Введение

Проникновение интернета в каждую сферу жизни стало неоспоримым фактом в современном мире. С каждым днем растет число пользователей, а, следовательно, и количество передаваемой информации.

Одной из главных задач при передаче данных является сохранение их целостности. Этого можно достичь, если передавать информацию по надежному каналу связи.

В свою очередь надежность канала связи подразумевает непрерывность сигнала в различных географических условиях. Для того чтобы оценить возможности создания такого канала на какой-либо местности производят расчет радиотрассы.

Расчет радиотрассы включает в себя построение графика высот для выбранной местности, построение линии сигнала с учетом высоты антенн. В результате расчета можно увидеть существует ли возможность беспрепятственного прохождения сигнала на данной местности.

1. Анализ предметной области

1.1 Формулирование технического задания

1.1.1. Наименование приложения

Полное наименование приложения: приложение для расчета и построения радиотрассы

1.1.2. Краткая характеристика области применения

Приложение предназначено для расчета и построения радиотрассы

1.1.3. Цели создания

Функциональным назначение приложения является предоставление возможности построения радиотрассы в офлайн режиме.

1.1.4. Требования к конечному пользователю

Особым требованием к конечному пользователю является базовое понимание принципов расчета и построения радиотрассы.

1.1.5. Требования к программе

Программа должна обеспечивать возможность выполнения перечисленных ниже функций:

1) функция выбора координат источника и приемника;

2) функция расчета радиотрассы;

3) функции взаимодействия с сервером для получения необходимых данных;

В состав технических средств должен входить ПК, включающий в себя:

1) 32 или 64 разрядный процессор с тактовой частотой не меньше 1.0ГГц;

2) дисплей;

3) не менее 1 Гб оперативной памяти;

4) не менее 100 мегабайт свободного дискового пространства;

5) клавиатура и мышь.

Системные программные средства, используемые программой, должны быть представлены ОС Windows 7/8/10;

Программа должна обеспечивать взаимодействие с пользователем посредством графического интерфейса и предоставлять возможность выполнять наиболее часто используемые операции с помощью сочетаний клавиш на клавиатуре.

1.1.5. Требования к программной документации

Состав программной документации должен включать в себя:

1) техническое задание;

2) руководство пользователя;

3) исходный код.

1.1.6. Стадии и этапы разработки

Разработка должна быть приведена в 3 стадии:

1) разработка технического задания;

2) проектирование;

3) внедрение.

На стадии разработки технического задания должен быть выполнен этап разработки, согласования и разработки технического задания.

На стадии проектирования необходимо выполнить следующие этапы:

1) разработка программы;

2) разработка программной документации;

3) испытание программы.

На этапе внедрения выполняется передача программы заказчику.

1.2 Расчет радиотрассы

В техническом задании сказано, что одной из функций, реализованных в приложении, должен быть расчет радиотрассы. Для выполнения данного условия необходимо разобраться в том, что подразумевается под расчетом радиотрассы.

В общем плане расчет включает в себя построение графика высот для выбранной местности, построение линии сигнала с учетом высоты антенн.

Для того чтобы построить график высот, необходимо выполнить следующие действия:

1) Выбрать 2 точки на карте

Эти точки – координаты объектов, между которыми планируется организовать канал связи

2) Вычислить координаты точек, находящихся между объектами

Для вычисления координат точек, находящихся между объектами, можно воспользоваться уравнением прямой. Вычисленная прямая вместе с объектами называется маршрутом радиотрассы. Пример маршрута радиотрассы представлен на рисунке 1.

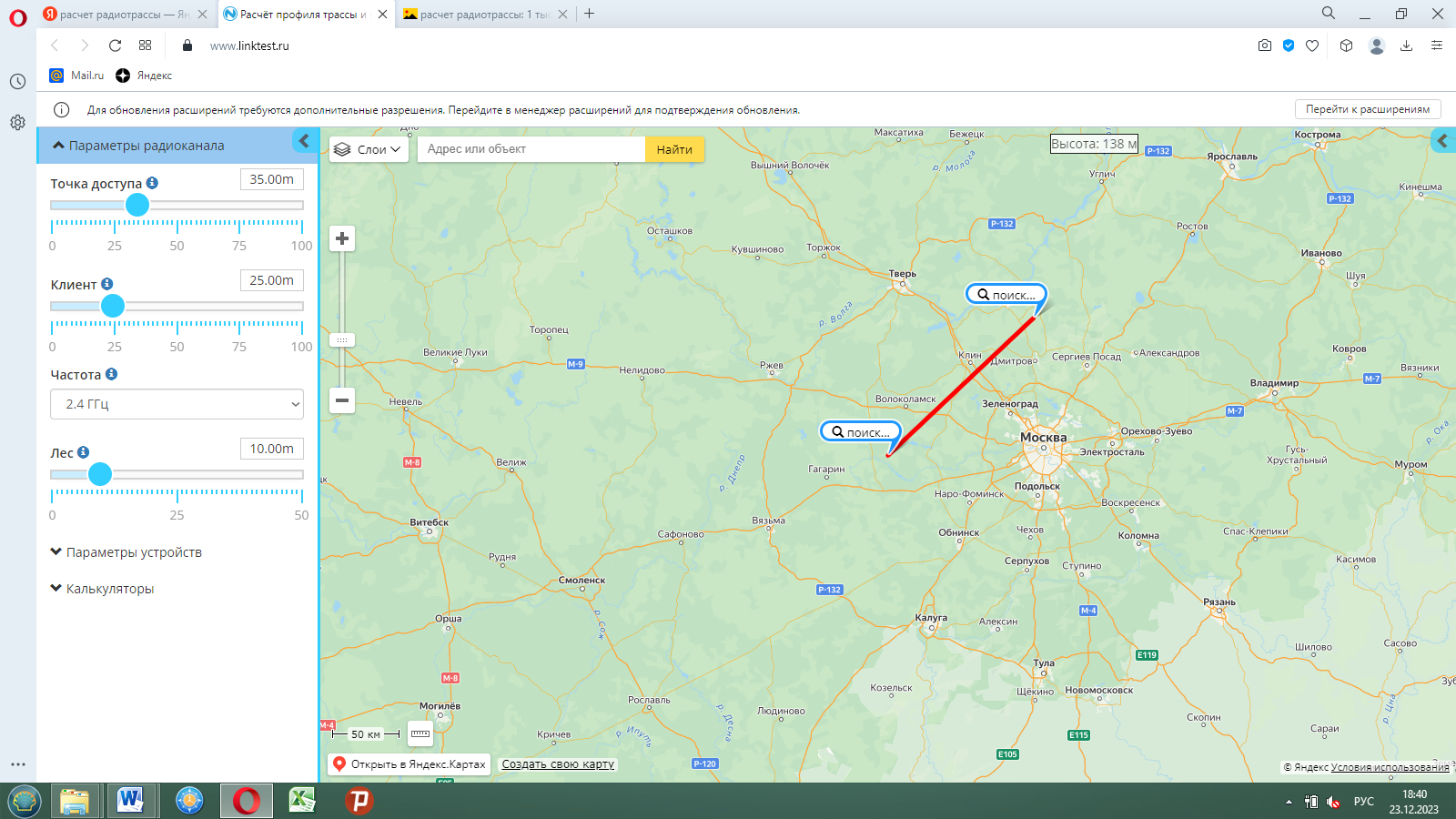


Рисунок 1 – Маршрут радиотрассы

Шаг для вычислений задается из соображений точности, желаемой достичь при расчете радиотрассы.

Если необходимо достичь высокой точности при построении профиля радиотрассы, то берется большое количество точек, находящихся на небольшом расстоянии друг от друга. Следовательно, и шаг должен быть небольшим.

3) Определить высоту для каждой точки маршрута радиотрассы, включая точки объектов

4) Построить профиль радиотрассы

Построение профиля подразумевает построение графика по всем значениям высот, полученным в ходе выполнения предыдущего шага. Стоит отметить, что точки высот наносятся в том порядке, в котором идут координаты точек маршрута трассы.

Дополнительно стоит отметить, что все высоты откладываются относительно нулевой дуги. Нулевая дуга, в свою очередь, рассчитывается по формуле:

Н(Х) = (D²/2R)\*К\*(1-К), км, где

D — расстояние между объектами, км;

R — радиус Земли — 6371 км;

К = Х/D — относительная координата промежуточной точки, 0<K<1;

Х — расстояние от 1-ого объекта до промежуточной точки, км

Для отображения линии связи на графике используются 2 точки: 1 точка – высота 1-ого объекта + высота антенны для 1-ого объекта, 2 точка – высота 2ого объекта + высота антенны для 2-ого объекта. Далее эти 2 точки соединяются линией. Пример расчета радиотрассы представлен на рисунке 2.

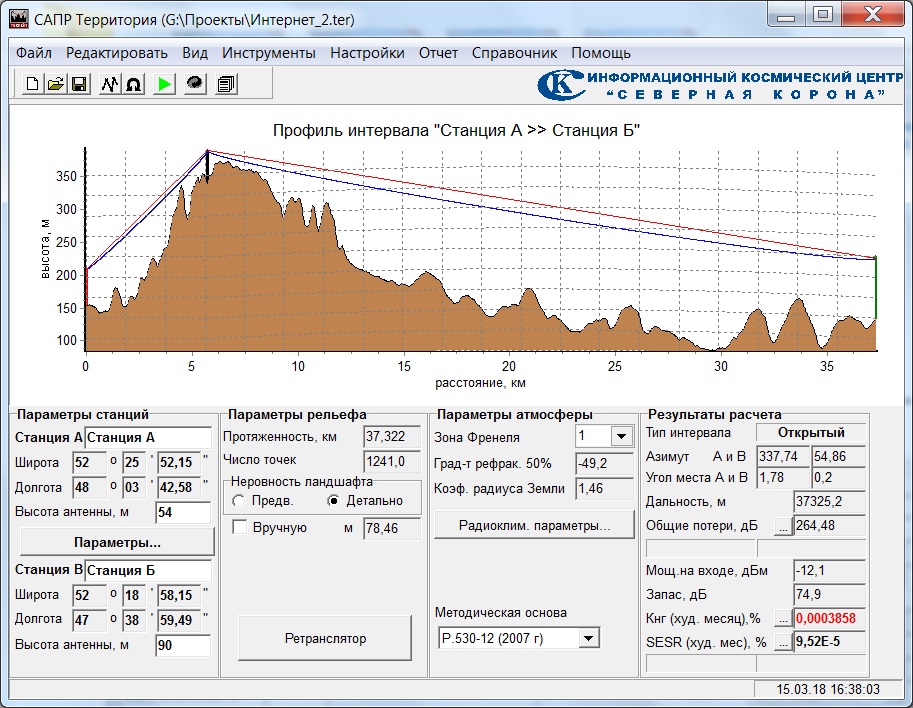


Рисунок 2 – Пример расчета радиотрассы

Пересечение линей связи профиля радиотрассы свидетельствует о невозможности прохождения сигнала. Другими словами, о невозможности установки надежной связи между объектами. Если же линия не пересекает профиль, то связь может быть установлена.

2. Анализ работы сервера карт

В результате разбора последовательности действий для расчета радиотрассы была определена необходимость в получении высот. Так как в техническом задании было указано наличие сервера карт, можно попробовать получать высоты от туда. Для определения этой возможности необходимо выполнить анализ сервера карт.

Анализ сервера карт включает в себя рассмотрение его структуры и принципов взаимодействия с ним.

2.1 Структура сервера карт

Структура сервера карт состоит из 2х частей: сервера и клиентской части. Клиентская часть имеет графический интерфейс и открытий исходный код. Все выводы о принципах работы, о предоставляемых возможностях сервера карт были сделаны при помощи изучения кода клиентской части.

Одной из возможностей, предоставляемой сервером карт, является добавление новой карты на сервер.

Что касается графического интерфейса, то он позволяет просматривать карты, находящиеся на сервере, а также определять координаты тайла карты, выводя их при наведении курсора на тайл.

Общая структура сервера карт представлена на рисунке 3.

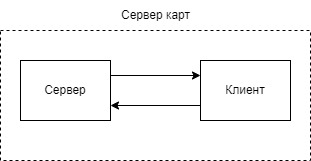


Рисунок 3 – Общая структура сервера карт

2.2 Взаимодействие с сервером карт

Схема взаимодействия с сервером карт представлена на рисунке 4.

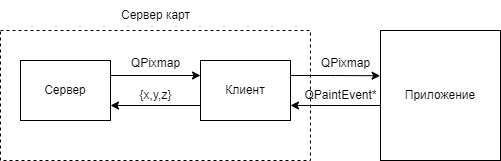


Рисунок 4 – Схема взаимодействия с сервером карт

Из рисунка видно, что в качестве ответа на запрос от клиента сервер возвращает изображение в формате QPixmap. В свою очередь клиент передает это изображение приложению.

Хотя QPixmap не позволяет организовывать прямой доступ к пикселям изображения и манипулировать с ними, но QPixmap можно преобразовать в QImage, который позволяет работать с пикселями.

В свою очередь возможность работать с пикселями может позволить получать значение высоты, если его закодировать в цвете пикселя.

В результате выполнения данного этапа работы было принято решение о создании карты высот, в которой бы цвет пикселя обозначал значение высоты. Дополнительно ко всем функциям, заявленным в техническом задании, в приложении необходимо добавить функцию для декодирования значения цвета пикселя в высоту.

3. Создание карты

На данном этапе работы будет выполнен обзор аналогов. На основании этого обзора будет принято решение об используемом формате данных, о необходимости использования сторонних приложений или программной реализации генерации карты.

3.1 Выбор формата данных

В качестве данных для цифровой модели рельефа можно выделить следующие доступные:

- SRTM

- AsterGDEM

- Etopo2

1) SRTM

Наиболее распространенный формат, используется во многих программах.

В качестве формата для записи высот используются файлы с расширением hgt. Используемое здесь слово «HGT» - это просто сокращение от «высота». Каждый файл представляет собой последовательность 16-разрядных целых чисел, задающих высоту каждой ячейки в метрах, расположенных с запада на восток, а затем с севера на юг. Каждый фрагмент данных продолжительностью 3 угловых секунды содержит 1442401 целое число, представляющее сетку размером 1201 ×1201. Формат доработан для разных систем координат.

В старых версиях данные есть только для области между 60 ° с.ш. и 54 ° ю.ш. С 2014 года данные есть для всей области карты.

Обладает высокой точностью за счет того, что съемка производилась при помощи радиолокации.

2) AsterGDEM

Менее распространенный формат, используется в малом числе программ.

ASTER GDEM охватывает поверхность суши между 83° с.ш. и 83° ю.ш. и насчитывает 22,600 фрагментов размером 1°х1°. Распространяется в формате GeoTIFF в географической системе координат (широта/долгота) и разрешением 1 угловая секунда (примерно 30 метров). Система координат данных WGS84/EGM96.

Точность меньше, чем у srtm, так для использовалась стереофотограмметрическая съемка. И при съемке лесной местности ASTER получает отражение солнечного излучения от кроны дерева, которое затем фотограмметрически обрабатывается для получения ЦМР.

3) Etopo2

Самый нераспространенный формат, потому что является самым неточным из представленных ранее. Его неточность связана с ограничением аппаратных и программных возможностей на момент произведения съемок – 1978г. В последующем доработки не производились.

Результаты сравнения всех рассмотренных данных цифровой модели рельефа сведены в таблицу 1.

Таблица 1 – Результаты сравнения данных цифровой модели рельефа

|  |  |  |  |
| --- | --- | --- | --- |
|  | SRTM | AsterGDEM | Etopo2 |
| Формат данных | Hgt | GeoTIFF | GeoTIFF |
| Система координат | Разные | WGS84/EGM96 | WGS84 |
| Точность | Высокая | Хорошая | средняя |
| Доступность | 85°с.ш. и 85°ю.ш | 83°с.ш. и 83°ю.ш. | 60°с.ш. и 54°ю.ш |

В результате сравнения было принято решение об использовании SRTM в качестве данных цифровой модели рельефа. После того, как был выбран формат данных, выполняется генерация карты.

3.2 Генерация карты

В разделе, посвященному анализу работы сервера карт, было сказано, что графический интерфейс предоставляет возможность просмотра позиции тайла. Это возможно благодаря тому, что вся карта разбита на тайлы. Пример разбиения карты на тайлы представлен на рисунке 5.

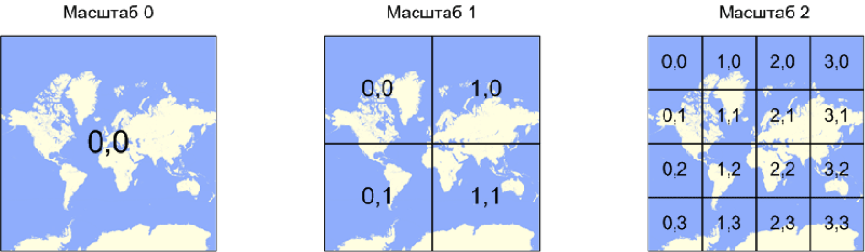


Рисунок 5 – Пример разбиения карты на тайлы

Следовательно, под генерацией карты для использования ее на сервере карт необходимо понимать создание тайлов карты.

3.2.1 Выбор масштаба

Перед тем, как приступить к генерации тайлов необходимо понимать, для какого или для каких масштабов необходимо генерировать тайлы.

Учитывая, что работа будет происходить с использованием фалов данных SRTM, то можно генерировать тайл для каждого значения высоты, то есть для каждых 3” создавать новый тайл. Это возможно выполнить, если масштаб будет равен 17. Достоинством использования такого уровня увеличения является высокая точность при декодировании. Недостатками такого увеличения является большое число тайлов (30 млрд), что требует большого объема памяти для хранения этих данных, а также большого времени для их генерации.

Кроме того, использование увеличение равного 17, довольно избыточно. Так как в результате будет генерироваться изображение 256х256, закрашенное только в один цвет и покрывающее область в 92 м. Пример сгенерированных тайлов представлен на рисунке 6.



Рисунок 6 – Пример сгенерированных тайлов

Можно уменьшить избыточность тайлов, если генерировать тайлы в которых 92м будут приходится на 1 пиксель. То есть каждое значение высоты будет помещаться в 1 пиксель. Это возможно для увеличения равного 11. При таком масштабе возможен компромисс между количеством сгенерированных тайлов и точностью последующего декодирования.

Уменьшение масштаба приведет к снижению точности декодирования. Так как при уменьшении масштаба происходит сжатие изображения и это приведет к «объединению» нескольких соседних пикселей в один.

Так как для выполнения поставленной задачи выводить карту высот для работы пользователю необязательно, а для определения высоты достаточно только увеличения 11. Следовательно, можно сгенерировать тайлы только для увеличения равного 11.

3.2.2 Генерация тайлов

Так как разбиение карты на тайлы является довольно стандартным решением для работы с картами, имеющими высокий уровень увеличения, то существуют приложения, в которых эта функция уже реализована. Как правило, такие приложения уже содержат набор готовых карт. Следовательно, среди этого набора может оказаться и карта высот.

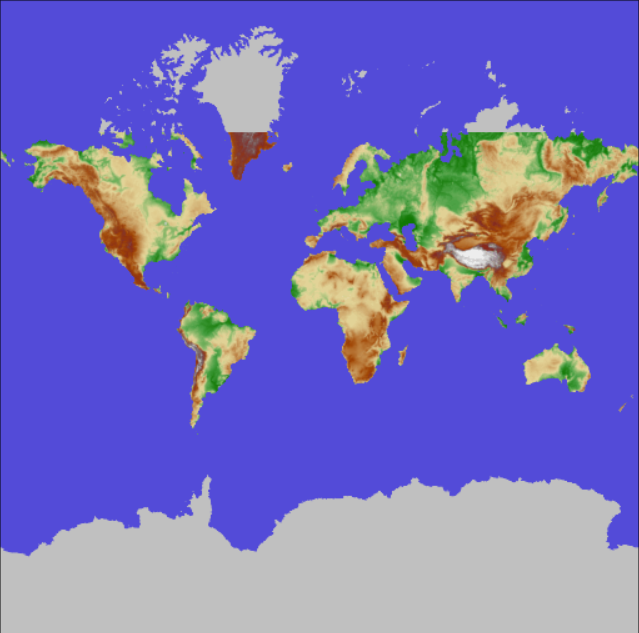
В результате поиска были обнаружены 2 приложения, отвечающие заявленным требованиям (возможность разбиения на тайлы, наличие карты высот): SasPlanet, QGis. Стоит отметить, что оба эти приложения в качестве данных для карты высот используют SRTM.

Далее был выполнен анализ карт высот, которые доступны в приложении. Как итог, карта из приложения SasPlanet не подходит для решения поставленной задачи по 2-м причинам:

1) карта является не полной, так как использует данные от первой версии SRTM.

2) нет сведений о соотношениях между цветом и высотой, а также о методах кодирования высоты цветом.

На рисунке 6 представлена карта, доступная в SasPlanet.

  
Рисунок 6 - Карта, доступная в SasPlanet

Что касается карты высот, доступной в приложении QGis, то она имеет данные для всей области карты, а также позволяет задавать значения цвета для высот и метод кодирования. Окно выбора цвета для высот и метода кодирования представлены на рисунке 7.

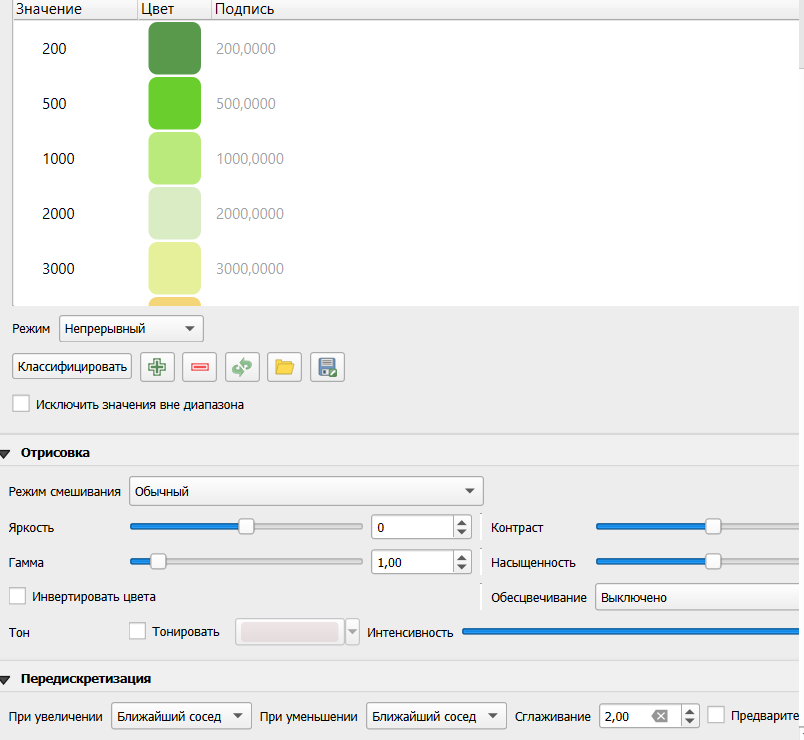


Рисунок 7 – Окно выбора цвета для высот и метода кодирования

В качестве метода кодирования был использован метод ближайшего соседа.

После того, как заданы цвета для высот и выбран метод кодирования, происходит автоматическое раскрашивание карты. Результат приведен на рисунке 8.

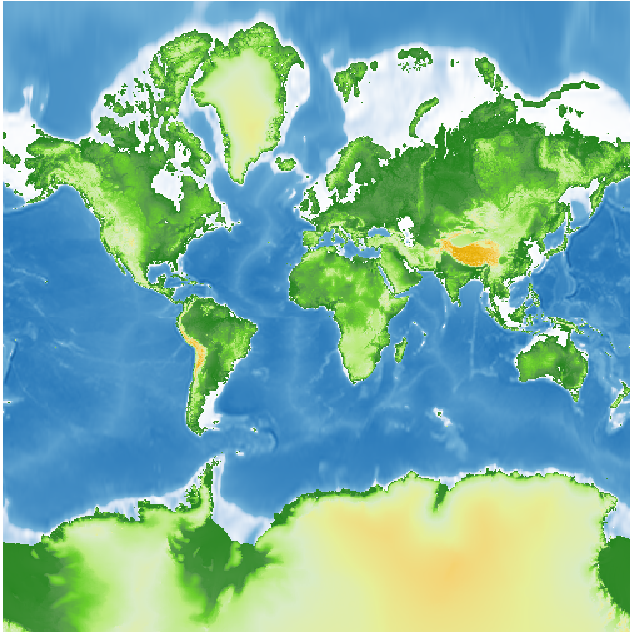


Рисунок 8 – Внешний вид карты после раскраски

После того, как карта готова, можно приступать к разбиению её на тайлы. Для этого, как уже было сказано ранее, в приложении уже есть функция. При выполнении этой функции начинается генерация 4 194 304 тайла. Каждый тайл генерируется, примерно, 1 секунду. Следовательно, для полного выполнения генерации потребуется 48 дней безостановочной работы. Это довольно много.

Можно попробовать уменьшить количество времени для генерации тайлов, если убрать из генерации тайлы, относящиеся к водному пространству, а потом учесть это при разработке приложения.

К сожалению, QGIS не предоставляет возможность генерации тайлов для некоторой области доступной карты, поэтому необходимо искать альтернативные подходы для решения задачи.

Можно попробовать генерировать тайлы программно. То есть написать скрипт, который бы читал данные из hgt файла, кодировал бы их цветом, вычислял по координатам позицию тайла и пикселя и закрашивал бы этот пиксель в полученный цвет.

Предположим, что обработку каждого пикселя потребуется 1 секунда. Так как тайл имеет размер 256х256, то для полной его генерации потребуется 18 часов. Даже если сделать параллельную обработку 6 тайлов на 6 процессорах, то все равно очень долго. Этот вариант не подходит.

Хотя в QGIS нет возможности сгенерировать тайлы для некоторой области, он позволяет выполнять разбиение на тайлы слоя, который можно создать из файла, содержащих данные высот в допустимом формате. Одним из таких допустимых форматов является hgt файл.

Добавить все имеющиеся файлы высот в QGIS в качестве новых слоев для последующего создания тайлов для каждого из них не предоставляется возможным из-за большого объема этих файлов – 70Гб. При попытке загрузить возникает зависание программы QGIS с последующим завершением работы программы.

Если не получается добавлять все разом, можно добавлять последовательно каждый файл, удаляя предыдущий. Но тут возникает ошибка при создании тайлов: файлы содержат информация для области 1°х1°, а тайл для масштаба 11 захватывает область размером больше, чем 1°х1°. Для решения этой проблемы достаточно добавлять дополнительные файлы, находящие вокруг необходимого файла. Пример представлен на рисунке 9.

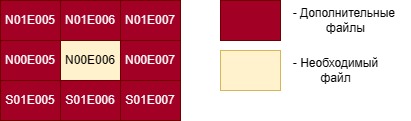


Рисунок 9 – Пример…

Стоит отметить, что в этом способе генерации карты используется кодирование по методу ближайшего соседа.

Так как добавление, удаление файлов, выбор слоя с последующим разбиением его на тайлы это рутинные задачи, их можно автоматизировать, написав скрипт на пайтоне, который будет выполнять все эти функции. Листинг кода этого скрипта приведен в приложении А.

При таком методе генерации тайла, каждый файл обрабатывается 2 мин. Но из-за того, что таких файлов всего 26 тысяч, то для полной генерации карты потребуется 36 суток безостановочной работы.

Это всё-таки меньше, чем при генерации тайлов для всей области карты. Следовательно, это решение нас устраивает.

3.3 Переименование файлов

После того, как все тайлы сгенерированы их необходимо переименовать и переместить в подкаталоги, которые необходимо создать. Это нужно сделать, потому что сервер карт может создавать карту из тайлов только в том случае, если эти тайлы находятся в нужных директориях, иначе он их не сможет обнаружить.

Для примера рассмотрена директория некоторого файла, созданная в QGIS.

…/656/1114.png

656 – координата тайла по x

1114 – координата тайла по y

Этот же файл, но допустимый для сервера карт будет иметь следующую запись:

…/0/x656/1/y1114.png

0 – координата тайла по х деленная на 1024

x656 - координата тайла по х

1 - координата тайла по y деленная на 1024

y1114 - координата тайла по y

Создание папок, переименование файлов и перенесение их в соответствующие папки – это довольно рутинная задача, поэтому можно прибегнуть к написанию скрипта на пайтоне. Листинг кода приведен в приложении Б.

4. Разработка приложения

В результате разбора последовательности действий для построения радиотрассы была выявлена необходимость в получении значении высоты для каждой точки маршрута радиотрассы. В ходе выполнения анализа работы сервера карт было принято решение о возможности использования изображений, возвращаемых сервером, в качестве источника данных высот при правильном кодировании последних. Теперь же необходимо разобраться в том, возможно ли выполнение запроса с конкретной координатой, принадлежащей маршруту радиотрассы, и получение тайла, для этой координатой. Для этого необходимо вновь обратится к схеме взаимодействия приложения с сервером, представленной на рисунке 10.

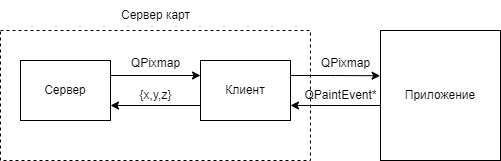


Рисунок 10 – Схема взаимодействия приложения с сервером карт

При определенных взаимодействиях с виджетом: прокрутка колеса мыши, удержание ПКМ и перемещение курсора, возникает событие об изменении содержимого виджета. Параметры для этих событий содержатся внутри класса QPaintEvent, экземпляр которого передается приложением клиенту. То есть в случае возникновения какого-то из событий возникает запрос к клиенту, информирующий о необходимости получения тайлов.

В свою очередь клиент вычисляет координаты тайлов, которые необходимо запросить и делает запрос на сервер.

Формирование запроса от приложения при возникновении таких событий, как прокрутка колеса мыши, удержание ПКМ и перемещение курсора, нам не подходит. Нам нужно, чтобы запрос был сформирован после того, как рассчитан маршрут радиотрассы, то есть после того, как выбраны 2 точки объектов. Следовательно, стоит переписать формирование запроса, но для того, чтобы его переписать надо понимать всю последовательность действий для его формирования. Для этого необходимо выполнить детализацию структуры клиента.

4.1 Последовательность формирования запроса

Под детализацией структуры клиента следует понимать рассмотрение последовательности формирования запроса и получения ответа. Для наглядности была разработана диаграмма последовательностей, представленная на рисунке 11.

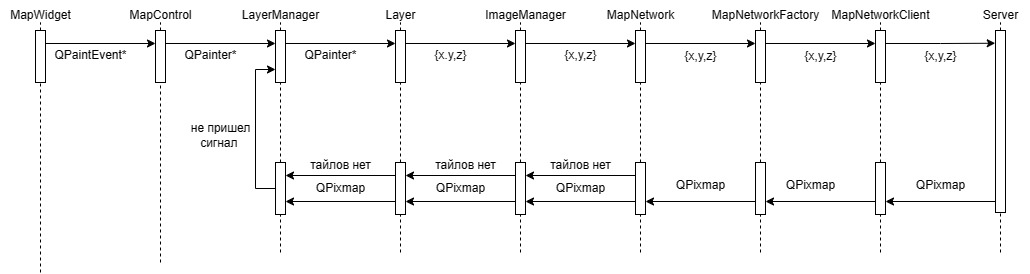


Рисунок 11 – Диаграмма последовательностей

При первом запросе от приложения мы настраиваем взаимодействие с клиентом. Делается это при помощи передачи ссылки на экземпляр класса QPaintEvent, который содержит в себе параметры для событий рисования и принадлежит приложению.

В свою очередь события рисования отправляются виджетам, которым необходимо сделать перерисовку содержимого. Например, при передвижении карты.

mapControl связывает класс QWidget с экземпляром класса QPaintEvent. Это нужно для того, чтобы любые изменения на виджете, сразу обрабатывалась QPaintEvent. Но так как основная задача – отрисовка карты, то mapControl передает управление LayerManager, передав в качестве параметра ссылку на объект класса QPainter, занимающийся отрисовкой объектов. И который необходим для отрисовки области карты

layerManager необходим, потому что карта может представлять собой несколько слоев. Например, гибридная карта создает при помощи наложения osm-карты, где обозначены границы государств и подписаны названия городов, на спутниковую карту. И для правильной отрисовки такой карты просто необходим layerManager.

При принятии управления layerManager передает управление Layer, делая это для каждого слоя. Передавая в качестве параметров ссылку на еще один экземпляр класса QPainter, который необходим для отрисовки одного изображения, и координаты центра карты.

Кроме того, layerManager соединяет все тайлы, создавая тем самым область карты и отрисовывает ее поверх виджета при помощи Qpainter, который был передан от mapControl.

При получении управления layer вычисляет значения позиции тайла через смещения. Схема алгоритма определения позиции тайла через смещение представлена на рисунке 12.

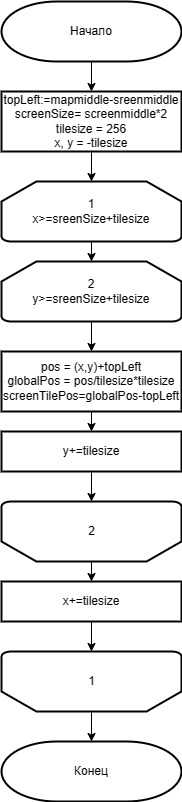


Рисунок 12 – Схема алгоритма определения позиции тайла через смещение

Далее для всех вычисленных позиций идет проверка наличия тайлов в кэше. Если есть то, тайл извлекается из кэша, отрисовывается и отсылается. Если же нет, то происходит передача управления ImageManager.

В свою очередь imageManager делает запрос на сервер, указывая в качестве параметров позицию тайла и увеличение.

При получении ответа от сервера получаем изображение вместе с его координатами, сохраняем его, а потом перекидываем сигнал, что изображение загружено. Его ловит layermanager, который занимается тем, что запускает метод перерисовки, тем самым запрашивая следующий тайл.

И так делается до тех пор, пока количество тайлов в очереди к серверу не станет равным 0. Когда очередь тайлов пуста посылается сигнал об окончании. Его снова ловит layermanager. Процесс запроса тайлов приостанавливается до следующего события о перерисовке от виджета.

4.2 Программная реализация дополнительной клиентской части

В результате выполнения детализации клиентской части сервера карт стало очевидно, что существующая клиентская часть подходит для задачи отображения области карты для предоставления пользователю возможности выбора точек, где будут располагаться объекты, но не походит для задачи получения высоты для координаты, так как не имеет алгоритма для преобразования координаты в позицию тайла, а также не имеет возможности сохранения позиции тайла для последующей фильтрации изображений, полученных от сервера. Следовательно, необходимо переписать некоторые части клиентской части для того, чтобы они могли решать поставленную задачу.

Для начала в классе ImageManager необходимо добавить метод, который бы реализовывал сохранение позиций запрошенных тайлов для последующего выделения нужного тайла путем сравнения сохраненных значений координат тайла с теми, которые пришли вместе с изображением. Кроме того, необходимо добавить метод, реализующий обработку пустых изображений изображений. Для того, чтобы сделать добавление этих методов необходимо создать класс, являющийся наследником от ImageManager, в котором и реализовать эти методы.

Далее необходимо изменить алгоритм определения позиции тайла в классе layer, чтобы он мог работать с географическими координатами.

Алгоритм преобразования координат в позицию тайла заключается в преобразовании географических координат сначала в меркаторские, а затем в пиксели.

Преобразование в меркаторские осуществляется по формуле. Формула зависит от вида проекции. Вид проекции в свою очередь зависит от того, как представляют Землю: в виде сферы, в виде эллипса или в виде геоида. Существует еще одна проекция, в которой Землю представляют в виде сферы, но в которой упрощена формула расчета – web Mercator. Именно ей мы и воспользуемся.

Формула для расчета следующая:

qreal y = (1 - (log(tan(PI / 4 + deg\_rad(lat) / 2)) / PI)) / 2 \* (numberOfTiles \* mytilesize);

qreal x = (lon + 180) \* (numberOfTiles \* mytilesize) / 360.

Схема алгоритма для преобразования географических координат в пиксели представлена на рисунке 13.

Нужно ли ее вообще??? Если это просто применение формулы

Для того, чтобы определить позицию тайла необходимо полученные значения пикселей разделить нацело на 256, потому что размер тайла 256х256.

Получается, что в методе класса layer, который выполнял обращение к ImageManager необходимо изменить функция расчета позиции тайла, добавить вызов метода сохранения позиции тайла у ImageManager. Кроме того, для обращения ко всем методам необходимо создать экземпляр класса наследника ImageManager. Следовательно, для класса Layer также необходимо создавать новый дочерний класс, в котором выполнять переопределение метода.

Так как LayerManager взаимодействует с методом Layer, который был переопределен, необходимо переопределять и метод LayerManager. Раз происходит изменение метода, то из него можно убрать отрисовку изображения при помощи QPainter, так как для задачи этого не требуется.

По той же причине, по которой было выполнено переопределение layerManager, необходимо выполнить наследование класса mapControl, в котором выполнить переопределение метода обращения к классу Layermanager.

Диаграмма классов представлена на рисунке 14.

4.3 Проверка работы

В главе, посвященной анализу работы сервера карт, было упомянуто, что клиентская часть имеет графический интерфейс, позволяющий просматривать позиции тайла. Пришло время воспользоваться этой функцией для проверки корректности алгоритма преобразования географических координат в позицию тайла, а также корректности формирования запроса и получения ответа.

Проверка заключается в следующем: в программе задаются какие-то географические координаты; далее они преобразуются в запрос на сервер, проходя через всю последовательность действий; сервер в качестве результата вернет изображение, которое можно сохранить, как x\_y.jpg, где x – координата тайла по x, а y – координата тайла по y. После этого можно сравнить координаты тайла, указанные в клиентской части, с теми которые получились, также можно визуально сравнить изображения на предмет сходства.

Пример выполнения проверки представлен на рисунке 15.

Рисунок 15 – Пример выполнения проверки

4.4 Программная реализация алгоритмов

В конечном итоге приложение должно выполнять расчеты для следующих функций:

1. Расчет радиотрассы
2. Декодирование высоты по значению цвета

Если в случае расчета радиотрассы алгоритм было подробно описан в пункте 1.2 ПЗ, и поэтому повторное описание будет упущено в этом разделе.

Что касается декодирование высоты по значению цвета, то тут необходимо учитывать, что для кодирования по методу ближайшего соседа. Этот метод подразумевает, что выбираются несколько значений высот, которые будут являться границами. Для каждого такого значения задается цвет. Далее считывается значение высоты из файла. Для того, чтобы вычислить значение цвета, соответствующей этой высоте, необходимо выполнить следующие действия:

1) определить между какими заданными границами высотами располагается полученная высота

2) вычислить коэффициент «отдаленности»:

p = (h-h1)/(h2-h1), где

h – полученная высота;

h1–нижняя граница интервала высот;  
 h2 – верхняя граница интервала высот;

3) вычислить значения цвета

r = (1-p)\*r1+p\*r2

g = (1-p)\*g1+p\*g2

b = (1-p)\*b1+p\*b2

r2, g2, b2 – цвет для верхней границы интервала высот

r1, g1, b1 – цвет для нижней границы интервала высот

Следовательно, для декодирования нужно выполнить последовательность действий в обратном порядке.

Программная реализация обоих алгоритмов не будет приведена в ПЗ из-за простоты алгоритмов.

4.5 Интерфейс

При разработке интерфейса был произведен обзор интерфейса нескольких приложений и сайтов для расчета радиотрассы с целью выявить наиболее понятный интерфейс.

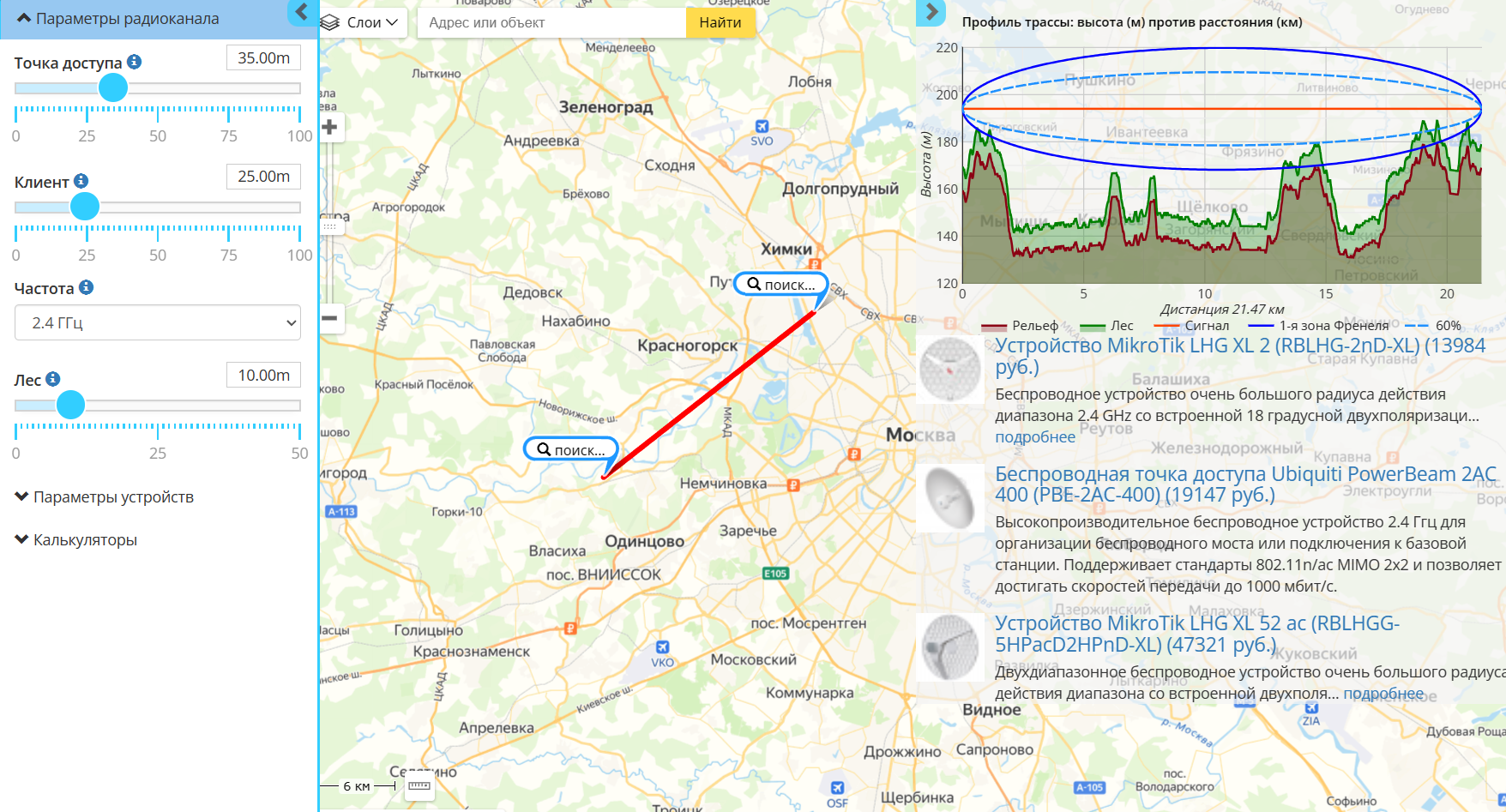
В качестве критерия непонятности интерфейса оценивалось количество вопросов, обучающих статей, посвященных тому, как выполнить расчет радиотрассы.

Для анализа были выбраны:

- сайт linktest.ru

- приложение radio mobile

Интерфейс сайта представлен на рисунке 16



Интерфейс приложения radio mobile приведен на рисунке 17

Сравнение их интерфейсов сведено в таблицу 2.

Таблица 2 – Сравнение интерфейсов

В результате сравнения было принято решение о том, что интерфейс приложения будет содержать главное окно, в котором будет отображаться карта; боковую панель, на которой будет выводиться результат построения радиотрассы, а также на ней будут выводиться координаты источника и приемника с высотами антенн, которые можно менять по возможности.

Дополнительно была реализована возможность не только отображения координат объекта, но и их ввода.

Интерфейс разработанного приложения представлен на рисунке 18

Рисунок 18 – Интерфейс разработанного приложения