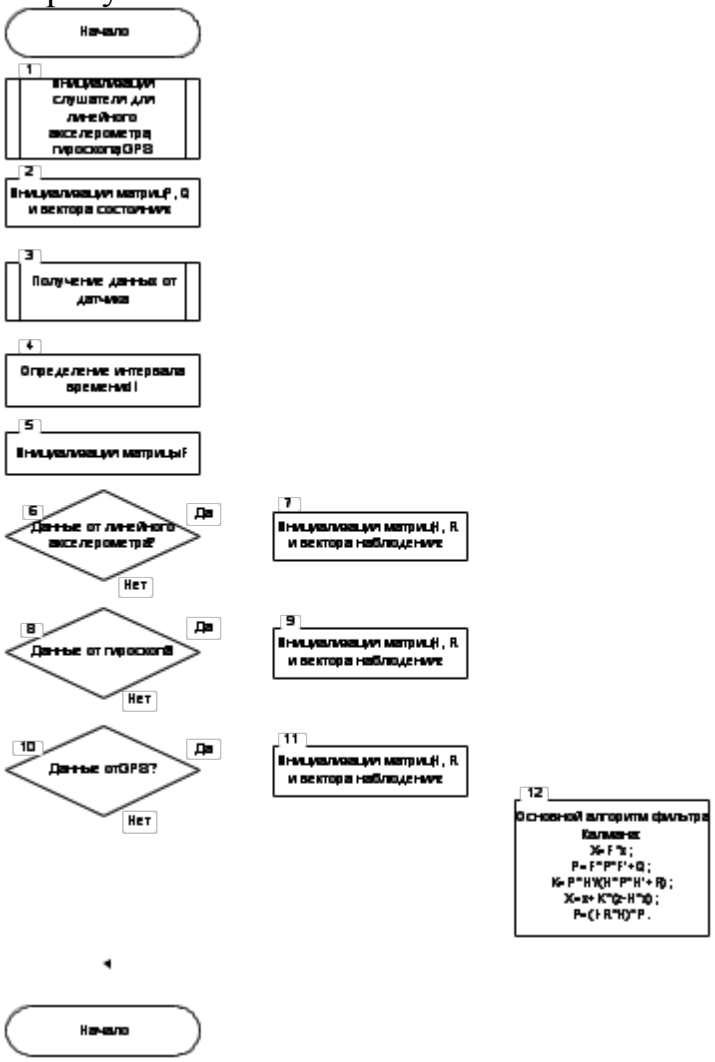

Рис. 2.3. Схема алгоритма фильтра Калмана

2.4. Выводы

Для работы приложения необходимо иметь android устройство с версией операционной системы 5.0-5.1 и интегрированными датчиками: GPS, линейный

акселерометр и гироскоп. Android устройство должно быть зафиксировано на объекте отслеживания.

Данные, полученные от линейного акселерометра, необходимо дважды интегрировать по времени, чтобы получить пройденный путь за период времени. Ошибка, которая имеется в измеренных данных линейным акселерометром также дважды интегрируется по времени и абсолютная ошибка пройденного пути начинает резко расти по экспоненциальной прямой с увеличением количества итераций получения данных от датчика.

Данные, полученные от гироскопа, в отличии от линейного акселерометра, необходимо только один раз интегрировать по времени, чтобы получить углы поворота устройства вместе с объектом . Также ошибка в измеренных данных от гироскопа будет интегрироваться по времени, но не так резко расти с поступление новых данных от датчика, как в случае линейного акселерометра.

Для фильтрации ошибок данных используется оптимальный рекуррентный линейный фильтр Калмана.

Матрица эволюции физического процесса F описывает динамическое движение в глобальной системе координат. Она предсказывает будущее значение вектора состояния. Вектор состояния содержит данные, которые характеризуют объект в инерционной системе отчета, а именно: угловое ускорение вокруг осей OX, OY и OZ, угол поворота относительно начала глобальной системы координат вокруг осей OX, OY и OZ, линейное ускорение и скорость вдоль осей OX, OY и OZ, координаты в глобальной системы координат.

Для матрицы эволюции F определена ковариационная матрица ошибки Q. Так ошибка ускорения и углового ускорения пропорциональна промежутку времени между приходящими данными от датчиков GPS, линейного акселерометра и гироскопа. Чтобы задать матрицу ковариации необходимо подсчитать среднеквадратичную ошибку покоя от датчиков линейного акселерометра и гироскопа.

Из-за того, что данные от датчиков поступают не одновременно, в реализации линейного фильтра Калмана будет присутствовать три матрицы наблюдения:

1). Для линейного акселерометра, сопоставляет входные данные, описывающие процесс линейного ускорения в локальной системе координат, и предсказанные данные вектора состояния системы, в глобальной системе координат. Матрица наблюдения содержит в себе поворот на углы, которые были проинтегрированы от данных гироскопа за все время работы алгоритма, по осям OX, OY и OZ, чтобы преобразовать предсказанное ускорение устройства в глобальной системе координат к предсказанному ускорению в локальной системе координат.

2). Для гироскопа, сопоставляет входные данные от датчика, а именно, угловое ускорение устройства за dt период времени и предсказанное угловое ускорение.

3). Для GPS датчика, сопоставляет входные данные в мировой системе координат к предсказанных данным в глобальной системе координат.

Для каждого из трех используемых датчиков в фильтре Калмана на этапе коррекции задана матрица ковариации вектора наблюдения.

ГЛАВА 3. ОЦЕНКА ЭФФЕКТИВНОСТИ РАБОТЫ АЛГОРИТМА

3.1.Оценка качества работы фильтра Калмана

Алгоритм фильтрации данных от ошибок тестировался на парковке шириной 10 метров и длиной 20 метров, траектория пути описывает букву "П". График тестирования фильтра Калмана отображен на рисунке 3.1, где траектория по зеленым точкам – это траектория пути по данным от линейного акселерометра и гироскопа без фильтрации, траектория по красным звездочкам – это траектория пути по данным от GPS, траектория по синим точкам – это траектория пути по данным от GPS, линейного акселерометра и гироскопа в фильтре Калмана. На двадцатой секунде в алгоритме перестают использоваться данные от GPS. Этот момент на рисунке отображен сильным расхождением траектории по данным от GPS и фильтрованными данными. Как можно увидеть по рисунку, ошибка линейного акселерометра начинает со временем сильнее сказываться на результат фильтрации данных. Вся траектория составляет 76 секунд, 20 из которых ушли на адаптацию параметров фильтра Калмана. Траектория пути по данным от GPS также содержит ошибки, в траектории невозможно разглядеть букву "П".

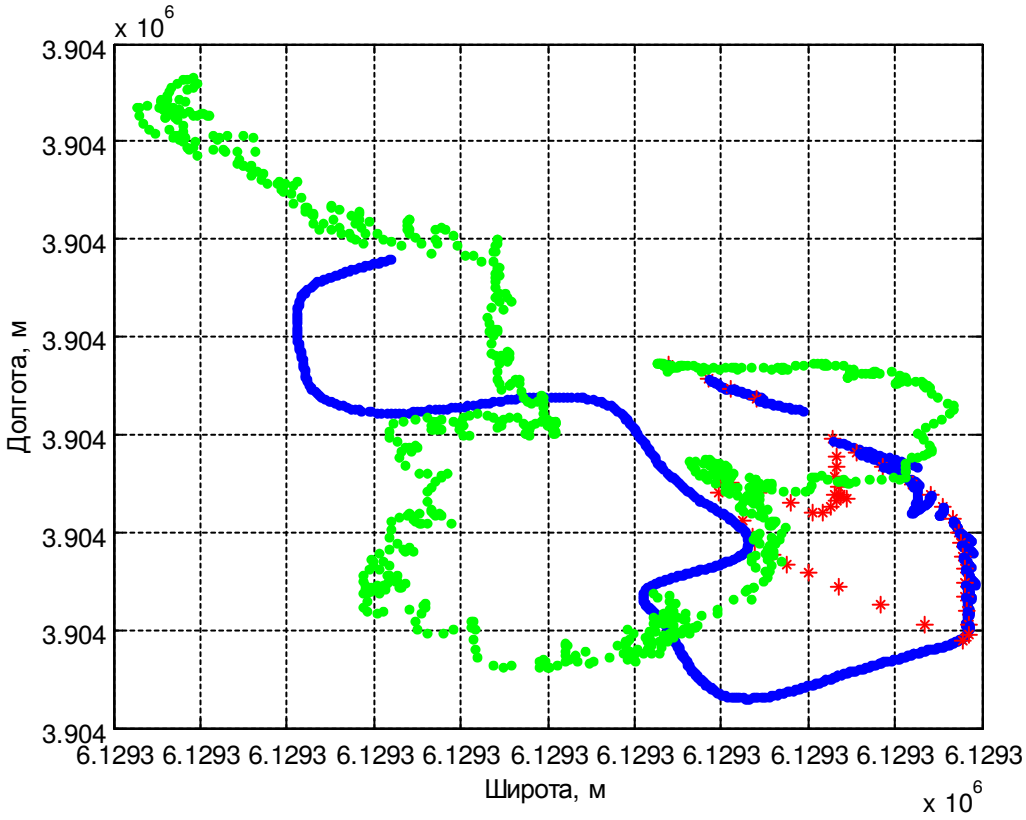

Рис. 3.1. График предсказания ошибки фильтром Калмана

Пройденный путь по данным от фильтра Калмана определяется с точностью, которая прямо пропорциональна времени, когда данные от GPS перестали поступать в алгоритм.

На рисунке 3.2 отображен прямолинейный участок пути. Красная линия – это путь по данным от GPS, синяя линия – это путь по данным от линейного акселерометра и гироскопа. Первые 180 метров идут на адаптацию параметров фильтра Калмана, это также видно по графику, синяя и красная линии наложены друг на друга. Затем на фильтр Калмана перестают поступать данные от GPS и можно увидеть на рисунке, как красная и синяя линии расходятся.

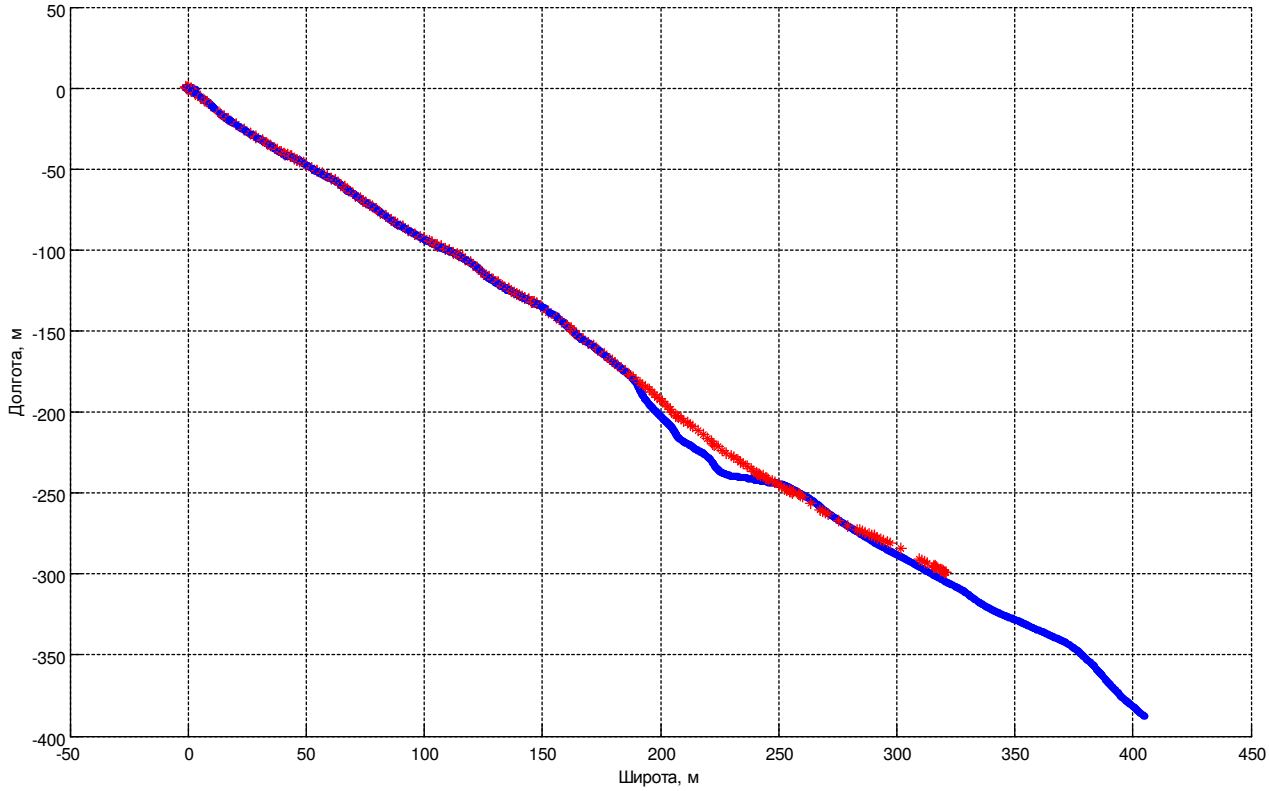

Рис. 3.2. График предсказания ошибки фильтром Калмана на прямолинейном участке пути

Данные, смоделированной ситуации, отображены в таблице 3.1. По данным из таблицы видно, как возрастает ошибка, которая накапливается по данным от линейного акселерометра и гироскопа.

Таблица 3.1

Абсолютный путь по GPS, м	Абсолютный путь по линейному акселерометру и гироскопу, м	Абсолютная ошибка, м
1,356599621474743	0,954689664766192	0,401909956708551
2,525717821903527	2,676689349114895	0,150971527211368
5,205842683091760	4,887426840141416	0,318415842950344
23,969455719925463	28,316955675370991	4,347499955445528
27,861415018327534	32,969619445502758	5,108204427175224
30,518553709611297	37,237880124710500	6,719326415099204

3.2.Выводы

Разработанный фильтр Калмана на вход принимает данные от линейного акселерометра, гироскопа и GPS. На выходе из фильтра получаем координаты местоположения устройства. Когда фильтр Калмана перестает получать данные от GPS, то входные данные от гироскопа и линейного акселерометра начинают сильнее вносить ошибку в расчет координат, чем дольше нет данных от GPS, тем сильнее сказывается ошибка. В такой ситуации можно говорить о местоположении устройства с некоторой точностью. Если два объекта будут перемещаться с различной друг от друга скоростью, то отношение ошибки и пройденного пути будет также различным.

Фильтр Калмана аппроксимирует входные данные от линейного акселерометра и гироскопа от шумов, которые появились во время движения, тем самым по данных из фильтра можно определить траекторию с некоторой точностью.

ЗАКЛЮЧЕНИЕ

Сервисы геолокации подвижных объектов обладают множеством функций, большая часть из которых относится к реализации серверной части данного сервиса. Функции, которые можно отнести к клиентской части, это определение местоположение через GPS / ГЛОНАСС с высокой точностью и, если нет сигнала GPS / ГЛОНАСС, то определение местоположение от базовых станций GSM и WiFi-сетей. Точность определения местоположения от базовых станций GSM и WiFi-сетей зависит от количества помех, стоящих на пути прохождения сигнала, погрешность может определяться от нескольких метров, до нескольких сотен метров. Что не так хорошо, если объект находится в городе с плотной застройкой территории высокими зданиями. Не все WiFi-сети имеют возможность определять местоположения, связано это с тем, что многие WiFi-модемы не имеют возможности привязки к определенному месту положению.

Технологии определения местоположения по GPS и ГЛОНАСС похожи. Не все android устройства имеют возможность определять местоположение по ГЛОНАСС, так как данная технология появилась намного позже, чем GPS и многие android устройства не были укомплектованы ГЛОНАСС приемниками.

Для работы приложения необходимо иметь android устройство с версией операционной системы 5.0-5.1 и интегрированными датчиками: GPS, линейный акселерометр и гироскоп. Android устройство должно быть зафиксировано на объекте отслеживания.

Данные, полученные от линейного акселерометра, необходимо дважды интегрировать по времени, чтобы получить пройденный путь за период времени. Ошибка, которая имеется в измеренных данных линейным акселерометром также дважды интегрируется по времени и абсолютная ошибка пройденного пути начинает резко расти по экспоненциальной прямой с увеличением количества итераций получения данных от датчика.

Данные, полученные от гироскопа, в отличии от линейного акселерометра, необходимо только один раз интегрировать по времени, чтобы получить углы поворота устройства вместе с объектом . Также ошибка в измеренных данных от гироскопа будет интегрироваться по времени, но не так резко расти с поступление новых данных от датчика, как в случае линейного акселерометра.

Для фильтрации ошибок данных используется оптимальный рекуррентный линейный фильтр Калмана.

Матрица эволюции физического процесса F описывает динамическое движение в глобальной системе координат. Она предсказывает будущее значение вектора состояния. Вектор состояние содержит данные, которые характеризуют объект в инерционной системе отчета, а именно: угловое ускорение вокруг осей OX, OY и OZ, угол поворота относительно начала глобальной системы координат вокруг осей OX, OY и OZ, линейное ускорение и скорость вдоль осей OX, OY и OZ, координаты в глобальной системы координат.

Для матрицы эволюции F определена ковариационная матрица ошибки Q. Так ошибка ускорения и углового ускорения пропорциональна промежутку времени между приходящими данными от датчиков GPS, линейного акселерометра и гироскопа. Чтобы задать матрицу ковариации необходимо подсчитать среднеквадратичную ошибку покоя от датчиков линейного акселерометра и гироскопа.

Из-за того, что данные от датчиков поступают не одновременно, в реализации линейного фильтра Калмана будет присутствовать три матрицы наблюдения: для GPS, для линейного акселерометра и для гироскопа.

Для каждого из трех используемых датчиков в фильтре Калмана на этапе коррекции задана матрица ковариации R вектора наблюдения Z.

Разработанный фильтр Калмана на вход принимает данные от линейного акселерометра, гироскопа и GPS. На выходе из фильтра получаем оптимальные координаты местоположения устройства. Когда фильтр Калмана перестает получать данные от GPS, то входные данные от гироскопа и линейного акселерометра начинают сильнее вносить ошибку в расчет оптимальных координат, чем дольше нет данных от GPS, тем сильнее сказывается ошибка. В такой ситуации можно говорить о местоположении устройства с некоторой точностью.

Данная программа может пригодиться компаниям, которые занимаются уборкой улиц города. Кратковременное пропадание GPS сигнала из-за высотных зданий или иных, подавляющих сигнал, устройств, может спровоцировать ситуацию, когда уборочная техника покидает указанный маршрут. Что соответственно может повлечь за собой претензии представителей компании к водителю уборочной техники. Подобная ситуация может произойти и с инкассаторской службой.

GPS сигнал не проходит через толщу металло-бетонных конструкций зданий и предложенный метод может определить местоположение автомобиля в многоярусных, подземных парковках и туннелях города.

ЛИТЕРАТУРА

1. W.J. Sung, S.O. Choi, K.H. You, TDoA based UGV Localization using Adaptive Kalman Filter Algorithm / Sungkyunkwan University, 2009.
2. Rudy Negenborn, Robot Localization and Kalman Filters On finding your position in a noisy world / Thesis, utrecht university, 2003.
3. Zhan, R. Iterated Unscented Kalman Filter for passive target tracking. // R. Zhan, J. Wan. – IEEE Transaction on A&ES. – 2007. – Vol.43. – №3. – P.1155 – 1163.
4. Shahram Rezaei, Raja Sengupta, Kalman Filter Based Integration of DGPS and Vehicle Sensors for Localization / Department of Civil Engineering University of California at Berkeley, 2003.
5. Somphop Limsoonthrakul, Matthew N. Dailey, Manukid Parnichkun, Intelligent Vehicle Localization Using GPS, Compass, and Machine Vision / Asian Institute of Technology, 2008.
6. Xiong K., Zhang H., Liu L. Adaptive Robust Extended Kalman Filter // Интернет. 2009. URL: http://cdn.intechopen.com/pdfs/6327/InTech-Adaptive_robust_extended_kalman_filter.pdf (дата обращения: 08.06.2016).
7. Jwo D-J., Chung F-C. Adaptive Kalman filter for navigation sensor fusion // Интернет. 2009. URL: http://cdn.intechweb.org/pdfs/11661.pdf (дата обращения: 08.06.2016).
8. Jwoi D-J., Cheni M-Y. Adaptive and Nonlinear Kalman filtering for GPS Navigation Processing // Интернет. 2008. https://www.intechopen.com/books/kalman_filter_recent_adavnces_and_applications/adaptive_and_nonlinear_kalman_filtering_for_gps_navigation_processing (дата обращения: 08.06.2016).
9. Moriya N. Non-Stationary noise estimation in Adaptive Linear and Extended Kalman Filtering // Интернет. 2007. URL: https://www.semanticscholar.org/paper/Non-stationary-Noise-Estimation-in-Adaptive-Linear-Moriya/b10a61d1d958205a1e8afd267c28e25342a58bf3 (дата обращения: 08.06.2016).
10. Li, X.R. Best linear unbiased filtering with nonlinear measurement for target tracking // X.R. Li, Z. Zhao, V.P. Jilkov. IEEE Transaction on A&ES. – 2004. – Vol.40. – №4. – P.1324 – 1336.
11. Li, X.R. Survey of maneuvering target tracking – Part III: Measurement models // X.R. Li, V.P. Jilkov. – Proceedings of SPIE Conference on signal and Data processing of small target, USA. – 2001. – 24p.
12. M. Ribeiro. Kalman and extended Kalman filters: Concept, derivation and properties. Institute for Systems and Robotics, p. 43, 2004.
13. R. Cucchiara, C. Grana, M. Piccardi, and A. Prati. Detecting moving objects, ghosts and shadows in video streams. IEEE Trans. on Patt. Anal. and Machine Intell., vol. 25, no. 10, Oct. 2003, pp. 1337-1342 .
14. Peter S. Maybeck, ”The Kalman Filter: An Introduction to Concepts,” in Autonomous Robot Vehciles, I.J. Cox, G. T. Wilfong (eds), Springer-Verlag, 1990.

15. B.P.L. Lo and S.A. Velastin. Automatic congestion detection system for underground platforms. Proc. of 2001 Int. Symp. on Intell. Multimedia, Video and Speech Processing, pp. 158-161, 2000.
16. C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder:Real-time Tracking of the Human Body. IEEE Trans. on Patt. Anal. and Machine Intell., vol. 19, no. 7, pp. 780-785, 1997.
17. C. Stauffer, W.E.L. Grimson. Adaptive background mixture models for real-time tracking. Proc. of CVPR 1999, pp. 246-252.
18. Elgammal, A., Harwood, D., and Davis, L.S. Non-parametric Model for Background Subtraction. Proc. of ICCV '99 FRAME-RATE Workshop, 1999.
19. J. M. F. Moura, “Linear and Nonlinear Stochastic Filtering,” NATO Advanced Study Institute, Les Houches, September 1985.
20. R.E. Kalman A New Approach to Linear Filtering and Prediction Problems, ASME J. Basic Eng'g, 82 (Series D) 35-45.
21. R.E. Kalman Mathematical Description of Linear Dynamical Systems, J. SIAM Control, A 1(2) (1963) 152-192.
22. R.E. Kalman and R.S. Bucy A New Approach to Linear Filtering and Prediction Problems , ASME J. Basic Eng'g, 82D(1) (1960) 35-45.
23. R.E. Kalman and R.S. Bucy New Results in Linear Filtering and Prediction Theory ASME J. Basic Eng'g, 83D(1) (1961) 95-108.
24. Kalman, R.E. (1960). A new approach to linear filtering and prediction problems. Journal of Basic Engineering 82 (1): pp. 35—45.
25. E. Foxlin, ”Inertial heading-tracker sensor fusion by a complementary separate-bias Kalman filter,” in Proceeding of VRAIS IEEE, 1996.
26. Haykin, Simon. Kalman filtering and neural networks. Vol. 47. John Wiley & Sons, 2004.
27. Gordon, Neil, B. Ristic, and S. Arulampalam. "Beyond the kalman filter: Particle filters for tracking applications." Artech House, London (2004).
28. Schmidt, Stanley F. "The Kalman filter-Its recognition and development for aerospace applications." Journal of Guidance, Control, and Dynamics 4.1 (1981): 4-7.
29. Fritsche C., Klein A., Wurtz D. Hybrid GPS/GSM Localization of Mobile Terminals using the Extended Kalman Filter // IEEE. — 2009.
30. Мэйбек, Питер С. 1979. Стохастические модели, оценка и контроль, Том 1, Academic Press, Inc.
31. Where'sMyDroid - Features. [Электронныйресурс]. – Режим доступа: http://wheresmydroid.com/features.html, свободный – Дата обращения 8.01.2016
32. GPS Trace Orange 2.0 – GPS для авто, семьи, детей и собак. [Электронныйресурс]. – Режим доступа: http://gps-trace.com/#features, свободный – Дата обращения 8.01.2016.
33. ГдеМои®. [Электронныйресурс]. – Режим доступа: https://www.gdemoi.ru, свободный – Дата обращения 8.01.2016.
34. Загрузка Android Studio и инструментов SDK | Android Developers. [Электронныйресурс]. – Режим доступа:

http://developer.android.com/intl/ru/sdk/index.html, свободный – Дата обращения 8.01.2016.
35. А.А. Дегтярёв, Ш. Тайль. Элементы теории адаптивного расширенного фильтра Калмана / Препринт ИПМ им. М.В. Келдыша РАН. – М., 2003. – №26. – 35 с.
36. Граничин О.Н., Поляк Б.Т. Рандомизированные алгоритмы оценивания и оптимизации при почти произвольных помехах. М.: Наука. 2003. 291с.
37. Вахитов А.Т., Граничин О.Н., Гуревич Л.С. Алгоритм стохастической аппроксимации с пробным возмущением на входе в нестационарной задаче оптимизации. Автоматика и телемеханика, 2009, № 11. С. 70-79.
38. Степанов, О.А. Сравнительное использование линейного и нелинейного оптимальных алгоритмов оценивания в задачах обработки навигационной информации. // О.А. Степанов, А.Б. Торопов. – Спб: Гироскопия и навигация. – 2010. – №3. – С.24-36
39. Степанов, О.А. Линейные оптимальные алгоритмы в задачах оценивания с нелинейными измерениями. Связь с алгоритмами калмановского типа // О.А. Степанов, А.Б. Торопов. – Известия ТулГУ. Технический вестник, 2012. – № 7 – С. 172-189.
40. Хмарский П.А. Влияние выбора моделей входного воздействия на точность измерений вектора состояния для фильтров Калмана // П.А. Хмарский, А.С. Солонар. – Минск: Доклады БГУИР. – 2012. – №7. – С.47-53.
41. Хмарский, П.А. Особенности работы алгоритма ансцентного фильтра Калмана при наблюдении объектов в полярных координатах // П.А. Хмарский, А.С. Солонар. – Минск: Доклады БГУИР. – 2013. – №2. – С 79-86.
42. Хмарский, П.А. Оценка влияния условий наблюдения на показатели качества модификаций дискретных фильтров Калмана при косвенном измерении // П.А. Хмарский, А.С. Солонар. –материалы III Международной научно-практической конференции [Электронный ресурс] / УО «Гр. ун-т им. Я. Купалы». – Гродно, 2013. – С. 143-157.
43. Грачев, А.Н. Методика синтеза итерационных алгоритмов совместного оценивания параметров и состояния линейных дискретных систем // А.Н. Грачев, С.В. Шурыгин. М. – Труды VII Международной конференции «Идентификация систем и задачи управления», 2008. – С. 204-219.
44. Понятский, В.М. Повышение точности системы управления при использовании фильтра Калмана // Сборник материалов XIX Всероссийская межвузовская научно-техническая конференция «Электромеханические и внутрикамерные процессы в энергетических установках, струйная акустика и диагностика, приборы и методы контроля природной среды, веществ, материалов и изделий» (14-16 мая 2007 г) – Казань. КВАКУ, 2007. Часть 1, С 265 – 267.
45. Браммер, К. Фильтр Калмана-Бъюси. Детерминированное наблюдение и стохастическая фильтрация /К. Браммер, Г. Зиффлинг. — М.: Наука, 1982. Згуровский М.З., Подладчиков В.Н. Аналитические методы калмановской

фильтрации для систем с априорной неопределенностью. – К.: Наукова думка, 1995. – 298 с.
46. Забегаев А.Н., Павловский В.Е. Адаптация фильтра Калмана для использования с локальной и глобальной системами навигации // Интернет. 2009. URL: http://www.raai.org/resurs/papers/kii-2010/doklad/zabegaev.pdf (дата обращения: 08.06.2016).
47. Шахтарин Б.И. Фильтры Винеры и Калмана. М.: Гелиос АРВ, 2008. 408 с.
48. Цибизова Т.Ю., Шэнь Кай, Неусыпин К.А. Исследование алгоритмов оценивания в задаче коррекции навигационных систем летательных аппаратов // Фундаментальные исследования. – 2015. – № 6-2. – С. 301-305;
49. Кузовков Н. Т., Салычев О. С. Инерциальная навигация и оптимальная фильтрация. — М.: Машиностроение, 1982. — 216 с.
50. Кошаев Д.А. Многоальтернативный метод обнаружения и оценки нарушений на основе расширенного фильтра Калмана // Автоматика и Телемеханика. 2010. №5. С. 70-83.

ПРИЛОЖЕНИЕ 1
Описание программы

П1.1. ОБЩИЕ СВЕДЕНИЯ О ПРОГРАММЕ

Название программного продукта – «android tracker». Для использования программы необходимо android устройство с операционной системой android версии 5.0-5.1 и наличие интегрированных датчиков: линейный акселерометр, гироскоп и GPS. Для передачи данных необходим беспроводной интернет. Программа написана на языке Java.

П1.2. ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ

Программа предназначена для определения местоположения android устройства в зонах, где нет GPS сигнала.

Программа обладает следующими возможностями.

1. Фильтрация данных от датчиков.
2. Определение местоположения по данным от GPS.
3. Определение местоположения по данным от линейного акселерометра и гироскопа.
4. Отправление данных о местоположении по указанному адресу сервера и с указанным номером пользователя.

П1.3. ОПИСАНИЕ ЛОГИЧЕСКОЙ СТРУКТУРЫ

Работа приложения происходит в соответствии с алгоритмом, представленным на рис. П1.1.

В файле «activity_main.xml» содержится описание графического интерфейс программы «android tracker».

В файле «MainActivity.java» содержится весь исполняемы код программы.

В файле «AndroidManifest.xml» содержится информация о программе, которая требуется системе Android.

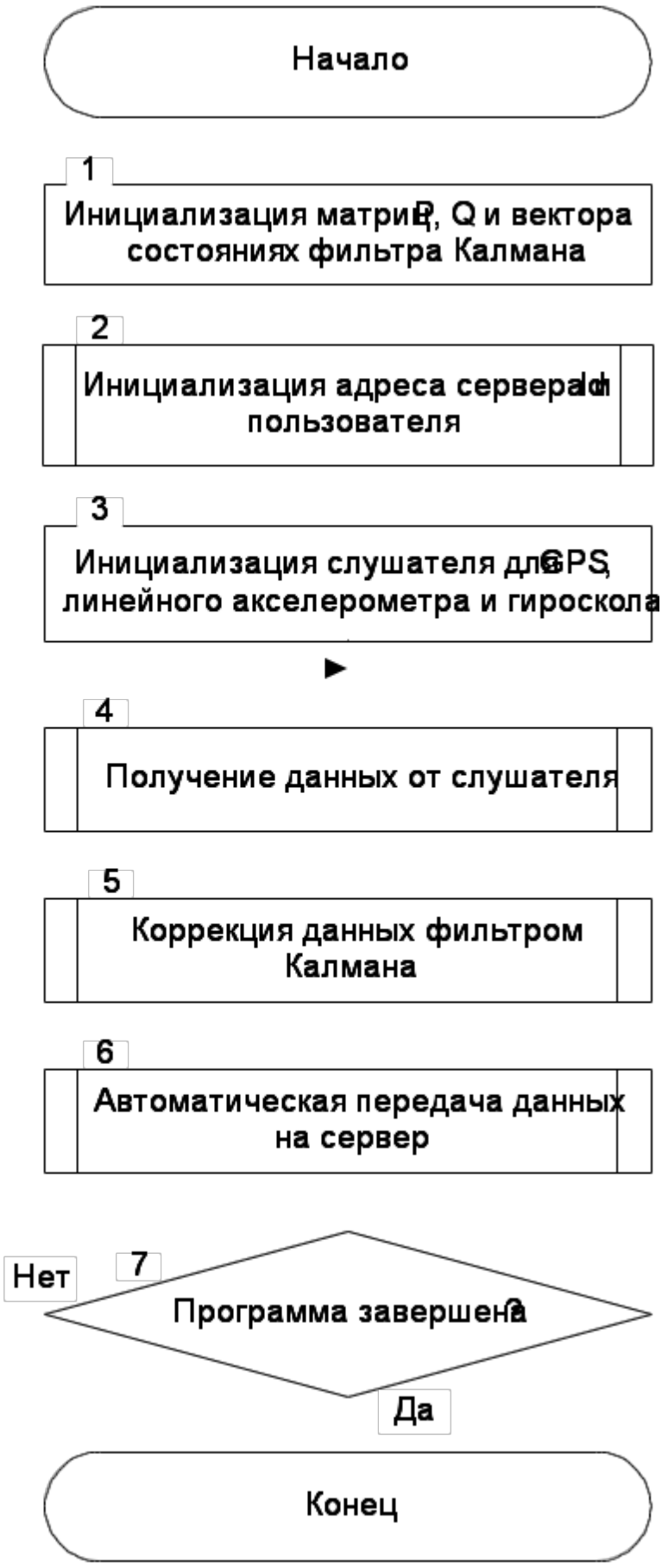

Рис. П1.1. Схема основного алгоритма программы

П1.4. ИСПОЛЬЗУЕМЫЕ ТЕХНИЧЕСКИЕ СРЕДСТВА

Для работы программы необходимо android устройство со следующей конфигурацией:

– процессор с тактовой частотой 1 ГГц;

– объем оперативной памяти 512 Мб;

– объем памяти на постоянном запоминающем устройстве 25 МБ;

– интегрированные датчики GPS, гироскоп и линейный акселерометр.

На android устройстве должна быть установлена операционная система android версии 5.0-5.1.

П1.5. ВЫЗОВ И ЗАГРУЗКА

Чтобы установить программу, необходимо открыть файл «android tracker.apk», в появившемся окне согласиться о предоставлении программе разрешений. Программа установлена.

Чтобы запустить программу, необходимо один раз прикоснуться к иконке программы «android tracker» в меню «Приложения» в операционной системе android.

П1.6. ВХОДНЫЕ И ВЫХОДНЫЕ ДАННЫЕ

Входными данными являются электронный адрес сервера, номер пользователя и данные от датчиков GPS, гироскопа и линейного акселерометра.

Выходными данными являются координаты местоположения android устройства и номер пользователя.

ПРИЛОЖЕНИЕ 2

Руководство пользователя

П2.1. ОБЩИЕ СВЕДЕНИЯ О ПРОГРАММЕ

Программа предназначена для отслеживания местоположения автотранспорта для увеличения уровня защиты.

Для запуска и работы программы требуются следующие аппаратные и программные компоненты:

- android устройство;
- операционная система android 5.0-5.1;
- процессор с тактовой частотой 1 ГГц;
- объем оперативной памяти 512 Мб;
- объем памяти на постоянном запоминающем устройстве 25 МБ;
- интегрированные датчики GPS, гироскоп и линейный акселерометр.

П2.2. СТРУКТУРА ПРОГРАММЫ

Программа распространяется одним файлом «android tracker.apk».

П2.3. НАСТРОЙКА ПРОГРАММЫ

Программное и аппаратное обеспечение должно соответствовать требованиям в разделе П2.1.

Перед использованием программы по назначению необходимо положить android устройство на не двигающуюся поверхность и нажать кнопку «Настроить».

Чтобы использовать программу по назначению необходимо задать электронный адрес сервера и номер пользователя, который будет привязан к пользователю на сервере, и нажать кнопку «Применить сервер и id».

П2.4. ВЫПОЛНЕНИЕ ПРОГРАММЫ

Чтобы запустить программу, необходимо один раз прикоснуться к иконке программы «android tracker» в меню «Приложения» в операционной системе android.

П2.5. ПРОВЕРКА ПРОГРАММЫ

Программа проверена на сервере «http://www.tracker11111.somee.com». На рис. П2.1. показан графический интерфейс программы «android tracker». Задается

сервер «http://www.tracker11111.somee.com» и пользователю устанавливаем идентификатор под номером один. Обязательно надо нажать кнопку «ПРИМЕНИТЬ СЕРВЕР И ID». После нажатия на кнопку должно появиться внизу сообщение «сервер и id заданы», которое обозначает, что данные о местоположении уже отправляются на сервер (рис. П2.2). Для фильтрации ошибки входных данных от датчиков необходимо подсчитать среднеквадратичную ошибку покоя, для этого устройство необходимо положить на поверхность стола и нажать кнопку «НАСТРОИТЬ». По окончанию появится сообщение о завершении настройки (рис П2.3).

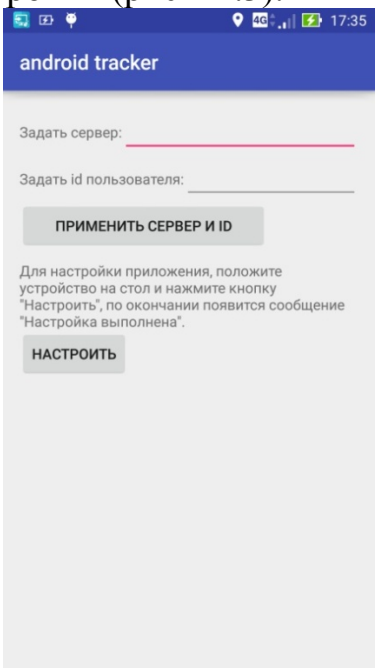

Рис. П2.1. Графический интерфейс программы android tracker

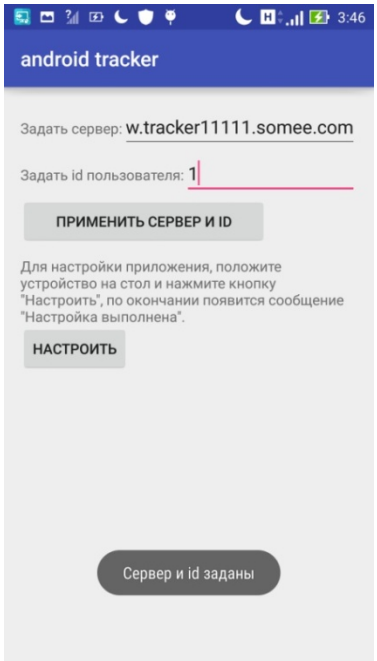

Рис. П2.2. Оповещение о задании электронного адреса сайта и id пользователя

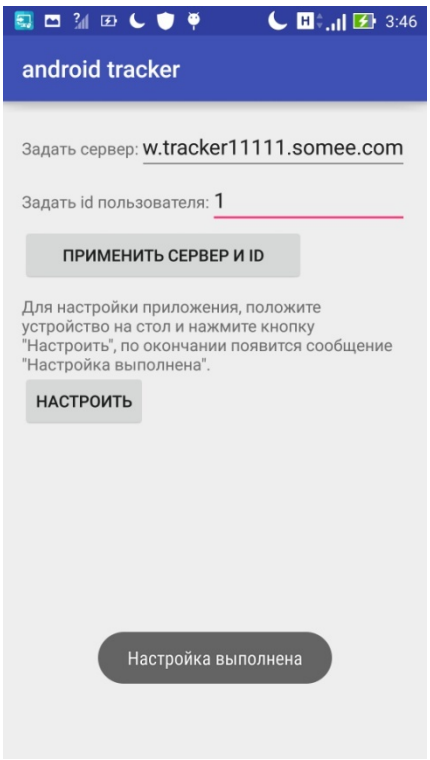

Рис. П2.3. Сообщение о завершении настройки

П2.6. ВХОДНЫЕ И ВЫХОДНЫЕ ДАННЫЕ

Входными данными являются электронный адрес сервера и номер пользователя.

Выходными данными являются координаты местоположения android устройства и номер пользователя.

ПРИЛОЖЕНИЕ 3

Текст программы

П3.1. ФАЙЛ «AndroidManifest.xml»

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.androidtracker" >

    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
    <uses-permission android:name="com.google.android.providers.gsf.permis-
sion.READ_GSERVICES" />
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme" >
        <activity android:name=".MainActivity" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

П3.2. ФАЙЛ « MainActivity.java»

```
package com.example.androidtracker;

import android.content.Intent;
import android.hardware.Sensor;
import android.hardware.SensorEvent;
import android.hardware.SensorEventListener;
import android.hardware.SensorManager;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.TextView;
import android.widget.Toast;

import com.google.android.gms.maps.GoogleMap;
```

```
import com.google.android.gms.maps.LocationSource;
import com.google.android.gms.maps.OnMapReadyCallback;

import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import java.util.Timer;
import java.util.TimerTask;

public class MainActivity extends AppCompatActivity implements LocationSource, Lo-
cationListener,OnMapReadyCallback {

    SensorManager sensorManager;
    Sensor sensorLinAccel;
    Sensor sensorGyroscope;
    private LocationManager locationManager;

    StringBuilder sb = new StringBuilder();

    TextView tvEnabledGPS;
    TextView tvStatusGPS;
    TextView tvLocationGPS;
    TextView tvEnabledNet;
    TextView tvStatusNet;
    TextView tvLocationNet;

    TextView tvText;
    Timer timer;

    String linAcc="";
    String gyro="";

    Location loc;

    //по диплому
    ArrayList<float[]> mas_lacc_sqr= new ArrayList<float[]>();
    ArrayList<float[]> mas_gyro_sqr= new ArrayList<float[]>();

    float[] sqr_err_lacc=new float[]{0,0,0};
    float[] sqr_err_gyro=new float[]{0,0,0};

    Boolean srart_calc_sqr_err=false;

    long prev_time_lacc=0;
    long prev_time_gyro=0;
    long startTime=0;
    //

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        sensorManager = (SensorManager) getSystemService(SENSOR_SERVICE);
        sensorLinAccel = sensorManager.getDefaultSensor(Sensor.TYPE_LINEAR_ACCEL-
ERATION);
        sensorGyroscope=sensorManager.getDefaultSensor(Sensor.TYPE_GYROSCOPE);
        locationManager = (LocationManager) getSystemService(LOCATION_SERVICE);

        tvText = (TextView) findViewById(R.id.tvText);
       /* tvEnabledGPS = (TextView) findViewById(R.id.tvEnabledGPS);
        tvStatusGPS = (TextView) findViewById(R.id.tvStatusGPS);
```

```
        tvLocationGPS = (TextView) findViewById(R.id.tvLocationGPS);
        tvEnabledNet = (TextView) findViewById(R.id.tvEnabledNet);
        tvStatusNet = (TextView) findViewById(R.id.tvStatusNet);
        tvLocationNet = (TextView) findViewById(R.id.tvLocationNet);*/
    }

    @Override
    public void onMapReady(GoogleMap googleMap) {
        showInfo();
        // Add a marker in Sydney and move the camera
    }

    @Override
    protected void onResume() {
        super.onResume();

        sensorManager.registerListener(listener, sensorLinAccel,SensorManager.SEN-
SOR_DELAY_UI);

        sensorManager.registerListener(listener, sensorGyroscope, SensorManag-
er.SENSOR_DELAY_UI);

        locationManager.requestLocationUpdates(LocationManager.GPS_PROVIDER, 0,
(1/10), locationListener);
        locationManager.requestLocationUpdates( LocationManager.NETWORK_PROVIDER,
0, (1/10),locationListener);
        checkEnabled();
        timer = new Timer();
        TimerTask task = new TimerTask() {
            @Override
            public void run() {
                runOnUiThread(new Runnable() {
                    @Override
                    public void run() {
                        showInfo();
                    }
                });
            }
        };
        timer.schedule(task, 0, 200);
    }
    @Override
    protected void onPause() {
        super.onPause();
        sensorManager.unregisterListener(listener);
        locationManager.removeUpdates(locationListener);
        timer.cancel();
    }

    private LocationListener locationListener = new LocationListener() {

        @Override
        public void onLocationChanged(Location location) {
            showLocation(location);
            loc=location;

            if (location.getProvider().equals(LocationManager.GPS_PROVIDER)) {


            }
```

```
    }

    @Override
    public void onProviderDisabled(String provider) {
        checkEnabled();
    }

    @Override
    public void onProviderEnabled(String provider) {
        checkEnabled();
        showLocation(locationManager.getLastKnownLocation(provider));
        loc=locationManager.getLastKnownLocation(provider);
    }

    @Override
    public void onStatusChanged(String provider, int status, Bundle extras) {
        if (provider.equals(LocationManager.GPS_PROVIDER)) {
          /*  tvStatusGPS.setTextSize(11);
            tvStatusGPS.setText("Status: " + String.valueOf(status));*/

        } else if (provider.equals(LocationManager.NETWORK_PROVIDER)) {
          /*  tvStatusNet.setTextSize(11);
            tvStatusNet.setText("Status: " + String.valueOf(status));*/
        }
        else
        {
          /*  tvStatusNet.setTextSize(11);
            tvStatusNet.setText("Status: NON");*/
        }

    }
};

String format(float values[]) {
    return String.format("%1$.2f\t\t%2$.2f\t\t%3$.2f", values[0], values[1],
            values[2]);
}

void showInfo() {

    sb.setLength(0);
    sb
          /*  .append("\n\nLin accel : " + format(valuesLinAccel))*/
          /*  .append("\n\n err lacc:"+ format(sqr_err_lacc))*/
          /*  .append("\nGyroscope : " + format(valuesGyroscop))*/
          /*  .append("\n\nerr gyro:"+ format(sqr_err_gyro))*/
            .append("")
    ;

    tvText.setText(sb);
}

float[] valuesLinAccel = new float[3];

float[] valuesGyroscop= new float[3];

SensorEventListener listener = new SensorEventListener() {
    @Override
    public void onAccuracyChanged(Sensor sensor, int accuracy) {
    }
    @Override
    public void onSensorChanged(SensorEvent event) {
```

```
            if(event.sensor.getType()==Sensor.TYPE_LINEAR_ACCELERATION) {

                for (int i = 0; i < 3; i++) {
                    valuesLinAccel[i] = event.values[i];
                }
                if(srart_calc_sqr_err==true){
                    if(prev_time_lacc!=0) {
                        float dt =(System.currentTimeMillis() - prev_time_lacc);
                        if(dt!=0) {
                            dt=dt/1000;
                            float a0 = event.values[0] / dt ;
                            float a1 = event.values[1] / dt ;
                            float a2 = event.values[2] / dt ;
                            mas_lacc_sqr.add(new float[]{a0, a1, a2});
                        }

                    }
                    prev_time_lacc = System.currentTimeMillis();

                }

            }

            if(event.sensor.getType()==Sensor.TYPE_GYROSCOPE) {

                for (int i = 0; i <3; i++) {
                    valuesGyroscop[i] = event.values[i];
                }

                if(srart_calc_sqr_err==true){
                    if(prev_time_gyro != 0) {

                        float dt = (System.currentTimeMillis() - prev_time_gyro);
                        if(dt!=0) {
                            dt=dt/1000;
                            float g0 = event.values[0] / dt ;
                            float g1 = event.values[1] / dt ;
                            float g2 = event.values[2] / dt ;
                            mas_gyro_sqr.add(new float[]{g0, g1, g2});
                        }

                    }
                    prev_time_gyro = System.currentTimeMillis();
                }

            }

            if(startTime+10000<System.currentTimeMillis() &&
srart_calc_sqr_err==true){
                srart_calc_sqr_err=false;

                calc_sqr_err();
            }

        }
    };

    private void showLocation(Location location) {
        if (location == null)
            return;
```

```
        if (location.getProvider().equals(LocationManager.GPS_PROVIDER)) {
           /* tvLocationGPS.setText(formatLocation(location));*/
        } else if (location.getProvider().equals(
                LocationManager.NETWORK_PROVIDER)) {
           /* tvLocationNet.setText(formatLocation(location));*/
        }
        else{
            /*tvStatusNet.setText("Status: NON and
"+location.getProvider().equals(LocationManager.GPS_PROVIDER));*/
        }
    }

    private String formatLocation(Location location) {
        if (location == null)
            return "";
        return String.format(
                "Coordinates: lat = %1$.4f, lon = %2$.4f,alt = %3$.4f, time =
%4$tF %4$tT, v=%5$.4f, az=%6$.4f",
                location.getLatitude(), location.getLongitude(),location.getAlti-
tude(), new Date(
                        location.getTime()),location.getSpeed(),location.getBear-
ing());
    }

    private void checkEnabled() {
      /*  tvEnabledGPS.setText("Enabled: "
                + locationManager
                .isProviderEnabled(LocationManager.GPS_PROVIDER));
        tvEnabledNet.setText("Enabled: "
                + locationManager
                .isProviderEnabled(LocationManager.NETWORK_PROVIDER));*/
    }

    public void onClickLocationSettings(View view) {
        startActivity(new Intent(
                android.provider.Settings.ACTION_LOCATION_SOURCE_SETTINGS));
    };



    @Override
    public void activate(LocationSource.OnLocationChangedListener listener) {


    }

    @Override
    public void deactivate() {

    }

    @Override
    public void onProviderDisabled(String provider) {
        // TODO Auto-generated method stub

    }

    @Override
    public void onProviderEnabled(String provider) {
        // TODO Auto-generated method stub

    }
```

```
    @Override
    public void onStatusChanged(String provider, int status, Bundle extras) {
        // TODO Auto-generated method stub

    }

    @Override
    public void onLocationChanged(Location location) {
        //  showLocation(location);
    }
    public void calc_root_mean_sqr_err() {
        srart_calc_sqr_err=true;
        prev_time_lacc=0;
        prev_time_gyro=0;
        startTime=System.currentTimeMillis();

    }

    public void set_serv_and_id() {

        Toast.makeText(this, "Сервер и id заданы", Toast.LENGTH_SHORT).show();
    }

    public void onclick(View v) {
        switch (v.getId()) {
            case R.id.btn_set_serv_and_id:
                set_serv_and_id();
                break;
            case R.id.btn_calc_root_mean_sqr_err:
                calc_root_mean_sqr_err();
                break;

        }

    }
    public void calc_sqr_err(){
        //для акселерометра
        Integer j=0;

        for(float[] item : mas_lacc_sqr) {
            j++;
            for(Integer i=0;i<3;i++ ) {
                sqr_err_lacc[i] = sqr_err_lacc[i] + item[i];
            }
        }
        //находим среднее для акселерометра
        for(Integer i=0;i<3;i++ ) {
            sqr_err_lacc[i] = sqr_err_lacc[i]/j;
        }
        //
        float[] sum=new float[]{0,0,0};

        for(float[] item : mas_lacc_sqr) {
            for(Integer i=0;i<3;i++ ) {
                sum[i] =sum[i]+ ( item[i]-sqr_err_lacc[i] )*( item[i]-
sqr_err_lacc[i] );
            }
        }
        for(Integer i=0;i<3;i++ ) {
            sqr_err_lacc[i] = sum[i]/j;
        }
```

```
        //для гироскопа
         j=0;

        for(float[] item : mas_gyro_sqr) {
            j++;
            for(Integer i=0;i<3;i++ ) {
                sqr_err_gyro[i] = sqr_err_gyro[i] + item[i];
            }
        }
        //находим среднее для гироскопа
        for(Integer i=0;i<3;i++ ) {
            sqr_err_gyro[i] = sqr_err_gyro[i]/j;
        }
        //
        sum[0]=0;
        sum[1]=0;
        sum[2]=0;
        for(float[] item : mas_gyro_sqr) {
            for(Integer i=0;i<3;i++ ) {
                sum[i] =sum[i]+ ( item[i]-sqr_err_gyro[i] )*( item[i]-
sqr_err_gyro[i] );
            }
        }
        for(Integer i=0;i<3;i++ ) {
            sqr_err_gyro[i] = sum[i]/j;
        }
        Toast.makeText(this, "Настройка выполнена", Toast.LENGTH_SHORT).show();
        mas_gyro_sqr.clear();
        mas_lacc_sqr.clear();
    }

}
```

П3.3. ФАЙЛ «activity_main.xml»

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools" an-
droid:layout_width="match_parent"
    android:layout_height="match_parent" android:paddingLeft="@dimen/activity_hor-
izontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin" tools:context=".Main-
Activity">

    <LinearLayout
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_alignParentTop="true"
        android:layout_alignParentStart="true"
        android:weightSum="1">

        <LinearLayout
            android:orientation="horizontal"
            android:layout_width="match_parent"
            android:layout_height="wrap_content">

            <TextView
                android:layout_width="wrap_content"
```

```
                android:layout_height="wrap_content"
                android:text="Задать сервер:"
                android:id="@+id/textView1" />

            <EditText
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:id="@+id/editText"
                android:inputType="textWebEmailAddress" />
        </LinearLayout>

        <LinearLayout
            android:orientation="horizontal"
            android:layout_width="match_parent"
            android:layout_height="wrap_content">

            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="Задать id пользователя:"
                android:id="@+id/textView2" />

            <EditText
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:id="@+id/editText2" />

        </LinearLayout>

        <LinearLayout
            android:orientation="horizontal"
            android:layout_width="match_parent"
            android:layout_height="wrap_content">

            <Button
                android:layout_width="240dp"
                android:layout_height="wrap_content"
                android:text="Применить сервер и id"
                android:onClick="onclick"
                android:id="@+id/btn_set_serv_and_id" />
        </LinearLayout>

        <LinearLayout
            android:orientation="horizontal"
            android:layout_width="match_parent"
            android:layout_height="76dp">

            <TextView
                android:layout_width="wrap_content"
                android:layout_height="80dp"
                android:textAppearance="?android:attr/textAppearanceSmall"
                android:text="Для настройки приложения, положите устройство на
стол и нажмите кнопку &quot;Настроить&quot;, по окончании появится сообщение
&quot;Настройка выполнена&quot;."
                android:id="@+id/textView"
                android:layout_marginBottom="10dp"
                android:layout_marginTop="10dp" />

        </LinearLayout>

        <LinearLayout
            android:orientation="horizontal"
```

```
            android:layout_width="match_parent"
            android:layout_height="30dp"
            android:weightSum="1"
            android:layout_weight="0.21">

            <Button
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="Настроить"
                android:onClick="onclick"
                android:id="@+id/btn_calc_root_mean_sqr_err" />

        </LinearLayout>

        <LinearLayout
            android:orientation="horizontal"
            android:layout_width="match_parent"
            android:layout_height="113dp"
            android:weightSum="1"
            android:layout_weight="0.39">

            <TextView
                android:id="@+id/tvText"
                android:layout_marginTop="0dp"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:textSize="11sp"
                android:top="11sp">
            </TextView>

        </LinearLayout>

    </LinearLayout>

</RelativeLayout>
```