

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ – ПРОЦЕССОВ УПРАВЛЕНИЯ
КАФЕДРА ТЕХНОЛОГИИ ПРОГРАММИРОВАНИЯ

Степанов Сергей Владимирович

Выпускная квалификационная работа бакалавра

**Оптимизация планов капитального строительства
линейных объектов
на базе многофакторного анализа
с учетом геологических ограничений**

Направление 02.03.02

"Фундаментальная информатика и информационные технологии"

ООП "Программирование и информационные технологии"

Научный руководитель,
кандидат физ.-мат. наук,
доцент

Просолупов Е. В.

Рецензент,
руководитель научной лаборатории Digital Design

Ашихмин И. А.

Санкт-Петербург
2018

Содержание

Введение	4
Постановка задачи	6
Обзор литературы	9
Глава 1. Метрики	11
1.1 Цифровая модель рельефа	11
1.2 Измерение расстояний	12
1.3 Вычисление стоимости пути	14
1.4 Вычисление стоимости строительства в точке	14
Глава 2. Метод поиска кратчайшего пути	16
Глава 3. Генерация графа	18
3.1 Построение вычислительной сетки	18
3.2 Получение точек	19
3.3 Выбор модельных карт местности	20
3.4 Проведение измерений	22
3.5 Анализ результатов	27
Глава 4. Построение оптимальной сети	29
4.1 Описание используемого алгоритма	29
4.2 Описание модельных карт для тестирования	33
Выводы	37
Заключение	38
Список литературы	39

Приложение	42
Приложение А	42

Введение

В современном мире каждая компания хочет минимизировать свои затраты, как по времени, так и в финансовом плане. Это не обошло и сферу строительства. В данной области существует ряд открытых задач, требующих решения, в том числе это касается построения оптимальных траекторий линейных объектов.

Объектами капитального строительства в данной работе называются площадные объекты, такие как здания, строения, сооружения, и линейные объекты, такие как линии электропередачи, линии связи, трубопроводы, автомобильные дороги, железнодорожные линии и другие подобные сооружения [1].

Во время строительства линейного объекта человеку приходится сталкиваться с преодолением естественных препятствий, таких как леса, болота, реки, а также препятствий, представляющих собой разного рода сооружения и области, на которых строительство запрещено по юридическим и иным причинам. Эти ограничения на строительство называются инженерно-геологическими условиями и включают в себе несколько составляющих, например:

- геологическое строение местности;
- рельеф;
- гидрогеологические условия;
- мерзлотные условия;
- современные геологические процессы [2].

Помимо нахождения оптимальной траектории линейного объекта, соединяющего две точки, возникает проблема проектирования набора оптимальных трасс линейных объектов для сети, соединяющих большее количество точек. Кроме построения оптимальной сети линейных объектов, учиты-

вая только затраты на строительство, имеет смысл проектировать сеть линейных объектов, опираясь также на затраты по её эксплуатации и ряд других факторов, так как именно поддержание инфраструктуры в работоспособном состоянии может отнимать наибольшее количество средств.

Постановка задачи

Задача построения оптимальных трасс линейных объектов выглядит следующим образом. Пусть задан определённый участок поверхности Земли вместе с соответствующим ему рельефом. Также на этом участке присутствуют области различных типов местности, такие как леса, болота, реки и т.п., с заданной для каждого типа местности стоимостью строительства и имеющий вид многоугольников. Помимо этого присутствуют уже существующие линейные объекты, такие как дороги, трубопроводы и т.п., которые можно представить в виде ломаных линий с заданной стоимостью строительства на них. И также задан набор объектов, представляющих собой точки земной поверхности, которые нужно связать оптимальной сетью линейных объектов.

В математическом виде задачу можно представить так: задана поверхность в трёхмерном пространстве \hat{S} , описывающая земной рельеф, также задано множество объектов $\hat{P} \in \hat{S}$. Требуется связать множество объектов \hat{P} коммуникационной сетью так, чтобы затраты на строительство данной сети были минимальны.

Обозначим за область расчетов Q проекцию поверхности \hat{S} на плоскость xOy . При проектировании \hat{P} на Q , будет получено множество координат точек $P = \{(x_i, y_i) \in Q, i = \overline{1..n}\}$. Помимо области проектирования и объектов также заданы множество областей $A = \{\{x_j, y_j\}_{j=1}^{n_i} \in Q, i = \overline{1..m}\}$ в виде произвольных многоугольников и множество существующих линейных объектов $L = \{\{x_j, y_j\}_{j=1}^{n_i} \in Q, i = \overline{1..k}\}$ в виде ломаных линий. Основываясь на стоимостях строительства в областях A и вдоль линейных объектов L задана функция строительных затрат $c : \{Q \rightarrow \mathbb{R}, \forall(x, y) \in Q : c(x, y) \geq 0\}$, обозначающая стоимость строительства в каждой конкретной точке плоскости Q , а также функция задающая высоту точки $h : \{Q \rightarrow \mathbb{R}, \forall(x, y) \in Q\}$.

Целью данной работы является написание программы в результате работы которой должна быть построена сеть линейных объектов, представляю-

щая собой набор ломаных линий $N = \{\{x_j, y_j\}_{j=1}^{n_i} \in Q, i = \overline{1..s}\}$ оптимальная по заданным параметрам.

Для решения необходимо выполнить следующее:

1. Определить способ вычисления стоимости строительства линейного объекта.
2. Проанализировать основные алгоритмы построения пути между двумя объектами и реализовать оптимальный.
3. Проанализировать основные алгоритмы построения сети линейных объектов и реализовать оптимальный.
4. Исследовать различные случаи для тестирования работы программы.

Актуальность проблемы

При проектировании линейного объекта инженер сталкивается с проблемой правильности составления стоимостной оценки проекта. Область проектирования представляет собой неоднородную территорию, на которой удельные затраты на строительство линейного объекта в различных ее точках различны. Значения удельных строительных затрат в значительной мере определяются типом грунтов различных категорий: болото, суходол, лес, вечномерзлые грунты, речная пойма и т.д. Помимо разного вида грунтов нужно также учитывать уже существующую инфраструктуру, ранее созданную человеком. Например, прокладка автодороги между двумя объектами с учетом уже существующей дороги, зачастую, окажется дешевле, чем если не учитывать построенную дорогу.

Но помимо удешевления затрат от использования инфраструктуры существует ряд ограничений, которые могут существенно увеличить стоимость строительства. В качестве примера рассмотрим проектирование высоковольтной линии. Для каждой из них предусмотрена санитарно-защитная зона, определяющая минимальное расстояние до ближайших жилых, производственных

и непроизводственных зданий и сооружений. И для различных классов линейных объектов есть свои строительные нормы и правила (СНиП), которые нужно учитывать в обязательном порядке. В том числе в них прописаны допустимые перепады высот на линейном объекте, что заставляет инженера использовать данные о рельефе местности.

Если список ограничений на строительство одного линейного объекта вполне поддается анализу, то при проектировании сети линейных объектов список факторов, влияющих на размещение точек разветвления, то есть оптимального расположения перекрестков, становится достаточно большим, и однозначно обосновать правильность места расположения точки разветвления является нетривиальной задачей. В качестве примера рассмотрим задачу проектирования сети трубопроводов для месторождения, когда у каждой скважины есть график выработки и график выхода на проектную мощность. Стоимость строительства трубопровода зависит от диаметра трубы, а также от поддерживающей инфраструктуры месторождения, как электрические подстанции, объема используемой воды. И если не учитывать стоимость эксплуатации при проектировании месторождения, то расчетные местоположения точек разветвления могут показать себя крайне неэффективными [3].

Исходя из всего этого, проблема является актуальной, так как рынку требуется, как просто инструмент для анализа стоимости строительства уже спроектированных сетей линейных объектов, так и создание системы, которая сможет строить сети линейных объектов в автоматическом режиме, учитывая ряд факторов, как эксплуатационные расходы, расходы на транспортировку строительных материалов, а также различные СНиП и требования законодательства.

Обзор литературы

Проблема соединения нескольких точек лежащих на плоскости системой дорог наименьшей суммарной длины таким образом, что из каждой точки можно было добраться в любую другую возникла еще очень давно. Её постановка для случая $n = 3$ была предложена математиком П. Ферма, а первое решение было получено Б. Кавальери и Э. Торричелли. Данную задачу можно считать одной из старейших оптимизационных проблем в математике. В 1934 году В. Ярник и О. Кесслер обобщили эту задачу для произвольных n точек. А в 1941 году вышла книга "Что такое математика?" в которой авторы называют эти две задачи проблемой Штейнера в честь математика Я. Штейнера, первым получившим чисто геометрическое решение задачи Ферма. Книга была достаточно популярной, поэтому название задачи закрепилось за именем Штейнера [4].

Если в ранние годы задача формулировалась строго математически, то двадцатый век принес её в прикладную область. В настоящее время можно выделить несколько различных постановок задачи Штейнера, для каждой из которых используются свои методы решения. В классической постановке задачи используется понятие Евклидовой плоскости, а, соответственно, и Евклидовой метрики. Она может быть интересна в теоретическом плане. Вторая постановка задачи предполагает использование прямоугольной метрики, что позволяет снизить затраты при проектировании сверх-больших интегральных схем. Третья постановка задачи базируется на теории графов, и она является наиболее широкой в применении. С помощью решения задачи в этой постановке можно повысить эффективность проектирования коммуникационных, электрических и механических сетей. Задача Штейнера является NP-полной, и к настоящему моменту не существует алгоритмов для любой из постановок задачи, способных решить её за полиномиальное время [5].

Решение данной задачи до сих пор является актуальным, что подтвер-

ждают работы, как российских, так и зарубежных ученых. Например, для неориентированных графов данная задача рассматривалась Хакими в 1971 году, Дрейфусом и Вагнером в 1972 году и Такахаши и Матцуямой в 1980 году. Данные алгоритмы являются вычислительно неэффективными, хотя они и позволяют находить точное решение [6]. В 1980 году Д.Т. Лотарев сталкивался с задачей проектирования транспортной сети, но в своей работе он рассматривал задачу с учетом потоков в графе [7].

Также помимо публикаций отдельных научных работ в 2014 году проходила конференция-соревнование DIMACS-11, на которой участники соревновались в скорости работы их алгоритмов для решения задачи Штейнера в различных её постановках [8].

Глава 1. Метрики

В связи с тем, что мы работаем над созданием системы, которая будет использоваться в реальном мире, то нужно понять, каким образом можно представлять данные, используемые для решения задачи. Так как в нашем продукте при строительстве линейного объекта должен учитываться рельеф, то каждую точку земной поверхности можно представить в виде:

$$p = \{lat, long, height\},$$

где *lat* - широта точки, *long* - долгота точки, *height* - геодезическая высота. Для получения высоты точки на поверхности Земли используются цифровые модели рельефа.

1.1 Цифровая модель рельефа

Цифровая модель рельефа — это математическое представление участка земной поверхности, полученное путём обработки материалов топографической съёмки. Представляет собой информацию о высоте поверхности земли без учёта растительности, зданий и других высотных объектов, которые на ней находятся [9].

Существуют разные источники данных цифровой модели местности. У каждого из них есть свои преимущества и недостатки (см. табл. 1).

Источник	Разрешение	Ограничение	Стоимость
SRTM [10]	90 м	Имеются данные высот только между 60° с.ш. и 54° ю.ш.	Бесплатно
ASTER GDEM [11]	15 м	Имеются достаточно большие шумы, ухудшающие возможность практического использования. Данные высот между 83° с.ш. и 83° ю.ш.	Бесплатно
GMTED2010 [12]	250 м	Имеются данные высот для всей поверхности планеты, но в достаточно низком разрешении.	Бесплатно
WorldDEM [13]	12 м	Без ограничений. Есть данные высот для всей территории планеты.	Платно

Таблица 1. Примеры источников данных цифровой модели местности

В данной работе будем использовать данные SRTM, так как исследуемые территории попадают в указанные ограничения по широтам, и точности

данной цифровой модели рельефа достаточно для наших целей. Помимо этого для языка программирования Python существует удобный модуль SRTM.py, служащий для получения высоты точки, заданной географическими координатами. В остальных случаях требуется скачивание и дополнительная подготовка данных цифровых моделей рельефа.

1.2 Измерение расстояний

Фигура Земли имеет форму геоида. Геоид — выпуклая замкнутая поверхность, примерно совпадающая с поверхностью воды в морях и океанах в спокойном состоянии и перпендикулярная к направлению силы тяжести в любой её точке [14].

Земной эллипсоид — эллипсоид вращения, размеры которого подбираются при условии наилучшего соответствия фигуре квазигеоида для Земли в целом или отдельных её частей [15]. Все координаты и дистанции вычисляются именно на эллипсоиде.

Референц-эллипсоид — приближение формы поверхности Земли эллипсоидом вращения, используемое для нужд геодезии на некотором участке земной поверхности [16].

Нахождение расстояние между двумя точками — это решение обратной геодезической задачи. Есть несколько способов измерения расстояния между двумя точками, лежащими на земной поверхности:

- Вместо использования географической системы координат, можно использовать метрическую систему координат, например, Universal Transverse Mercator (UTM). Это позволит вычислять расстояние между точками, используя Евклидову метрику. В случае использования данного способа требуется подбирать датум, то есть набор параметров, используемых для смещения и трансформации референц-эллипса в локальные географические координаты с целью снижения ошибки в расчетах, для каждой конкретной области, так как при использовании датума для общезем-

ного эллипсоида WGS84 относительная ошибка измерения может превышать 100% при измерении расстояний в областях близких к полярным широтам.

- Использование метрического тензора $S = \int \sqrt{g_{\alpha\beta}(x)dx_\alpha dx_\beta}$, где $g = \begin{bmatrix} R^2 & 0 \\ 0 & R^2 \sin^2 \theta \end{bmatrix}$. Данный способ хорошо работает только на достаточно близких расстояниях, а вот на дистанции около 10 км и больше данный метод дает погрешность более 2%.
- Формула гаверсинусов [17]. При использовании данной формулы мы пользуемся моделью Земли в виде сферы и с помощью тригонометрических преобразований находим кратчайшее расстояние между точками. Данный способ дает погрешность порядка 0.3%.
- Формула Винценти [18]. При использовании данной формулы мы представляем фигуру Земли в форме эллипсоида и вычисляем расстояние с помощью итерационной формулы. Данная формула является наиболее точной для вычисления расстояния и дает погрешность около в 1 мм вне зависимости от удаленности точек.

Для выбора оптимального способа, нужно учитывать два параметра: время вычисления и точность измерения. Первые два способа мы не будем использовать в связи с отсутствием универсальности, а для третьего и четвертого способа проведем замеры производительности. Создадим тестовую выборку из десяти тысяч точек со случайно выбранными координатами и проведем измерение расстояний для каждой пары точек двумя разными способами. В среднем время вычисления расстояния третьим способом в 4 раза меньше, чем четвертым, и составляет $4 \cdot 10^{-6}$ сек против $16 \cdot 10^{-6}$ сек для каждой пары точек. Таким образом, оптимальным выбором является вычисление расстояния с помощью формулы гаверсинусов. Но все формулы представленные выше не учитывают рельеф, поэтому при окончательном расчете длины пути потребуется добавить также и перепады высот между двумя точками.

1.3 Вычисление стоимости пути

Для того чтобы оптимизировать расположение линейного объекта требуется уметь вычислять его стоимость строительства. Каждый линейный объект можно представить в виде состоящей из отрезков ломаной $L = \{\{A_i, B_i\}_{i=0}^n$, где $A_i, B_i \in Q\}$. Стоимость строительства каждого прямого отрезка AB_i можно представить в виде значения функции C_{AB_i} , где каждый отрезок задается параметрически:

$$AB_i: \begin{cases} x = x(t), \\ h = h(x(t)), \quad t \in [0, 1], \text{ где } [0, 1] - \text{отрезок параметризации.} \\ c = c(x(t)) \end{cases}$$

$$C_{AB_i} = \int_{AB_i} c(x(t)) \sqrt{\left[g_{\alpha\beta}(x(t)) + \frac{\partial h}{\partial x_\alpha} \cdot \frac{\partial h}{\partial x_\beta} \right] \dot{x}_\alpha \dot{x}_\beta dt},$$

где g — метрический тензор поверхности Земли без учета рельефа.

Таким образом стоимость строительства линейного объекта можно представить в виде суммы $C_L = \sum_{i=0}^n C_{AB_i}$, а стоимость строительства сети линейных объектов, как $C_N = \sum_{j=0}^m C_{L_j}$. Общее решение задачи сводится к поиску минимальной по стоимости сети среди всех построенных сетей линейных объектов $N_{res} = \underset{N}{\operatorname{argmin}}(C_N)$.

1.4 Вычисление стоимости строительства в точке

Так как по условию задачи все области и линейные объекты заданы в виде набора вершин, лежащих на плоскости Q , то функцию стоимости строительства в точке можно определить через принадлежность точки определенному многоугольнику. У любого существующего линейного объекта есть ширина, поэтому при начальной инициализации алгоритма следует провести операцию по превращению линейного объекта в многоугольник.

Зачастую данные, получаемые на вход программой, являются несовершенными и содержат взаимопересечения, как линейных объектов, так и областей. Поэтому функцию строительства в точке определим, как максимум

мальное значение стоимости, среди всех областей, в которых лежит точка. И если точка лежит в области, принадлежащей линейному объекту, то стоимость строительства в этой точке будет рассчитываться, как минимальная стоимость среди всех линейных объектов, в которых лежит точка. То есть формально:

$$\begin{cases} \min(L_{cost}), & \text{если точка лежит на линейном объекте;} \\ \max(A_{cost}), & \text{иначе.} \end{cases}$$

Существует два подхода для вычисления стоимости строительства в каждой точке области Q : используя векторные данные и используя растр.

Векторный способ получения стоимости строительства в точке предполагает проверку принадлежности каждой точки всем многоугольникам, которые лежат в области проектирования Q . Основными преимуществами данного способа является очень высокая точность определения стоимости относительно координат точки и малый объем занимаемой памяти. Главным минусом является низкое быстродействие, которое можно повысить, используя алгоритмы на основе пространственной сетки [19].

Вторым способом получения стоимости строительства в точке, является использование растра. Растр — это матрица, каждое значение которой представляет собой проекцию набора стоимости точек области проектирования Q , выбранных определенным способом. Из главных плюсов этого способа хранения — это быстродействие, ведь для того чтобы получить значение стоимости в точке, нужно лишь вычислить индексы этой точки для матрицы растра. Но минусами этого способа являются низкая точность определения стоимости точки, так как тут возникают абсолютно те же проблемы, как и при представлении линии в растровом изображении на компьютере. А вторым недостатком является очень большой объем занимаемой памяти, так как чтобы нивелировать первую проблему, требуется повысить точность растра, увеличив число точек в нём.

Глава 2. Метод поиска кратчайшего пути

Существуют различные подходы для поиска оптимального пути между двумя объектами. Одним из способов решения данной задачи является её сведение к поиску кратчайшего пути на графе. Для этого требуется взять определенный набор точек P , лежащих в области построения Q , на их основе сгенерировать вычислительную сетку, которую можно преобразовать в граф. Каждое ребро получившегося графа является прямым отрезком, принадлежащим области Q , соответственно, его вес можно вычислять, как интегральную сумму, определение которой было дано в гл. 1. После того как график построен, можно пользоваться методами поиска кратчайшего пути.

Главным плюсом данного подхода является отсутствие локальных минимумов и то, что для каждого построенного графа найденный путь между точками будет являться кратчайшим. И не возникает такой проблемы, как обход области с высокой стоимостью строительства.

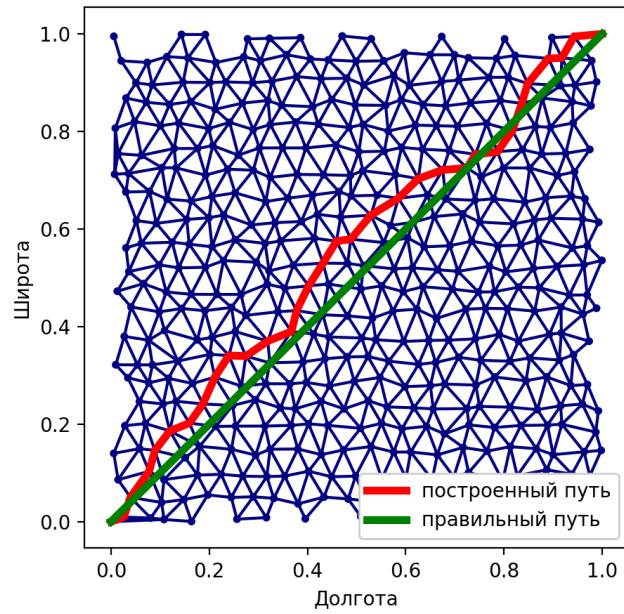


Рис. 1. Поиск пути на графике

Однако данный подход не лишен недостатков. Самая большая проблема — это точность получаемого решения, что можно увидеть на рис. 1. Оптимальный путь представляет собой прямую линию соединяющую два противоположных углов квадрата.

воположных угла квадрата, а путь, найденный на графе, представляет собой ломаную линию, лежащую, хоть и недалеко от оптимального решения, но и не совпадающую с ним. Поэтому остро стоит проблема поиска правильного метода генерации вычислительной сетки.

Существуют различные алгоритмы поиска кратчайшего пути на графе. Так как нашей задачей является нахождение пути с минимальной стоимостью между парой вершин без учета различного рода ограничений на число рёбер в построенном пути, было решено использовать алгоритм A* [20]. Основным его преимуществом является наличие эвристической функции, значительно уменьшающей время поиска пути. Только в случае нашей задачи поиск оптимальной эвристической функции для оценки расстояния между двумя вершинами нетривиален, в связи с неоднородностью стоимостной функции c и функцией высоты рельефа h .

Также решение, полученное на графике, является хорошим начальным приближением для дальнейшего расчёта, используя методы оптимизации.

Глава 3. Генерация графа

Для нахождения оптимального пути нам требуется построить планарный граф, связывающий заданные точки. Существует открытый вопрос, связанный со способом его построения.

3.1 Построение вычислительной сетки

Для чтобы построить график, требуется сгенерировать вычислительную сетку. Вычислительная (расчетная) сетка — набор точек (узлов сетки) в рассматриваемой области, соединенных в соответствующем порядке отрезками, образующими грани ячеек [21]. Есть несколько способов её генерации. Их можно разделить на два вида: равномерные и неравномерные.

Модель равномерной сетки описывает координаты отдельных точек поверхности следующим способом: каждому узлу сетки с индексами (i, j) отвечают определенные значения координат (x, y) такие, что расстояние между узлами одинаковое — dx по оси x , dy по оси y .

Неравномерной сеткой называется модель описания поверхности в виде множества отдельных точек $\{(x_0, y_0), \dots, (x_{n-1}, y_{n-1})\}$, принадлежащих поверхности.

Использование равномерной сетки не всегда будет давать качественный результат для поиска кратчайшего пути, поэтому необходимо опробовать алгоритмы поиска кратчайшего пути на неравномерной сетке. Наиболее универсальной является треугольная сетка, так как не возникает проблем построения треугольников по каким-либо имеющимся данным.

Для её построения воспользуемся триангуляцией Делоне. Триангуляция Делоне — триангуляция для заданного множества точек S на плоскости,

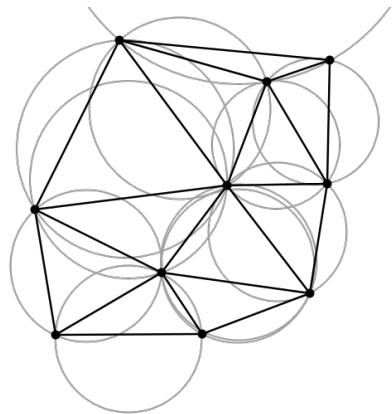


Рис. 2. Пример триангуляции Делоне

при которой для любого треугольника все точки из S за исключением точек, являющихся его вершинами, лежат вне окружности, описанной вокруг треугольника [22].

3.2 Получение точек

За получение координат точек сетки отвечает метод, логика работы которого реализована следующим образом. На вход ему подаются координаты точек, отвечающих за границы прямоугольника, лежащего на земной поверхности, внутри которого мы хотим найти кратчайший путь между парой точек. Результат работы метода зависит от двух параметров, которые задаются извне: d - размер сетки, γ - коэффициент случайного смещения точек упорядоченной сетки.

Далее мы вычисляем длины сторон заданного прямоугольника и делим их на размер сетки d , тем самым получая количество точек n и m , лежащих на каждой из сторон прямоугольника. После этого получаем набор точек с одинаковыми интервалами между ними, для каждой из сторон, таким образом получаем упорядоченную сетку. Но упорядоченная сетка не всегда хороша для поиска кратчайшего пути, поэтому за внесения смещения в финальный набор точек отвечает параметр $\gamma \in [0.0, 2.0]$. Таким образом из прямоугольной упорядоченной сетки мы получаем неупорядоченную. Итоговый набор точек получается по следующей формуле:

$$\begin{cases} \overrightarrow{M_x} = \overrightarrow{x} + \gamma \Delta_x \overrightarrow{\xi} \\ \overrightarrow{M_y} = \overrightarrow{y} + \gamma \Delta_y \overrightarrow{\xi} \end{cases}$$

где $\overrightarrow{M_x}, \overrightarrow{M_y}$ - итоговые координаты точек;

$\overrightarrow{x}, \overrightarrow{y}$ - координаты точек упорядоченной сетки;

Δ_x, Δ_y расстояние между соседними точками упорядоченной сетки по x и y ;

$\xi \in [-0.5, 0.5]$ - значение случайной величины.

Визуальные отличия сеток с различными значениями параметра γ представлены в Приложение А.

3.3 Выбор модельных карт местности

Для тестирования работы программы были выбраны следующие модельные карты местности:

1. Квадрат размером [1.0 x 1.0] градусов. Ищем путь между точками, лежащими в противоположных углах квадрата. Эталонная длина пути: 157252.787 м. Эта карта служит, как простейший случай для проверки правильности работы алгоритма поиска пути.



Рис. 3. Корректный путь для первой карты

2. Прямоугольник размером [1.0 x 0.1] градусов. Ищем путь между точками, лежащими в противоположных углах прямоугольника. Эталонная длина пути: 111751.882 м. Данная карта показывает некорректное построение пути, если использовать упорядоченную сетку.

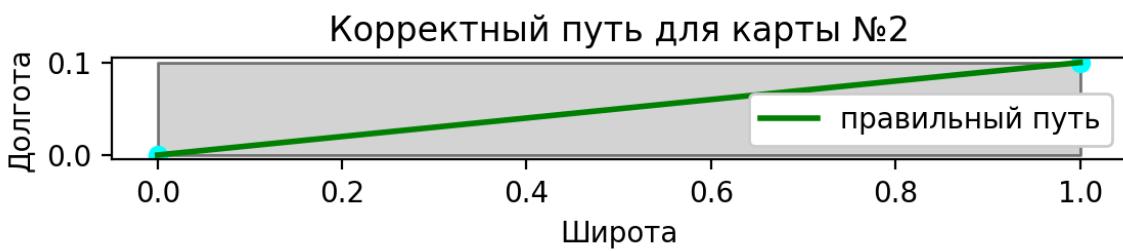


Рис. 4. Корректный путь для второй карты

3. Квадрат размером $[0.9 \times 0.9]$ градусов, который содержит в себе три зоны. Ищем путь между точками, лежащими в противоположных углах квадрата. Эталонный вес пути: 181256.383. Данная карта содержит в себе три зоны с разной стоимостью.

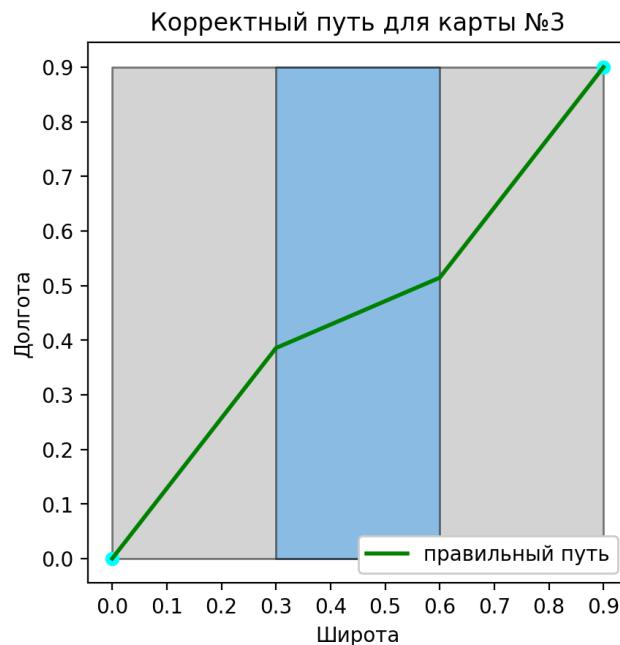


Рис. 5. Корректный путь для третьей карты

4. Квадрат размером $[1.0 \times 1.0]$ градусов, содержащий в себе вписанную в себя сферу. Ищем путь между противоположными углами квадрата. Эталонная длина: 198523.065 м. Добавлен рельеф на карту.

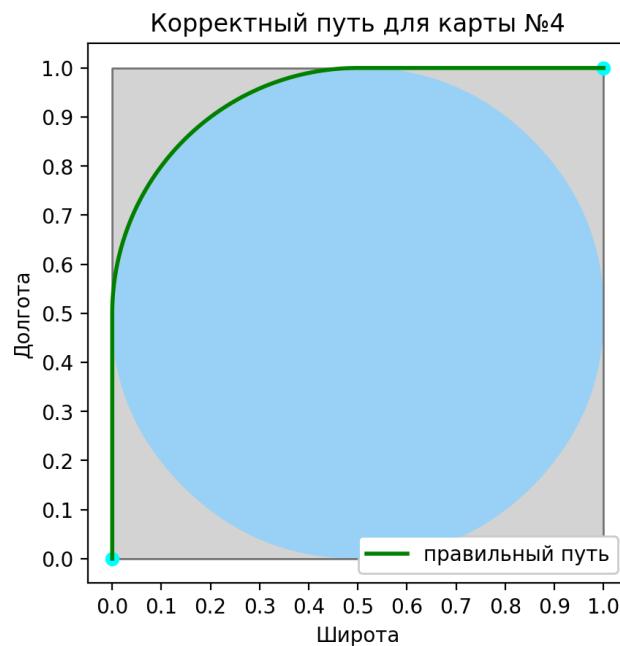


Рис. 6. Один из корректных путей для четвертой карты

5. Квадрат размером [1.0 x 1.0] градусов, содержащий в себе вписанную в себя сферу. Ищем путь между противоположными точками на сфере. Эталонная длина: 174668.365 м. Карта совпадает с предыдущей, но здесь мы ищем путь между противоположными точками, лежащими на сфере.

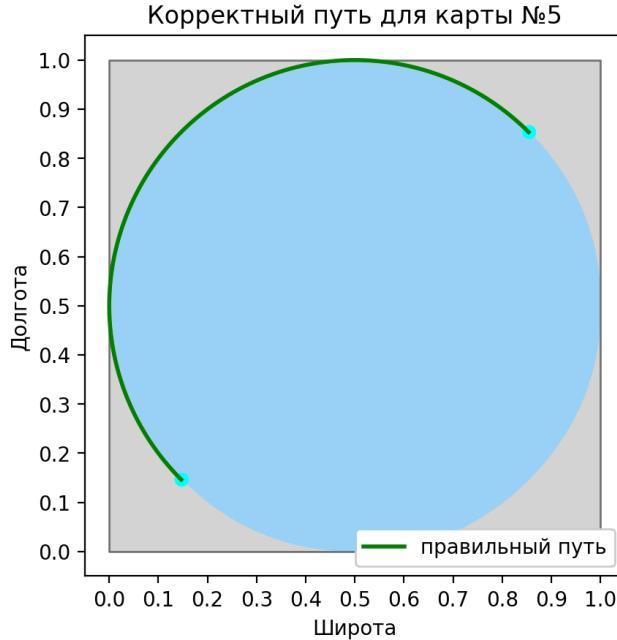


Рис. 7. Один из корректных путей для пятой карты

3.4 Проведение измерений

Основная задача данной главы заключалась в исследовании, насколько выбор параметров для построения сетки влияет на точность пути. Для оценки этих измерений мы использовали относительную ошибку пути, вычисляемую по формуле:

$$error = \frac{|Length_{correct} - Length_{path}|}{Length_{correct}} \cdot 100,$$

где $Length_{correct}$ - эталонная длина пути, найденная аналитически;

$Length_{path}$ - длина пути, полученная в результате работы алгоритма.

Путь представляет собой набор вершин $P : \forall \vec{p} \in P, \vec{p} = \begin{pmatrix} x \\ y \\ h \end{pmatrix}$, для которого длина вычисляется по формуле:

$$length = \sum_{i=0}^{n-1} \rho_T(p_i, p_{i+1}),$$

где $\rho_T(p_i, p_j) = \sqrt{\rho_S(x_i, y_i; x_j, y_j)^2 + (h_i - h_j)^2}$, ρ_S - геодезическое расстояние между точками на сфере.

Измерения проводились следующим образом: для каждой из карт мы перебирали значения γ на отрезке $[0.0, 2.0]$ с шагом 0.2 и различные размеры сетки. Для начала требовалось определить, какая верхняя граница размеров сетки будет давать достаточно низкое значение ошибки. Для этого была использована первая карта. На отрезке $[75, 27800]$ был выбран 51 различный размер d . В итоге был получен следующий график зависимости ошибки путем от значений γ и d (см. рис. 8):

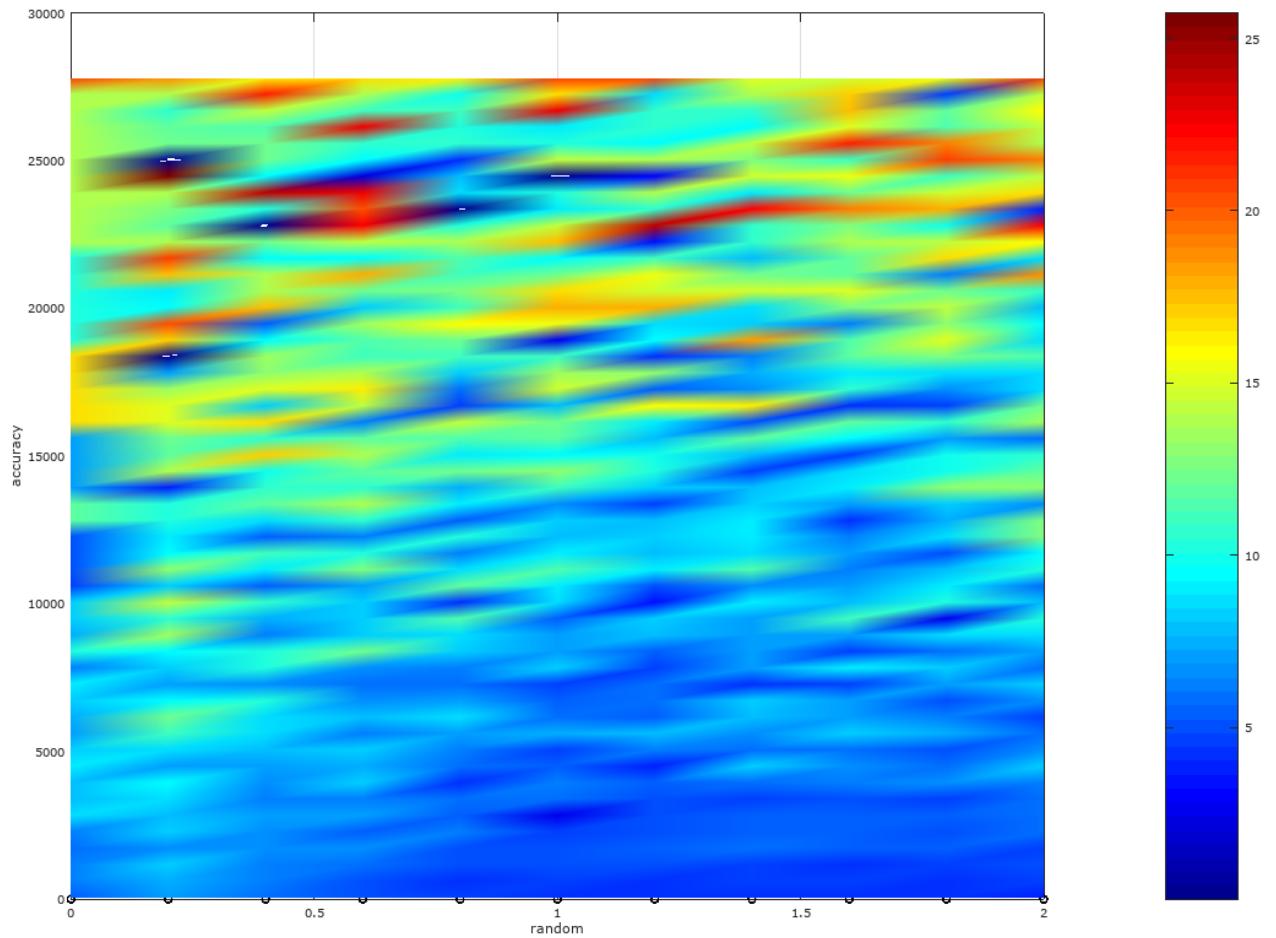


Рис. 8. График значений размера ошибки на $[75, 27800]$

Исходя из этого графика, видно, что достаточно низкие величины ошибок, которые стабильны на всем диапазоне случайной величины, достигаются, если размер сетки d не превышает 5000 м, что составляет чуть менее 5% от размеров стороны квадрата. Найдем еще более точную правую границу. Рассмотрим график значений ошибки для значений d на отрезке $[50, 5000]$ (см. рис. 9):

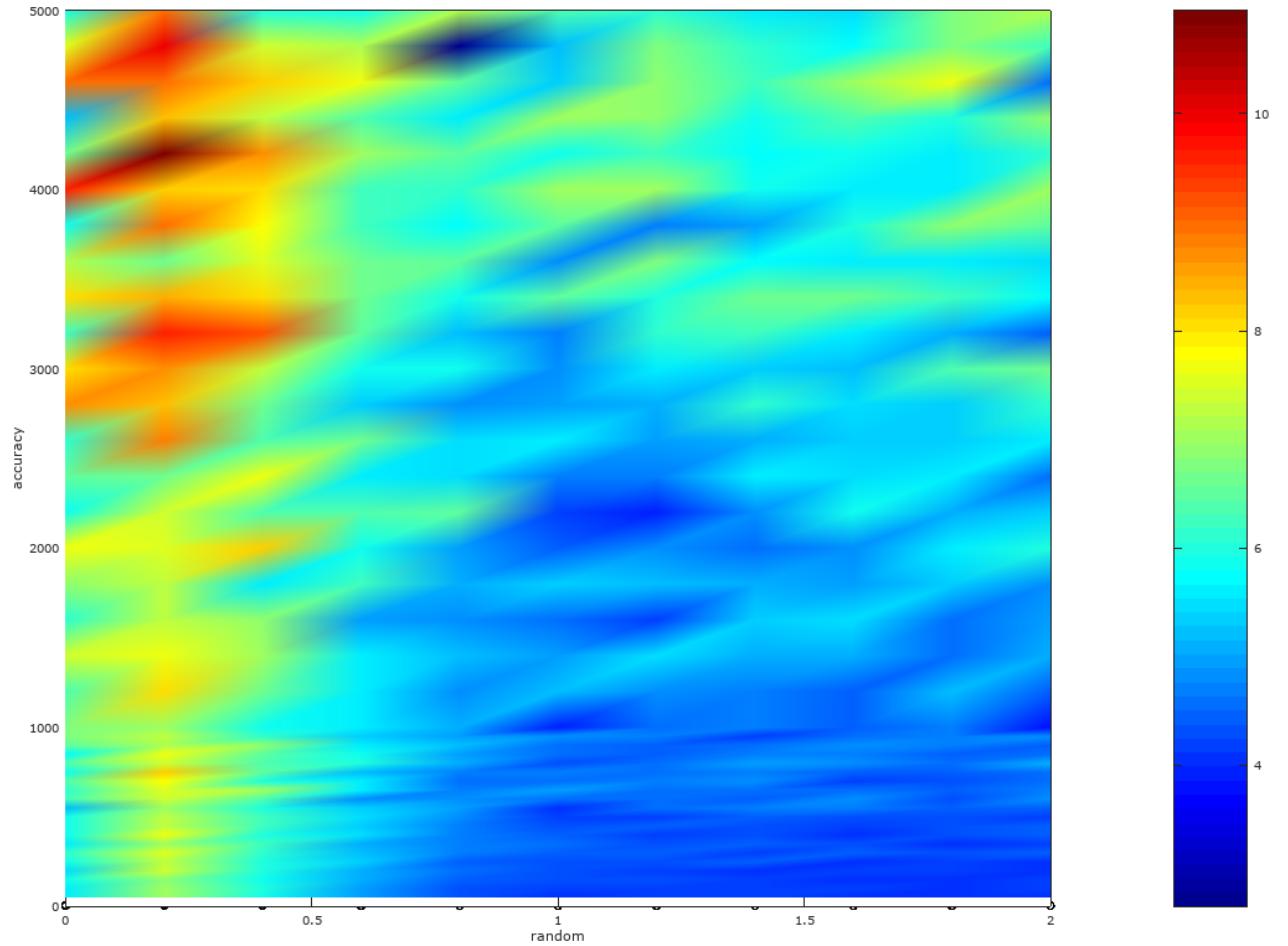


Рис. 9. График значений размера ошибки на $[50, 5000]$

Отсюда мы видим, что наиболее качественные результаты получаются на отрезке $[50, 500]$, что в относительных величинах, составляет от 0.05% до 0.5% относительно стороны квадрата. В дальнейшем мы будем искать кратчайшие пути, используя только этот диапазон.

Определив диапазон значений, перейдем непосредственно к результатам поиска кратчайшего пути и некоторым особенностям, связанным с этим.

- Значение $\gamma = 0.2$ оказалось наихудшим для первой карты, что видно на рисунке(см. рис. 10):

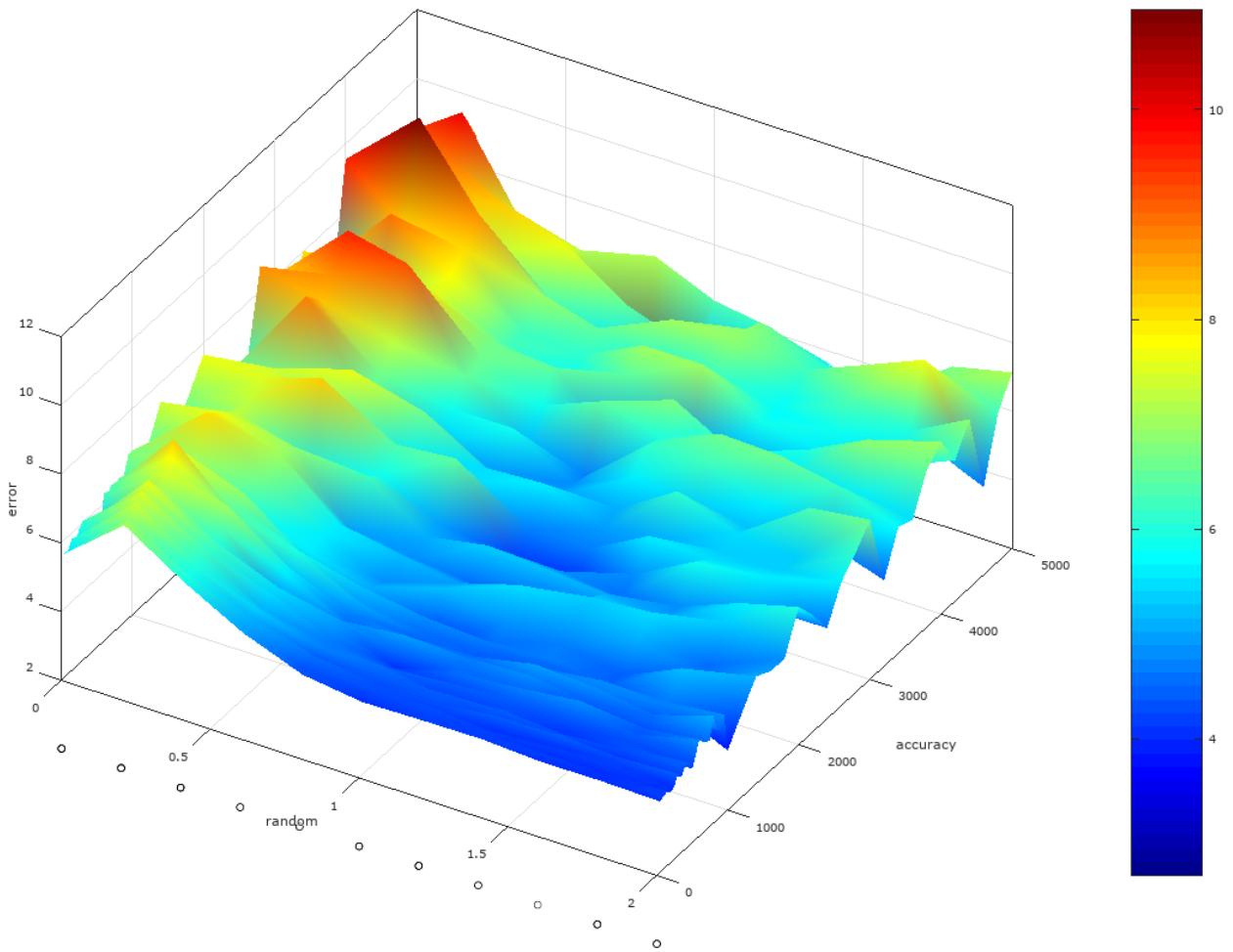


Рис. 10. Значения ошибки для первой карты

Была замечена закономерность, что при $\gamma < 0.5$ были получены наибольшие величины ошибки для всех карт. Но наиболее гладкий путь для второй карты получается, как раз, при значении $\gamma = 0.2$. Однако, величина ошибки от этого все равно не стала минимальной. Это явление можно увидеть на рисунках ниже.

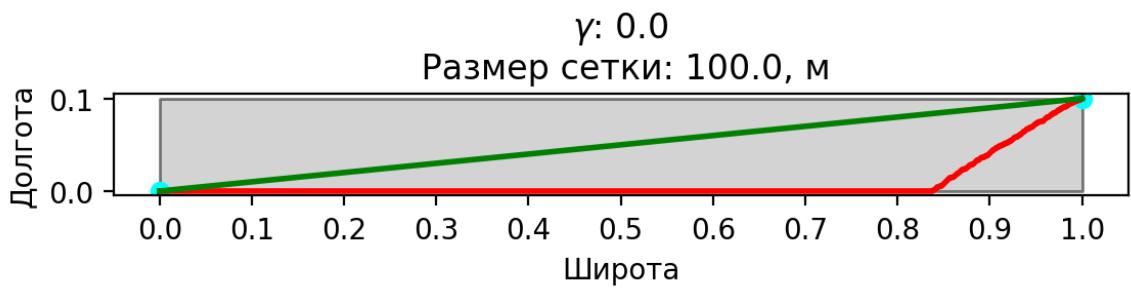


Рис. 11. Путь для второй карты при $\gamma = 0.0$

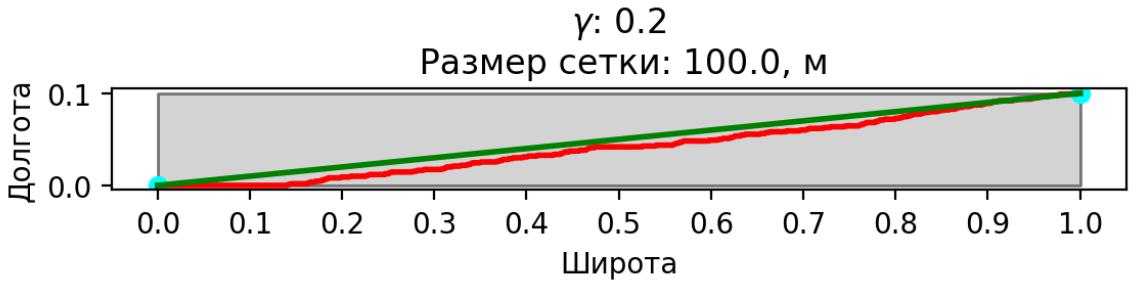


Рис. 12. Путь для второй карты при $\gamma = 0.2$

- Проблема правильности вычисления длины пути на карте с рельефом. При построении пути на карте со сферой появились отрицательные значения ошибки длины пути. Данная проблема была связана со следующим явлением (см. рис. 13):

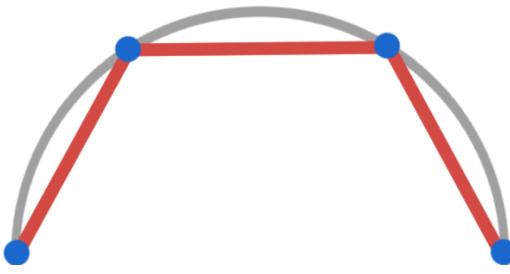


Рис. 13. Проблема меньшей длины пути

Длина дуги окружности будет всегда больше, чем длина пути на многоугольнике, который аппроксимирует эту окружность. Аналогичная ситуация возникает на абсолютно любой карте с использованием рельефа. Поэтому для карт с рельефом важно разбивать, полученный путь на более мелкие части, чтобы правильно вычислять длину пути.

Для сбора статистики для каждой карты были проверены размеры сетки d , лежащие в отрезке $[100, 500]$ с шагом в 50 м и значения γ на отрезке $[0.0, 2.0]$ с шагом 0.2. После получения наборов вершин, представляющих собой пути, к ним был применен алгоритм сглаживания методом скользящего среднего с разным размером окна w , который составлял от 3 вершин до трети количества вершин длины пути. Сглаживание пути на карте с рельефом проводилось на детализированном пути (см. рис. 14) для корректного вычисления длины пути: d – текущий размер сетки, m – минимальный размер сетки.

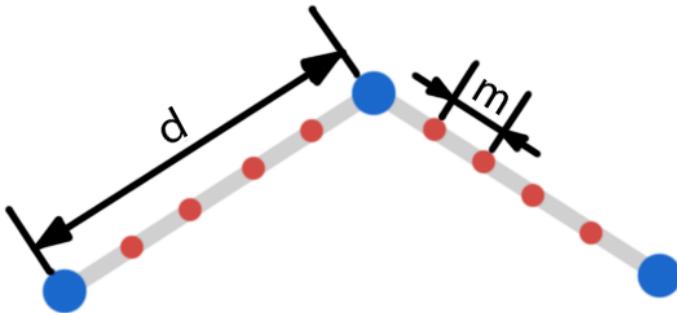


Рис. 14. Детализированный путь

3.5 Анализ результатов

По завершении работы были получены следующие итоговые таблицы с результатами.

Карта	Ошибка %	d , м	γ	w
1	3.973	500 м	2.0	None
2	2.029	500 м	1.4	None
3	3.731	300 м	1.2	None
4	1.734	100 м	2.0	None
5	1.600	300 м	1.8	None

Таблица 2. Наилучшие результаты для начального пути

Карта	Ошибка, %	d , м	γ	w
1	0.018	150 м	0.8	322
2	0.055	100 м	0.2	370
3	0.011	500 м	0.8	29
4	0.099	100 м	0.6	530
5	0.025	150 м	0.6	283

Таблица 3. Наилучшие результаты для сглаженного пути

Если мы хотим получить изначальный путь достаточно высокого качества, то значение γ должно лежать в отрезке $[1.4, 1.8]$. Если мы потом собираемся применять алгоритмы сглаживания к построенному пути, то значение γ должно лежать в отрезке $[0.6, 0.8]$. Сглаживание пути принесло улучшение результата во всех рассмотренных случаях. Размер окна сглаживания w в среднем не превосходит 15-20% от общего количества вершин пути. Поэтому как правое ограничение окна сглаживания можно установить на уровне 5-9% от количества вершин пути.

При значении γ равном 0.0 или 0.2 были получены пути с наибольшим значением ошибки, как для обычного пути, так и для сглаженного.

Временная сложность данного алгоритма является квадратичной(так как основную часть времени работы алгоритма составляет добавление элементов сетки в граф). Ниже представлена таблица зависимости времени работы алгоритма к размеру сетки для первой карты (см. табл. 4):

50	100	150	200	250	300	350	400	450	500	d , м
353	82	36	20	12.5	8.75	6.7	4.85	3.5	2.93	t , сек

Таблица 4. Время работы алгоритма в зависимости от размера сетки

Эти данные позволяют нам сделать вывод о том, что достаточно высокую точность пути можно получить и не используя очень мелкую сетку. И можно достигать вполне эффективных результатов и с $d = 300$ м.

Глава 4. Построение оптимальной сети

После того как мы научились находить оптимальные пути между двумя точками, стоит задача построения оптимальной сети линейных объектов. Встречаемые в литературе алгоритмы построение деревьев Штейнера на графах ориентированы на более общий случай задачи, поэтому будут недостаточно эффективны в нашем случае. Граф в нашем случае является сильно разреженным, с малым числом терминальных вершин, относительно общего количества вершин в графе, а также наша задача является прикладной. Поэтому было решено разработать собственный алгоритм, использующий определенную эвристику для эффективного построения оптимальной сети.

4.1 Описание используемого алгоритма

Введем ряд определений, используемых далее.

Пусть задан граф $\mathbb{G} = (\mathbb{V}_G, \mathbb{E}_G)$, где $\mathbb{V}_G = \{v_1, v_2, \dots, v_n\}$ — множество вершин, а $\mathbb{E}_G = \{e_1, e_2, \dots, e_m\}$ — множество рёбер.

Терминальными точками назовем множество точек $T \subseteq \mathbb{V}_G$, между которыми требуется построить дерево Штейнера.

При этом такой подграф $ST \subseteq \mathbb{G}$, соединяющий T , минимально возможного веса $w(ST)$ является деревом, которое называется минимальным деревом Штейнера на T .

Точками Штейнера называется множество точек $P \subseteq \mathbb{V}_{ST} \setminus T$.

Точками разветвления назовем подмножество точек Штейнера $S \subseteq P$, которые были добавлены во время работы алгоритма, реализующие лучшее решение, чем построенное только на терминальных точках.

Минимальное оставное дерево — это оставное дерево этого графа, имеющее минимальный возможный вес, где под весом дерева понимается сумма весов, входящих в него рёбер [23]. Обозначим сумму весов ребер графа за $w : \mathbb{G} \rightarrow \mathbb{R}$.

Идея алгоритма

Для начала, создается ограничительный прямоугольник, для которого мы генерируем набор точек разветвления. Мы разбиваем прямоугольник на n_x точек по горизонтали и n_y точек по вертикали. Итого, имеем $n_x \cdot n_y$ точек разветвления на плоскости (см. рис. 15). Для каждой точки разветвления мы находим ближайшую к ней по расстоянию вершину в графе.

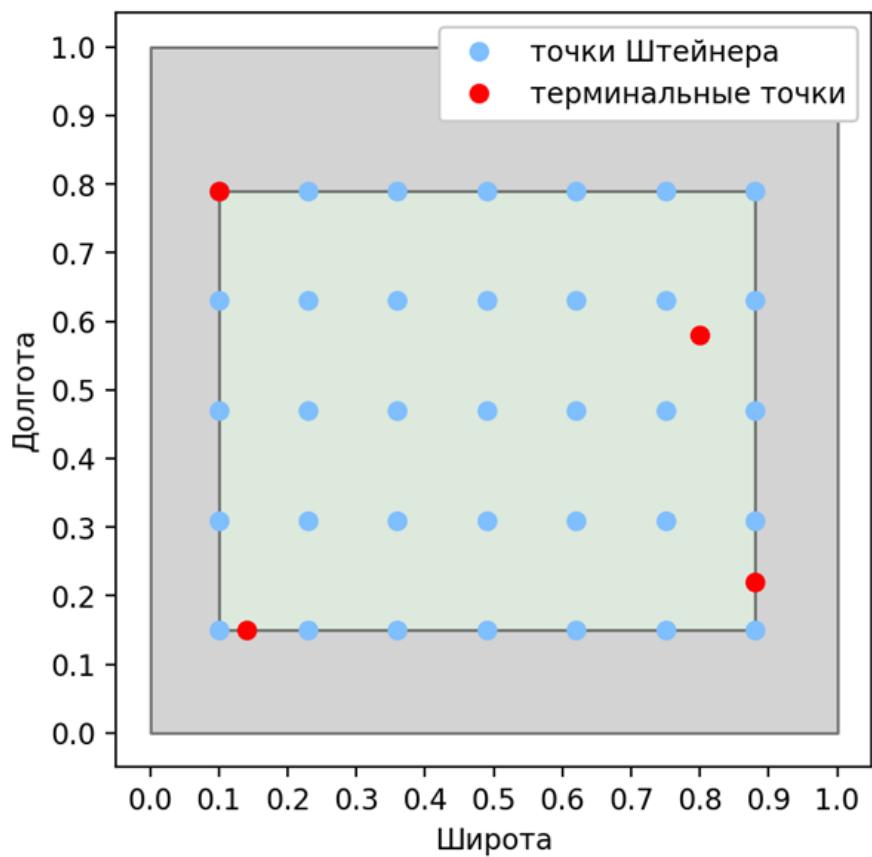


Рис. 15. Получение точек Штейнера

После получения набора точек разветвления, нам нужно выбрать такое их подмножество, которое образует дерево минимального веса (см. рис. 16). Для этого нам потребуется вычислять минимальные остовные деревья (далее MST).

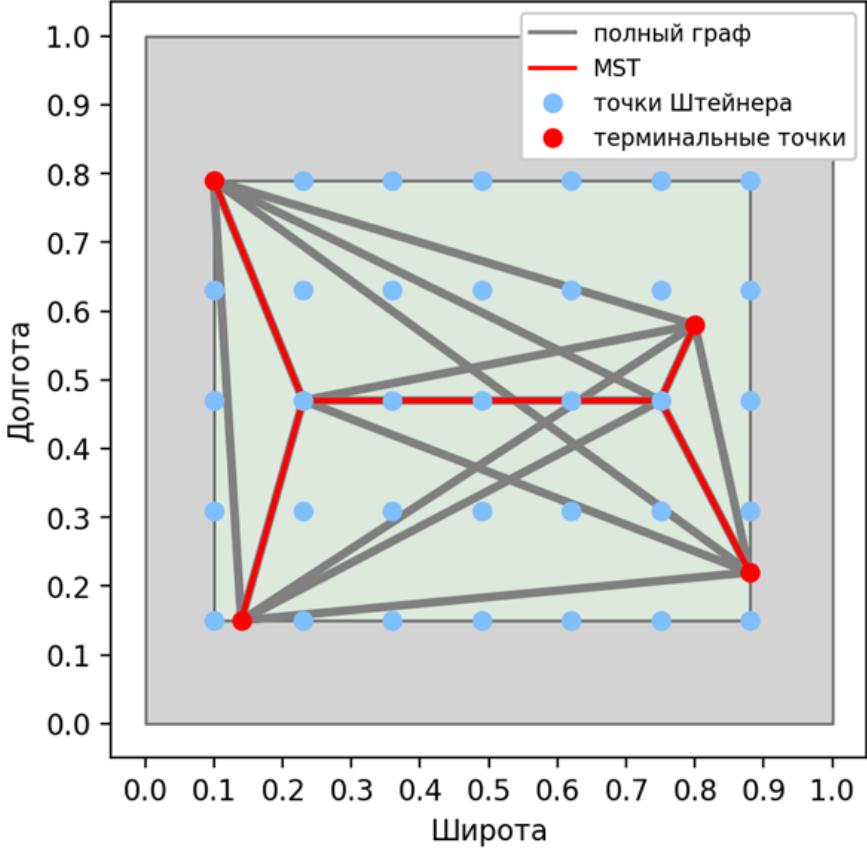


Рис. 16. Полный граф и MST

Известно, что количество точек разветвления в итоговом графе не может превышать $m = |T| - 2$, где $|T|$ - количество терминальных точек. Соответственно, наш алгоритм будет иметь m шагов. Создадим массив *step_solutions*, отвечающий за хранение k лучших результатов, полученных на каждом шаге алгоритма.

На каждой итерации алгоритма нам требуется построить набор полных графов для различных комбинаций терминальных точек и точек разветвления. Этот этап можно выделить в функцию `get_full_graphs`. Набор полных графов строится по следующему принципу: за основу для генерации берется набор терминальных точек, Весами ребёр в полном графе будет стоимость кратчайшего пути между двумя вершинами ребра в изначальном графе. То есть, пусть у нас задано $t = |T|$ терминальных точек, k лучших решений и n точек разветвления, таким образом у нас должно быть сгенерировано $k \cdot n$ полных графов, так как терминальные вершины добавляются всегда, как и

наборы вершин из лучших решений. И к каждому из этих наборов вершин нужно добавить по одной из точек разветвления.

После этого для каждого полученного полного графа требуется построить минимальное оствовное дерево и взвесить его.

Далее нужно отобрать k лучших по весу минимальных оствовных деревьев и добавить их в массив $step_solutions$, отвечающий за лучшие решения на текущем шаге.

После того, как все шаги выполнены, нужно найти минимальное по весу дерево среди всех полученных решений из $step_solutions$. А далее нам нужно восстановить кратчайшие пути между вершинами полученного решения и таким образом мы получим оптимальную сеть L_{res} , состоящую из набора ломаных линий.

Псевдокод данного алгоритма представлен на Algorithm 1.

Data: $\mathbb{G} = (\mathbb{V}_G, \mathbb{E}_G)$ - взвешенный граф, $T \subseteq \mathbb{V}_G$ - набор

терминальных точек

Result: $L_{res} = \{\{x_j, y_j\}_{j=1}^{n_i}, i = \overline{1..s}\}$ - оптимальная сеть линейных объектов

steiner_nodes := generate_steiner_nodes(\mathbb{G}, T)

step_solutions := \emptyset

$m = |T| - 2$

for $i = 0$ **to** $m - 1$ **do**

full_graphs := get_full_graphs($T, step_solutions[i-1], steiner_nodes$)

builded_MSTs := get_MSTs(full_graphs)

step_solutions[i] := k_best_MSTs(builded_MSTs)

end

best_solution := min(step_solutions)

result := recover_paths(best_solution)

Algorithm 1: Алгоритм нахождения оптимальной сети

4.2 Описание модельных карт для тестирования

Задача данного исследования найти зависимость между параметрами: временем работы программы и точностью получаемого решения. Для проведения этого исследования воспользуемся модельной картой в виде квадрата размером $[1.0 \times 1.0]$ градусов и построим вычислительную сетку с размером $d = 250$ м. Параметры n_x и n_y будем варьировать в диапазоне $[3, \frac{width}{3d}]$ и $[3, \frac{height}{3d}]$, где $width$ и $height$ – ширина и высота ограничивающего прямоугольника в метрах. Будем строить дерево Штейнера для трех объектов. В качестве эталонного решения, возьмем решение, вычисленное аналитически (см. рис. 17).

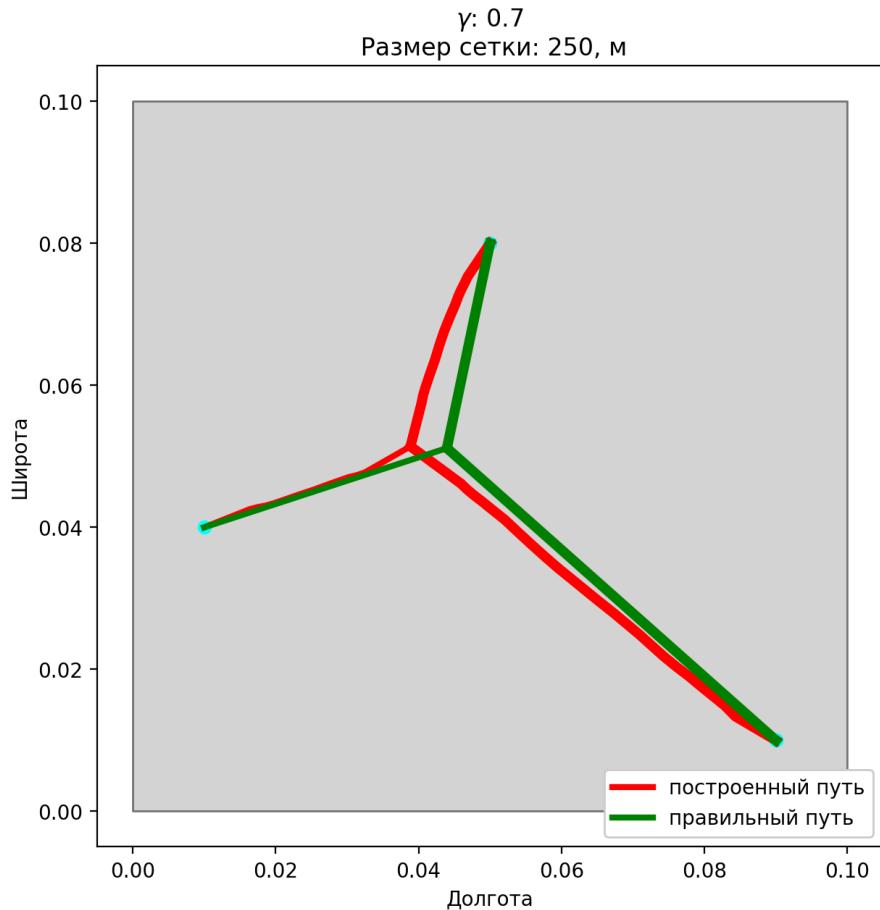


Рис. 17. Пример получившегося решения

После выполнения вычислений была получена следующая таблица (см. табл. 5). Всего точек в графе 198025.

Ошибка, %	Время, сек	Точки разветвления
4.812	3.0	9
1.216	68.6	143
0.897	125.4	480
1.061	236.3	980
0.979	372.6	1702
1.017	531.4	2565
0.985	665.1	3618
0.939	885.6	4836
0.855	1079.2	6319
0.924	1271.6	7900
0.912	1415.4	9768

Таблица 5. Результаты работы алгоритма

Исходя из данных в этой таблице видно, что на плоскости качественные результаты можно получать при количестве точек разветвления в размере около 0.5-1% от общего числа точек в ограничивающем прямоугольнике.

Помимо поиска решения в простейшем случае, основанных на Евклидовой метрике, требуется описание тестовых случаев для проверки алгоритма на корректность, опираясь на специфику нашей задачи.

1. Квадрат размером [1.0 x 1.0] градусов. Данный тестовый случай характерен тем, что три точки образуют угол в 120°.

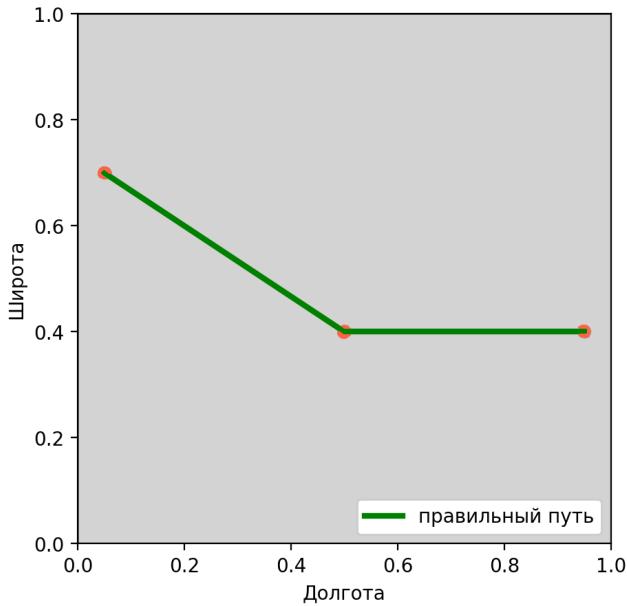


Рис. 18. Первый тестовый случай

2. Квадрат размером [1.0 x 1.0] градусов. Посередине квадрата задана до-

рога со стоимостью строительства на ней значительно меньшей, чем в окружающей её области. Точка разветвления в этом случае должна находиться на дороге.

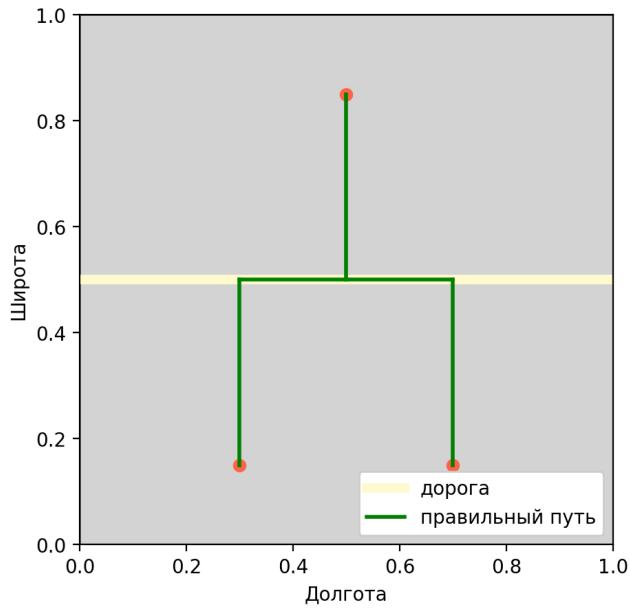


Рис. 19. Второй тестовый случай

3. Квадрат размером $[1.0 \times 1.0]$ градусов. В месте, где должна быть точка Штейнера расположена область с высокой стоимостью строительства.

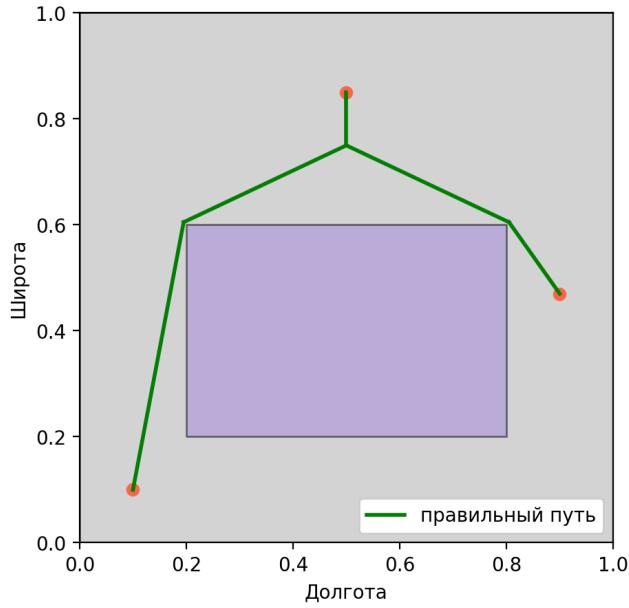


Рис. 20. Третий тестовый случай

4. Квадрат размером $[1.0 \times 1.0]$ градусов. Четвертый случай представляет комбинацию предыдущих двух случаев: посередине квадрата проходит

дорога с низкой стоимостью строительства на ней, а также на предполагаемом месте точки разветвления находится область с высокой стоимостью строительства.

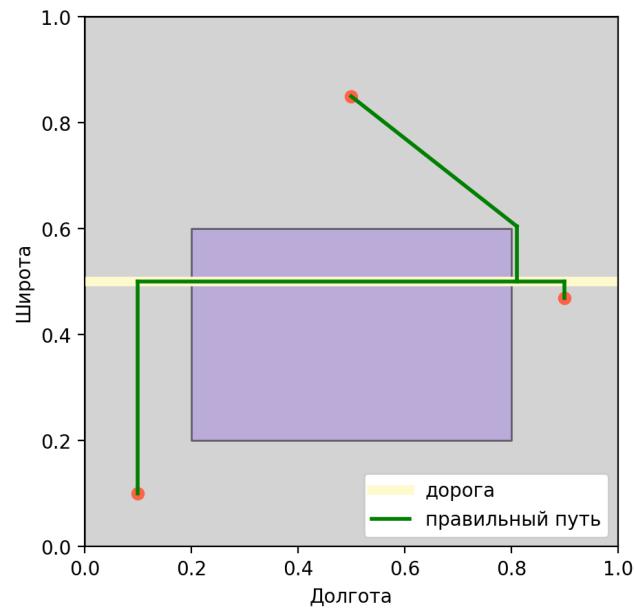


Рис. 21. Четвертый тестовый случай

Выводы

В ходе данной работы была составлена математическая модель задачи проектирования линейных объектов. Общая задача была разбита на два этапа: решение задачи оптимального построения линейного объекта и решение задачи построения оптимальных сетей.

В рамках первого этапа задача была сведена к поиску кратчайшего пути на графе. Было сгенерировано несколько модельных карт местности, для которых было найдено аналитическое решение и проведено сравнение с результатами полученными численно. Наименьшее значение ошибки для всех модельных карт составило менее 0.1% за время не превышающее 85 секунд.

В рамках второго этапа был реализован свой алгоритм на основе знаний о предметной области. Была найдена зависимость отношения ошибки между численным и аналитическим решением, временем работы алгоритма, количеством точек разветвления. На основании чего был сделан вывод о том, что можно получать решения с величиной ошибки около 1% с количеством точек разветвления менее 0.1% от общего числа точек в графе за время менее 130 секунд.

Заключение

В рамках данной работы была исследована проблема поиска оптимального пути между двумя объектами, был предложен и реализован алгоритм, позволяющий строить оптимальные сети линейных объектов с учетом рельефа и различных типов местности. Также были предложены различные случаи для тестирования корректности работы программы.

В качестве дальнейшего направления работы можно выделить повышение точности построенного маршрута, уменьшения времени работы алгоритма, введение дополнительных параметров, используемых в расчёте.

Список литературы

1. "Градостроительный кодекс Российской Федерации" от 29.12.2004 N 190-ФЗ (ред. от 31.12.2017)
2. Инженерно-геологические условия [Интернет ресурс]: URL:https://ru.wikipedia.org/wiki/Инженерно-геологические_условия (дата обращения: 10.03.18)
3. Справочное руководство по проектированию разработки и эксплуатации нефтяных месторождений. Добыча нефти. Под общ.ред. Ш.К. Гиматудина, Р.С. Андриасов, И.Т. Мищенко, А.И. Петров и др. - М.: Недра, 1983, 455 с.
4. Задача Штейнера о минимальном дереве [Интернет ресурс]: URL:https://ru.wikipedia.org/wiki/Задача_Штейнера_о_минимальном_дереве (дата обращения: 10.03.18)
5. Э. Н. Гордеев, О. Г. Тарасцов, Задача Штейнера. Обзор, Дискрет. матем., 1993, том 5, выпуск 2, 3–28
6. M. L. Shore, L. R. Foulds, P. B. Gibbons An Algorithm for the Steiner Problem in Graphs, Networks, Vol. 12, 1982, pp. 323-333.
7. Д. Т. Лотарев, Задача Штейнера для транспортной сети на поверхности, заданной цифровой моделью, Автомат. и телемех., 1980, выпуск 10, 104–115
8. DIMACS-11 [Интернет ресурс]: URL:<http://dimacs11.zib.de/> (дата обращения: 14.04.18)
9. Цифровая модель рельефа [Интернет ресурс]: URL:<http://www.geodata.ru/cifrovyemodelirelief> (дата обращения: 14.04.18)
10. Shuttle Radar Topography Mission (SRTM) [Интернет ресурс]: URL:<http://gis-lab.info/qa/srtm.html> (дата обращения: 14.04.18)

11. Advanced Spaceborne Thermal Emission and Reflection Radiometer Global Digital Elevation Model (ASTER GDEM) [Интернет ресурс]: URL:<http://gis-lab.info/qa/aster-gdem.html> (дата обращения: 14.04.18)
12. Global Multi-resolution Terrain Elevation Data 2010 (GMTED2010) [Интернет ресурс]: URL:<https://lta.cr.usgs.gov/GMTED2010> (дата обращения: 14.04.18)
13. WorldDEM [Интернет ресурс]: URL:<http://www.intelligence-airbusds.com/worlddem/> (дата обращения: 14.04.18)
14. Геоид [Интернет ресурс]:
URL:<https://ru.wikipedia.org/wiki/Геоид> (дата обращения: 14.04.18)
15. Земной эллипсоид [Интернет ресурс]:
URL:https://ru.wikipedia.org/wiki/Земной_эллипсоид (дата обращения: 14.04.18)
16. Референц-эллипсоид [Интернет ресурс]:
URL:<https://ru.wikipedia.org/wiki/Референц-эллипсоид> (дата обращения: 14.04.18)
17. Haversine formula [Интернет ресурс]: URL:https://en.wikipedia.org/wiki/Haversine_formula (дата обращения: 03.05.18)
18. Vincenty's formulae [Интернет ресурс]: URL:https://en.wikipedia.org/wiki/Vincenty's_formulae (дата обращения: 03.05.18)
19. Grid (Spatial index) [Интернет ресурс]: URL:[https://en.wikipedia.org/wiki/Grid_\(spatial_index\)](https://en.wikipedia.org/wiki/Grid_(spatial_index)) (дата обращения: 03.05.18)
20. A* search algorithm [Интернет ресурс]: URL:https://en.wikipedia.org/wiki/A*_search_algorithm (дата обращения: 07.05.18)
21. Митрушкин Д.А., Попов Ю.П. Об одном способе локального измельчения расчетной сетки вблизи кругового источника малого размера // Препринты ИПМ им. М.В.Келдыша. 2014. № 25. 32 с.

22. Триангуляция Делоне [Интернет ресурс]:
URL:https://ru.wikipedia.org/wiki/Триангуляция_Делоне (дата обращения: 18.05.18)
23. Минимальное оствовное дерево [Интернет ресурс]:
URL:https://ru.wikipedia.org/wiki/Минимальное_остовное_дерево (дата обращения: 18.05.18)

Приложение

Приложение А

Визуальные отличия вычислительных сеток в зависимости от параметра γ .

