

*Д. С. Парыгин,
А.В. Игнатьев,
Н. П. Садовникова,
А. С. Гуртяков*

*Геоинформационные
системы*

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
ВОЛГОГРАДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

На правах рукописи

Д. С. Парыгин,
А.В. Игнатъев, Н. П. Садовникова,
А. С. Гуртяков

Геоинформационные системы

Учебное пособие



Волгоград
2021

УДК: 004.89

Рецензенты:

Печатается по решению редакционно-издательского совета
Волгоградского государственного технического университета

Парыгин, Д. С.

Геоинформационные системы / Д. С. Парыгин, А. В. Игнатъев,
Н. П. Садовникова, А. С. Гуртяков ; ВолГТУ. – Волгоград, 2021. –
118 с.

ISBN 978-5-9948-0000-0

Материалы пособия посвящены вопросам сбора и подготовки географических данных, организации данных в геоинформационных системах, основам геопространственного анализа. Приведены примеры задач пространственного анализа и инструменты для их решения. Описаны возможности современных геоинформационных систем. Пособие предназначено для студентов очной и заочной форм обучения по направлениям 09.03.01 «Информатика и вычислительная техника», 09.03.02 «Информационные системы и технологии», 09.04.01 «Информатика и вычислительная техника», 09.04.02 «Информационные системы и технологии» (профиль «Искусственный интеллект в проектировании городской среды»), 08.04.01 «Строительство» (профиль «Организация информационного моделирования в строительстве»).

Ил. 32. Табл. 4. Библиогр.: 3 назв.

ISBN 978-5-9948-0000-0

© Волгоградский государственный
технический университет, 2021

© Д.С.Парыгин, А. В. Игнатъев,
Н. П. Садовникова, А. С. Гуртяков, 2021

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ	5
1. Курс лекций по дисциплине	7
1.1 Основы геоинформатики	7
1.1.1. Географическая информация и географические данные.....	9
1.1.2 Геоинформационные технологии и функции ГИС	10
1.1.3 Принципы организации ГИС	11
1.2 Модели пространственных данных	15
1.2.1 Растровые модели.....	15
1.2.2 Векторные модели	18
1.3 Задачи анализа пространственных данных.....	21
1.4 Методы анализа геопространственных данных	24
1.5 Картографические сервисы и пространственные данные.....	26
1.6 Управление данными в ГИС (на примере OpenStreetMap).....	31
1.7 Моделирование на основе пространственных данных.....	34
2. Методические указания к лабораторным работам.....	45
2.1 Лабораторная работа № 1. Модели пространственных данных.....	45
2.2 Лабораторная работа № 2. Отображение пространственных данных	51
2.2.1 Цель работы.....	51
2.2.2 Порядок выполнения работы	51
2.2.3 Теоретический материал.....	51
2.2.3.1 Добавление тайловых карт в QGIS	52
2.2.3.2 Подготовка данных с помощью сервиса BbVike.....	54
2.2.3.3 Гексагональная сетка	55
2.2.3.4 Тепловые карты	56
2.2.3.5 Диаграмма Вороного	58
2.2.3.6 Вывод изображений на печать	60
3.2.4 Практические задания	63

3.2.6 Вопросы к отчету.....	64
2.3 Лабораторная работа № 3. Создание карты субъекта Российской Федерации на основе данных OpenStreetMap	65
2.4 Лабораторная работа № 4. Анализ транспортных сетей.....	71
2.5 Лабораторная работа № 5. Методы и средства визуализации данных в ГИС	83
2.5.1 Цель работы:	83
2.5.3 Теоретический материал.....	90
2.5.4 Практические задания.....	90
2.5.6 Вопросы к отчету.....	91
2.6 Лабораторная работа № 6. Классификации мультиспектральных снимков со спутника	91
3. Методические указания к контрольной работе “Моделирование на основе пространственных данных”	103
3.1 Цель работы.....	103
3.2 Порядок выполнения работы	103
3.2.1 Настройка и запуск системы моделирования OSMLS	103
3.2.2 Создание и запуск тестового модуля.....	104
3.2.3 Создание и запуск модуля на основе индивидуального задания	108
3.3 Теоретический материал.....	108
3.4 Практические задания	109
Рекомендуемая литература по курсу.....	116

ВВЕДЕНИЕ

Первые геоинформационные системы (ГИС) были созданы в Канаде, США и Швеции для изучения природных ресурсов в середине 1960-х годах. Сегодня ГИС-технологии получили широкое распространение во многих сферах и обеспечивают поддержку работы с любыми данными, имеющими пространственно-временную привязку. С их помощью решаются задачи интеграции картографической информации, данных дистанционного зондирования и экологического мониторинга и пр. ГИС представляет собой централизованную базу данных пространственных объектов и инструмент, который предоставляет возможности хранения, анализа и обработки любой информации, связанной с тем или иным объектом. Эффективность использования ГИС, возможность реализовать требуемые функции, построить полноценную информационную систему, интегрировать ее в существующую информационную инфраструктуру существенным образом зависит от свойств и качества программного обеспечения ГИС.

Можно выделить наиболее известных поставщиков программного обеспечения для ГИС: Autodesk Inc (AutoCAD Map, AutoCAD Civil, MapGuide, ESRI (USA) (ArcGIS), MapInfo Corp (MapInfo), ЦГИ ИГ РАН (GeoGraph/ GeoDraw/GeoConstructor), GeoSpectrum International (ГИС Panorama), НИИПМК (ГИС Terra).

С каждым годом растет потребность в Web-сервисов, использующих пространственные данные и метаданные для решения самых разных прикладных задач:

- управление земельными ресурсами, ведение земельных кадастров;
- инвентаризация, учет, планирование размещения объектов распределенной производственной инфраструктуры;
- проектирование, планировка электросетей;

- управление наземным, воздушным и водным транспортом;
- экологический мониторинг;
- оперативная оценка распространения пожаров;
- прогноз полезных ископаемых, оценка и подсчет запасов месторождений в геологии;
- прогнозирование урожайности и пр.

В настоящее время ГИС является основой муниципальной информационной системы, так как интегрирует все пространственные данные по объектам городской территории и может обеспечить поддержку для анализа этих данных и решения аналитических задач.

Современные тенденции развития ГИС-технологий связаны с интеллектуальным анализом данных и искусственным интеллектом. Большие объемы данных, интегрированные в ГИС, обеспечивают возможность создания и обучения моделей ИИ для прогнозирования, бизнес-аналитики и создания систем управления на основе данных. Умение работать с инструментами анализа пространственных данных является одной из наиболее перспективных и востребованных компетенций ближайшего времени. Целью создания данного пособия является ознакомление студентов и других заинтересованных лиц с методами и технологиями ГИС, принципами разработки программных систем на их основе.

1. КУРС ЛЕКЦИЙ ПО ДИСЦИПЛИНЕ

1. 1 Основы геоинформатики

Геоинформатика – это наука, технология и производственная деятельность по научному обоснованию, проектированию, созданию, эксплуатации и использованию географических информационных систем, по разработке геоинформационных технологий, по приложению геоинформационных систем (ГИС) для практических и научных целей.

Геоинформатика – это наука, технология и производственная деятельность по научному обоснованию, проектированию, созданию, эксплуатации и использованию географических информационных систем, по разработке геоинформационных технологий, по приложению ГИС для практических и научных целей.

Основные части геоинформатики:

1. Общая геоинформатика
2. Прикладная геоинформатика
3. Специальная геоинформатика

Общая геоинформатика — это раздел геоинформатики, занимающийся исследованием и разработкой научных основ, концепций, обобщенным анализом геоинформационных систем безотносительно к их прикладному характеру.

Прикладная геоинформатика изучает практические методы работ с геоинформационными системами и геоинформационными технологиями

Специальная геоинформатика служит основой для анализа систем и методов обработки пространственных данных.

В основе теории геоинформатики как учения о ГИС лежит несколько базовых понятий. К ним относятся понятия пространственного объекта, пространственных данных, моделей пространственных данных, функций

их обработки, включая базовые функции пространственного анализа и геомоделирования как ядра ГИС.

Проблемы разработки, функционирования и использования ГИС находятся на стыке трех областей научных знаний: компьютерные науки, науки о земном пространстве, области ГИС-приложений (табл. 1).

Табл. 1. Базовые области научных знаний ГИС.

Компьютерные науки	Науки о земном пространстве	Области ГИС-приложений
Информатика и программирование, математическое моделирование, операционные системы, текстовые и графические редакторы, электронные таблицы, базы данных, информационные сети, обработка изображений, . . .	Геодезия, картография, география, аэрокосмическая съемка, дистанционное зондирование Земли, глобальные системы позиционирования, . . .	Управление территорией, градостроительство и архитектура, инженерная инфраструктура, управление недвижимостью, транспорт и логистика, экология, природные ресурсы, демографические исследования, оборона, сельское хозяйство, Правонарушения, чрезвычайные ситуации, . . . (около 80 дисциплин)

1.1.1. Географическая информация и географические данные

Географическая информация – это информация об объектах, системах объектов, явлениях и процессах реального мира, которые имеют или могут иметь пространственную привязку в реальном пространстве Земли.

Основной единицей в ГИС являются данные. Данные (лат. datum – акт) – совокупность фактов и сведений, представленных в каком-либо формализованном виде для их использования в науке и других сферах человеческой деятельности. Под данными в среде ГИС понимается информация, известная об объектах реального мира; результаты наблюдений и измерений этих объектов. Элемент данных содержит две главные компоненты: географические сведения, описывающие его местоположение в пространстве относительно других объектов (пространственные данные), и атрибутивные данные, которые описывают сущность, характеристики, переменные и значения (рис. 1).

Геопространственные данные – это данные о предметах, формах территории и инфраструктурах на поверхности Земли, причем как существенный элемент в них должны присутствовать пространственные отношения (связи). Геопространственные данные идентифицируют географическое местоположение и свойства естественных или искусственно созданных объектов, а также их границ на земле.

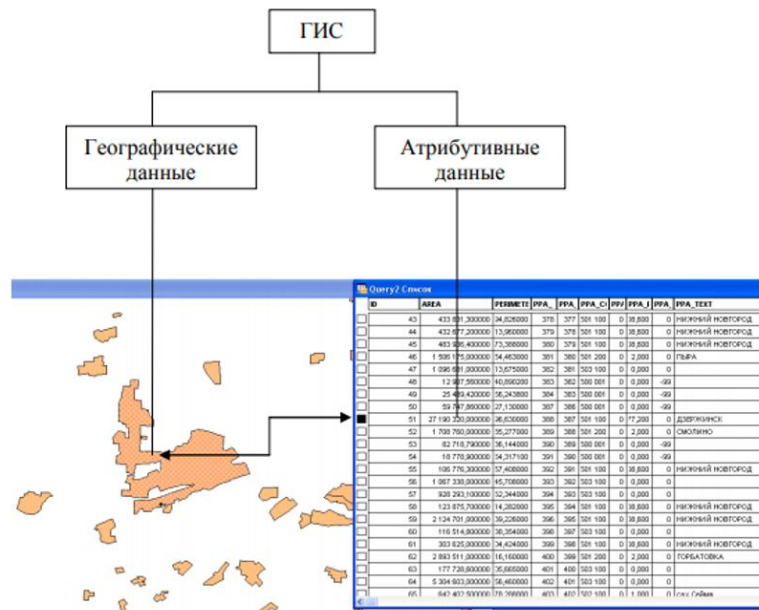


Рис. 1. Компоненты данных в ГИС

1.1.2 Геоинформационные технологии и функции ГИС

Геоинформационные технологии можно определить как совокупность программно технологических средств получения новых видов информации об окружающем мире. Геоинформационные технологии предназначены для повышения эффективности: процессов управления, хранения и представления информации, обработки и поддержки принятия решений.

Основные функции ГИС:

- 1) функции автоматизированного картографирования;
- 2) функции пространственного анализа;
- 3) функции управления данными.

Функции автоматизированного картографирования используют для производства высококачественных карт и изображений. Функции автоматизированного картографирования должны включать векторно-растровые преобразования, преобразования координатной системы,

картографических проекций и масштабов, «склейки» отдельных листов, осуществления картометрических измерений (вычисления площадей, расстояний), размещение текстовых надписей и внемасштабных картографических знаков, формирование макета печати.

Функции пространственного анализа должны обеспечивать совместное использование и обработку картографических и атрибутивных данных и включают анализ географической близости, анализ сетей, топологическое наложение полигонов, интерполяцию и изолинейное картографирование полей, вычисление буферных зон.

Функции управления данными должны обеспечивать работу с атрибутивными (неграфическими) данными ГИС с целью их отбора, обновления и преобразования для производства стандартных и рабочих отчетов. Функции управления данными должны включать пользовательские запросы, генерацию пользовательских документов, статистические вычисления, логические операции, поддержание информационной безопасности, стандартных форм запросов и представления их результатов. В общем случае ГИС должна состоять из следующих четырех подсистем:

- сбора, подготовки и ввода данных;
- хранения, обновления и управления данными;
- обработки, моделирования и анализа данных;
- контроля, визуализации и вывода данных.

1.1.3 Принципы организации ГИС

Рабочей средой при работе с ГИС-платформой является Проект. Проект может включать в себя все информационные компоненты, на которых строятся ГИС-технологии. Основной структурной единицей ГИС

является тематический слой, понятие которого тесно связано с более общим понятием покрытия, несущим в себе объектное содержание (например, единица административно-территориального деления).

Покрытие (Coverage) — цифровая модель единицы хранения базы векторных данных ГИС, хранит в виде записей все объекты первичного уровня (точки, дуги, полигоны) и вторичного уровня (координаты опорных точек, аннотации и т.д.) некоторого пространственного объекта и структуру отношений между ними, в том числе топологические. Пустое покрытие — покрытие, в котором отсутствуют какие-либо пространственные объекты.

Слой (Map Layer) — покрытие, рассматриваемое в контексте его содержательной определенности (растительность, рельеф, административное деление и т.п.) или его статуса в среде редактора (активный слой, пассивный слой).

Слой, как правило, однороден не только по тематике, но и по типам объектов (точечные, линейные, полигональные, растровые). Информационные компоненты могут быть внешними (векторные и растровые слои, таблицы, библиотеки символов) или внутренними (специальные типы слоев, запросы, макросы, карты, макеты печати и т. д.).

При создании нового проекта необходимо подключить или создать новые слои. Векторные слои (содержащие точечные, линейные, площадные объекты) могут быть созданы непосредственно в среде ГИС или в других программных средах (например, это может быть чертеж в обменном или двоичном формате AutoCAD). В качестве слоев могут быть загружены растровые изображения различных форматов (как правило, используемых в цифровой картографии). С каждым векторным слоем может быть связана таблица характеристик, хранимая с векторным слоем, и набор таблиц с атрибутивными (тематическими) данными, хранимый во внешней СУБД.

Для каждого слоя можно определить следующие объекты базы данных:

- Запросы к атрибутивным таблицам;

- Темы (варианты тематического картографирования слоя);
- Формы представления справочной информации об объектах;
- Диаграммы (представления результатов в виде различных графиков);
- Макросы — внешние исполняемые программы или внутренние функции ГИС (задаются пользователем для карты в целом, для слоя или для отдельных объектов).

Слои (или покрытия) объединяются в цифровые карты. Карты могут не поддерживать в своей структуре покрытия, но в этом случае берут часть или все функции покрытий на себя. В рамках одного проекта может быть создано неограниченное количество карт. Карты могут быть связаны друг с другом как вертикально, так и горизонтально. Работая внутри карты, можно добавлять слои, создавать и редактировать пространственные объекты, в том числе с соблюдением топологии, осуществлять работу с таблицами (записывать в таблицы результаты измерений по карте, производить изменение структуры, сортировку, редактирование, выборки вручную, запросы с отображением результатов выборок на карте).

Дополнительные возможности управления дает панель управления слоями (Легенда) карты, на которой представлены все слои. Здесь можно:

- включать и выключать отображение слоя;
- присваивать слоям диапазон масштабов, при которых они будут видимыми;
- удалять слои из списка слоев, отнесенных к карте;
- перемещать слои в списке (и, одновременно, в порядке воспроизведения) вверх или вниз;
- изменять тематическую классификацию для слоев и т. д.

Карты могут быть подготовлены к печати в виде макетов (Layouts) печати. В состав макета печати можно включить любые карты и их

легенды. В макет печати могут входить также тексты, таблицы, графики, растровые изображения и др.

Любую карту, макет печати, таблицы, темы, запросы, диаграммы, макросы — можно сохранить в проекте для последующего использования. Одним из важных для реализации роли ГИС, интегрирующей различные информационные среды, является контекстная ориентированность рабочей среды.

Это значит, что весь интерфейс ГИС (набор меню, панелей и инструментов, реакции по правой клавише мыши и т. д.) качественно меняется в зависимости от того, с каким объектом вы работаете в данный момент.

Выделяют три типа объектов, к которым можно отнести любой имеющийся на карте:

- Точечный объект. Объект, обозначенный точкой, поскольку его размеры слишком малы, чтобы можно было отразить его форму (границы, площадь) в масштабе карты. Может также представлять некий условный объект, не имеющий размеров, например отметку высот.
- Линейный объект. Объект, локализованный в виде линии, поскольку его ширина не выражается в масштабе карты-источника, — река, дорога и т. д. Может также представлять некий условный объект, например, границу.
- Полигональный объект. Объект, имеющий площадь, выражающуюся в масштабе карты-источника. Определяется замкнутым контуром и его внутренней областью, например лес, озеро.

1.2 Модели пространственных данных

Пространственные данные (или геоданные) можно определить как данные о географических объектах или явлениях, фиксирующие их местоположение и/или распределение в системе координат, привязанной к поверхности Земли [3].

Форматы геопространственных данных:

- Векторный формат (самый древний и самый распространённый стандарт. Файл в векторном формате фактически представляет собой набор файлов: в одном файле хранятся геометрические данные, в другом — специальные атрибуты данных и т. п.).
- GeoPackage (более новый стандарт, набирающий популярность. Данные хранятся в одном небольшом по размеру файле, реализованном в виде контейнера базы данных SQLite).
- GeoJSON (использует стандартный текстовый формат JSON).

1.2.1 Растровые модели

Растровая модель описывает не объекты, а пространственное распределение некоторой (выбранной исследователем) характеристики. Пространство разбивается регулярной сеткой ячеек, в каждой ячейке фиксируется значение исследуемого параметра (путем статистического осреднения, семплирования в центре ячейки и т.п.). Растровые данные могут быть как количественными (например, поле температуры), так и качественными (например, растр классифицированного снимка, каждая ячейка которого фиксирует принадлежность к тому или иному типу объекта). Таким образом, растровая модель является пространственно-ориентированной (или феномен-ориентированной).

Работа с растровыми данными в целом гораздо проще, чем работа с векторными объектами. Это обусловлено в том числе жесткой сеточной

структурой данных, которая предоставляет не так много свободы в различных сценариях обработки данных [3].

Растр представляет из себя матрицу значений. Каждой ячейке матрицы соответствует прямоугольная пространственная область фиксированного размера, которая называется пикселем. Различают растры непрерывные и категориальные (классифицированные).

В отличие от векторных данных, которые требуют указания координат для каждой вершины, регулярно-ячеистый характер растровой модели позволяет вычислять координаты пикселей на основе их индексов. Поэтому фактически растровые данные хранятся в виде линейно упорядоченного списка значений (raster values) и описания геометрии растра (raster geometry).

Геометрия растра определяет, где именно располагаются в пространстве пиксели растра и может быть описана путем указания следующих компонент (см. Таблица 2).

Таблица 2 - Компоненты растровых данных

Параметр	Назначение
NCOLS	Количество столбцов
NROWS	Количество строк
XLLCENTER	Координата X центра левой нижней ячейки растра
YLLCENTER	Координата Y центра левой нижней ячейки растра
CELLSIZE	Размер ячейки

Иногда вместо параметров XLLCENTER/YLLCENTER указываются XLLCORNER/YLLCORNER, которые кодируют координаты левого нижнего угла, а не центра левой нижней ячейки растра. Если геометрия растра характеризуется анизотропией, то вместо одного значения

CELLSIZE могут быть указаны разные размеры ячеек по осям координат CELLSIZEX и CELLSIZEY [3].

Модель растровых данных состоит из строк и столбцов пикселей одинакового размера, соединенных между собой для формирования плоской поверхности. Эти пиксели используются в качестве строительных блоков для создания точек, линий, областей, сетей и поверхностей. Большинство доступных растровых данных ГИС построено на квадратном пикселе (рис.6). Эти квадраты обычно преобразуются в прямоугольники различных измерений, если модель данных преобразуется из одной проекции в другую (например, из координат плоскости государства в координаты UTM [Universal Transverse Mercator]) [10].



Рис.6. Обычная растровая графика, используемая в приложениях ГИС: аэрофотоснимок (слева) и ЦМР США (справа)

Местоположение отдельной ячейки определяется точкой геопривязки растра, разрешением ячейки, номером колонки и номером ряда ячейки на растре. Точкой геопривязки растра может быть нижний левый или верхний левый угол растра; это условие учитывается программным продуктом.

Расстояние между двумя ячейками растра есть функция местоположения их средних точек и разрешения ячеек. В растровой модели в ГИС определение длин вертикальных или горизонтальных линий

проводится путем подсчета количества ячеек, по которым проходит линия, и умножением их на размер одной ячейки. Если линия ориентирована по диагоналям ячеек, то необходимо выполнить произведение количества ячеек на размер ячейки и на $\sqrt{2}$.

Площадь выбранных объектов на растре вычисляется как произведение количества ячеек на площадь отдельной ячейки.

Для хранения растровых данных используют несколько подходов, основанных на различных методах сжатия. Методы сжатия растровых данных работают внутри подсистемы хранения и редактирования ГИС, но они могут вызываться и напрямую на этапе ввода информации в ГИС.

1.2.2 Векторные модели

Векторная модель пространственных данных включает описание координатных данных пространственных объектов и, опционально, топологических отношений между ними. Векторные данные фиксируют местоположение и форму объектов в виде геометрических примитивов, таких как точки, линии, полигоны, объемные тела. Выбор модели объекта (например, представить город точкой или полигоном) зависит от масштаба анализа и целей исследования. Векторная модель данных является объектно-ориентированной.

Векторные модели данных используют точки и связанные с ними пары координат X , Y для представления вершин пространственных объектов, как если бы они были нарисованы на карте вручную. Атрибуты данных этих функций затем сохраняются в отдельной системе управления базами данных. Пространственная информация и информация об атрибутах для этих моделей связаны посредством простого идентификационного номера, который присваивается каждому объекту на карте.

В географических информационных системах (ГИС) существуют три основных типа вектора: точки, линии и многоугольники (рис.1). Точки — это нульмерные объекты, которые содержат только одну пару координат. Точки обычно используются для моделирования отдельных объектов, таких как здания, колодцы, опоры линий электропередач, места отбора проб и т. Д. Точки имеют только свойство местоположения. Другие типы точечных объектов включают в себя узел и вершину, В частности, точка — это автономный объект, а узел — это топологический узел, представляющий общую пару координат X, Y между пересекающимися линиями и / или многоугольниками. Вершины определяются как каждый изгиб вдоль линии или многоугольника, который не является пересечением линий или многоугольников.

Линии — это одномерные элементы, состоящие из нескольких явно связанных точек. Линии используются для представления линейных объектов, таких как дороги, ручьи, разломы, границы и т. Д. Линии имеют свойство длины. Линии, которые напрямую соединяют два узла, иногда называют цепями, ребрами, сегментами или дугами.

Полигоны — это двумерные объекты, созданные несколькими линиями, которые возвращаются назад для создания «закрытого» объекта. В случае многоугольников первая пара координат (точка) на первом отрезке линии совпадает с последней парой координат на последнем отрезке линии. Полигоны используются для представления таких объектов, как границы города, геологические формации, озера, ассоциации почв, растительные сообщества и т.д. Полигоны имеют свойства площади и периметра. Полигоны также называют областями.

Векторные данные

Simple Features (официально Simple Features Access) — это стандарт OGC 06-103, разработанный Open Geospatial Consortium (OGC) и реализованный также в виде международного стандарта ISO 19125,

который определяет общую модель хранения и доступа к векторным объектам (точка, линия, многоугольник, мульти точечные, мультилинии и т. д.), в географических информационных системах [3].

Геометрическое представление пространственных объектов базируется на следующих принципах:

- все геометрии состоят из точек;
- точки являются координатами в 2-, 3- или 4-мерном пространстве;
- все точки в геометрии имеют одинаковую размерность.

В дополнение к координатам X и Y имеются два дополнительных дополнительных параметра:

- координата Z, обозначающая высоту;
- координата M, обозначающая некоторую меру, связанную с точкой, а не с признаком в целом.

Всего стандарт Simple Features включает в себя 17 типов геометрий. Из них наиболее употребительными являются следующие 7 (см. Таблица 1).

Таблица 1 - Типа векторной геометрии

Тип	Описание
POINT	нуль-мерная геометрия, содержащая одну точку
LINestring	последовательность точек, соединенных прямыми, несамопересекающимися отрезками; одномерная геометрия
POLYGON	геометрия с положительной площадью (двумерная); последовательность точек, отрезки между которыми формируют замкнутое кольцо без самопересечений; первое кольцо является внешним, ноль и более остальных колец представляют дырки внутри полигона
MULTIPOINT	множество точек; геометрия типа MULTIPOINT называется простой если ни одна пара точек в MULTIPOINT не совпадает
MULTILINestring	множество линий
MULTIPOLYGON	множество полигонов

Примеры различных видов геометрий представлены на рисунке 10:

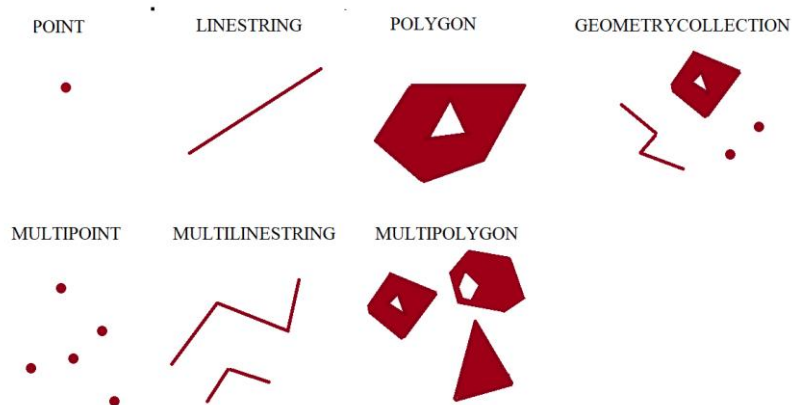


Рисунок 10 - Виды геометрий

1.3 Задачи анализа пространственных данных

Анализ пространственных данных представляет собой процесс выявления закономерностей пространственного распределения объектов и пространственных взаимоотношений между объектами исследования с использованием средств геоинформационных систем [1].

Задачи анализа пространственных данных.

1) Анализ местоположения. Этой категории соответствует пространственный запрос: что существует в конкретном месте на поверхности Земли? Чтобы увидеть, где расположен и как выглядит объект, который интересует, используются карты. На карте наглядно представлен характер пространственного распределения объектов, а это позволяет проявить связи между ними и лучше понять исследуемую область. Только увидев местоположения объектов, можно понять некоторые причины пространственных взаимосвязей. Для того, чтобы исследовать закономерности в распределении данных, нужно

определенным образом отобразить объекты, которые исследуются, опираясь на значения их характеристик. Например, а) эколог может оценить влияние особенностей рельефа или других факторов на пространственное распределение растительных сообществ, используя картографические данные; б) градостроитель планирует размещение определенного объекта в соответствии с генеральным планом города, планом существующей застройки и планом инженерной инфраструктуры; в) отдел милиции создает карту распределения преступлений разного типа, чтобы видеть, повторяются ли они в отдельных районах.

2) Удовлетворение пространственных условий. Этой категории соответствует пространственный запрос: где удовлетворяются конкретные пространственные условия? Простой запрос о местоположении объекта состоит из одного условия. Для получения ответа достаточно выполнить одну штатную операцию. Сложный запрос о местоположении объекта может включать определенный набор условий. Для получения ответа уже требуется использование ряда операций пространственного анализа. Например, а) где находится площадка для строительства площадью 2 га. в пределах до 200 м. от дороги районного значения, с грунтами несущей способности до I кг. на кв.см.? б) обосновать местоположение торгового, учебного заведения или бизнес-центра с учетом многих, в т.ч. пространственных факторов; в) найти оптимальную трассу трубопровода или путепровода, который проектируется.

3) Временной анализ. Этой категории соответствует запрос: что пространственно изменилось за указанный период? Ответ на этот вопрос представляет собой попытку определить изменения, произошедшие в пространстве и времени, тенденции этих изменений на определенной территории. Например, какова тенденция распространения гриппа в городе, какие новые объекты построены за последний год, каков рост

урбанизированных территорий? Сохраняя и сопоставляя карты разных дат, ГИС может выполнять временной анализ.

4) Выявление структуры. Этой категории соответствует пространственный запрос: какие пространственные структуры или распределения существуют? Например, сколько имеется аномалий, не соответствующих нормальному распределению, и где они находятся? Каково распределение населения в городе? Какие участки дороги являются наиболее опасными? Каково распределение стоимости недвижимости на территории? Выделение пространственных структур — это сложный вопрос, требующий применения арсенала мощных средств пространственного анализа.

5) Оценка различных сценариев. Сценарий потенциала является результатом вопросов типа "Что произойдет, если...". Например, что произойдет, если интенсивность дождя будет критической? Каковы издержки, если улицу расширить на 14 м.? Как изменится транспортное сообщение, если убрать трамвай с улицы Пушкинской? В таких случаях пользователь использует модель для прогнозирования и 44 карты потенциального воздействия. Применение такой модели позволяет построить гипотетическую ситуацию и прогнозировать развитие и следствия социологических и экономических ситуаций, стихийных бедствий и аварий естественного техногенного характера в пространстве и времени.

Базовые операции пространственного анализа:

1. Представление географического распределения данных, то есть отображение данных о пространственно-распределенных объектах на цифровой географической карте с учетом пространственной информации об объектах исследования.

2. Построение и выполнение запроса или получение выборки данных из базы географических данных. Запросы позволяют найти и рассмотреть определенные объекты. Существует два вида запросов в ГИС: атрибутивные и пространственные запросы. Атрибутивные запросы, называемые также не пространственными, ведут поиск объектов на основании значений их атрибутов. Запросы местоположения, называемые также пространственными запросами, ведут поиск объектов по пространственным характеристикам.

3. Поиск ближайших объектов к заданному и создание буферной зоны вокруг этого объекта.

4. Наложение различных пространственных (тематических) слоев объектов. Путем наложения одного набора объектов на другой можно получить новую информацию о закономерностях пространственного размещения объектов.

При проведении пространственного анализа результаты одной операции можно использовать в качестве исходных данных для другой, тем самым обеспечивая проведение сложного анализа, основанного на сочетании различных операций [1].

1.4 Методы анализа геопространственных данных

Наиболее распространенной проблемой при работе с пространственно распределенными данными является получение пространственной оценки. Для этого необходимы комплексные исследования на основе методов геостатистики – статистики пространственно распределенной (региональной) информации, к основным задачам которой относятся:

Геостатистические методы, (например, кригинг), базируются на математических и на статистических функциях (т.е. используют статистические свойства опорных точек), которые могут быть

использованы для построения поверхностей и оценки точности прогнозов.

Геостатистические методы количественно определяют пространственную корреляцию между опорными точками и учитывают расположение опорных точек в пространстве вокруг искомой точки [12]. Геостатистика изначально была связана со статистикой Земли, например, в области географии и геологии. Теперь геостатистика широко используется во многих областях и включает в себя пространственную статистику. Первоначально в пространственной статистике геостатистика была синонимом кригинга, который представляет собой статистическую версию интерполяции. Нынешнее определение расширилось и включает не только кригинг, но и многие другие методы интерполяции [13].

Географически взвешенная регрессия (ГВР) – один из нескольких методов пространственного регрессионного анализа, используемого в географии и других дисциплинах. Метод ГВР оценивает локальную модель переменной или процесса, которые вы прогнозируете или изучаете, применяя уравнение регрессии к каждому пространственному объекту в наборе данных. Географически взвешенная регрессия создает отдельные уравнения путем включения зависимых и независимых переменных объектов, попадающих в пределы окрестности каждого целевого объекта. Форма и экстенд каждой окрестности анализируется на основании входных параметров. Географически взвешенная регрессия должна быть применена к наборам данных с несколькими сотнями объектов. Для небольших наборов данных этот метод не пригоден, кроме того, он не работает с мультиточечными данными.

Географически взвешенная регрессия используется для решения множества вопросов, например [9]:

- Имеется ли связь между уровнем образования и доходом в изучаемой области?

- Возрастает ли заболеваемость определенными инфекциями по мере приближения к водным объектам?
- Какие ключевые переменные объясняют высокую частоту лесных пожаров?
- Какие среды обитания следует защитить, чтобы провести реинтродукцию исчезающих видов животных?

1.5 Картографические сервисы и пространственные данные

Существует широкое множество картографических сервисов, нацеленных на широкий спектр различных задач. Некоторые из них, наиболее известные, представлены ниже.

Google Maps - Картографическая веб-платформа и потребительское приложение, предлагаемое Google. Он предлагает спутниковые снимки, аэрофотосъемку, карты улиц, интерактивные панорамные виды улиц на 360 ° (Street View), условия движения в реальном времени и планирование маршрутов для путешествий пешком, на машине, по воздуху (в бета-версии) и на общественном транспорте (см. Рисунок 1).

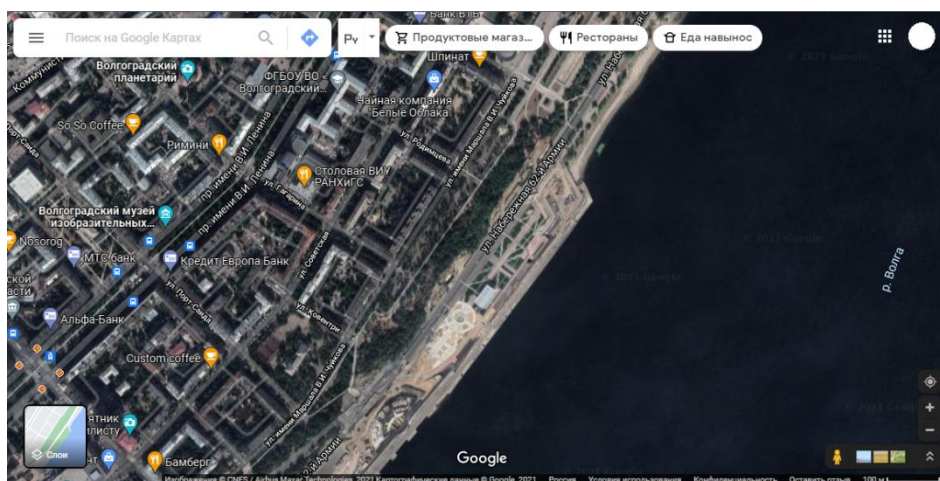


Рисунок 1 - Карта Google Maps

Microsoft Bing Maps - Картографический сервис Microsoft, часть портала Bing (см. Рисунок 2).

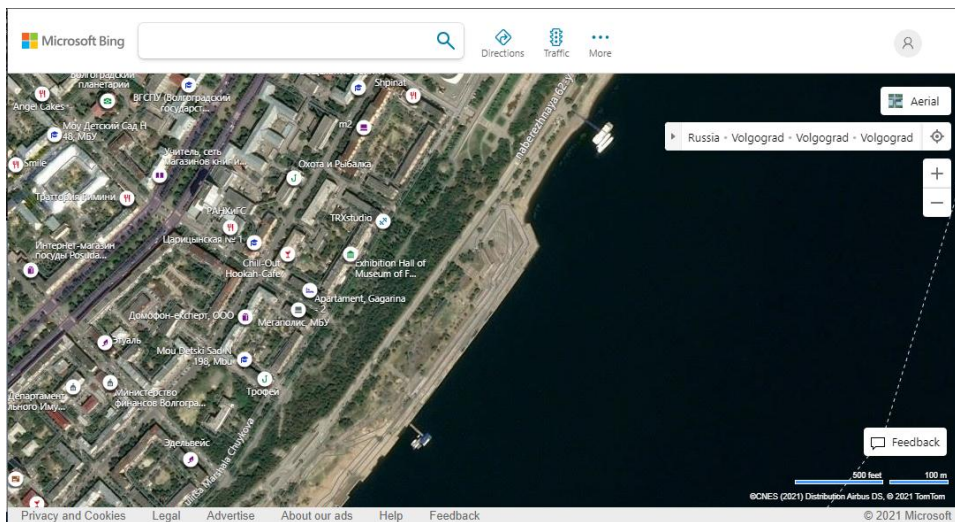


Рисунок 2 - Карта Microsoft Bing Maps

Яндекс.Карты - Российский картографический веб-сервис, разработанный Яндексом. Сервис предоставляет подробные карты всего мира. Включает в себя поиск, информацию о пробках, маршрутах и панорамах улиц. Из последних особенностей, в мобильные приложения добавили возможность просмотра общественного транспорта в режиме реального времени и встроили навигатор [1] (см. Рисунок 3).

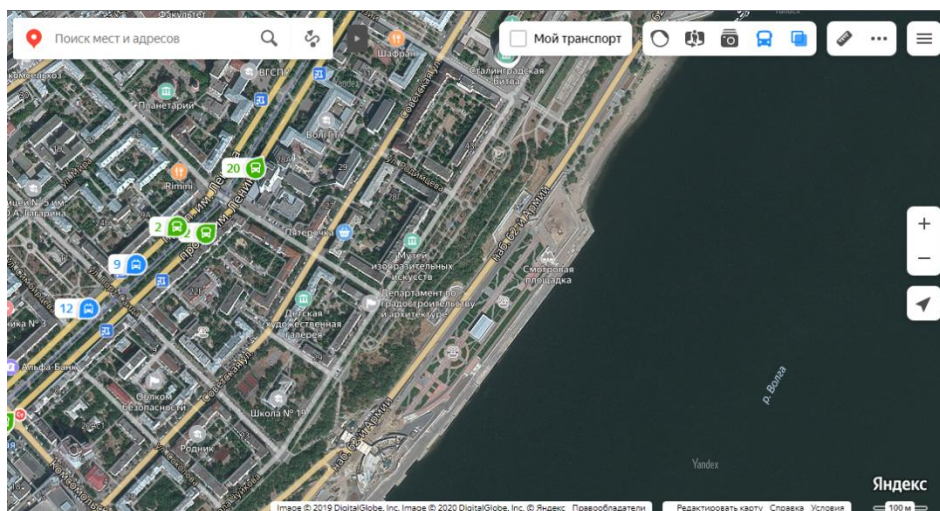


Рисунок 3 - Карта Яндекс.Карты

Apple Maps - Картографический сервис от компании Apple для операционных систем iOS и macOS. Карты от Apple по точности данных

различных городов проиграют всем остальным конкурентам [1] (см. Рисунок 4).

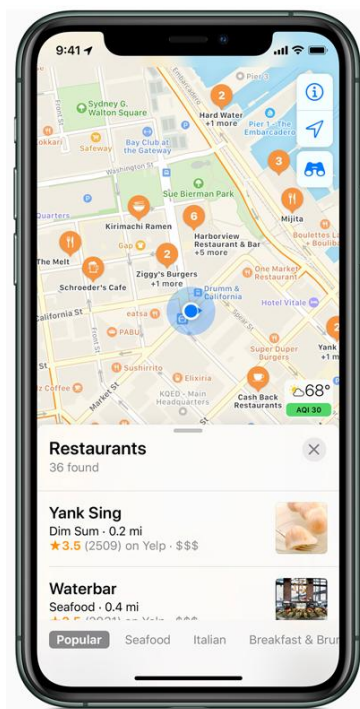


Рисунок 4 - Карта Apple Maps [2]

2ГИС - Российская местная поисковая компания, которая разрабатывает цифровые карты и путеводители по городам России, Казахстана, Италии, Чехии, Чили, ОАЭ, Узбекистана, Кыргызстана, Кипра, Азербайджана и Украины. Карты, в первую очередь нацелены на качественную работу при отсутствии или плохом интернет-соединении [1] (см. Рисунок 5).

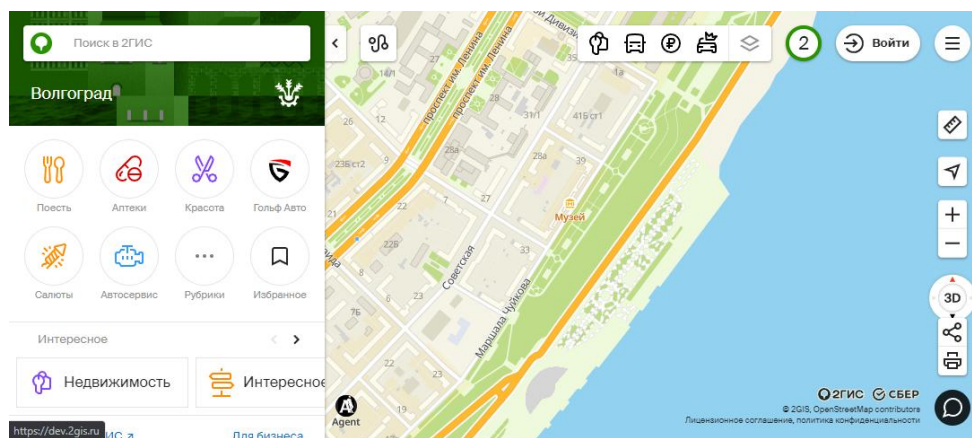


Рисунок 5 - Карта 2ГИС

OpenStreetMap - совместный веб-картографический проект по созданию бесплатной редактируемой географической базы данных мира. Карты очень детализированы и во многих регионах обходят всех своих конкурентов. Именно на базе OpenStreetMap работают многие сервисы, такие как OsmAnd, MAPS.ME и многие другие [1] (см. Рисунок 6).

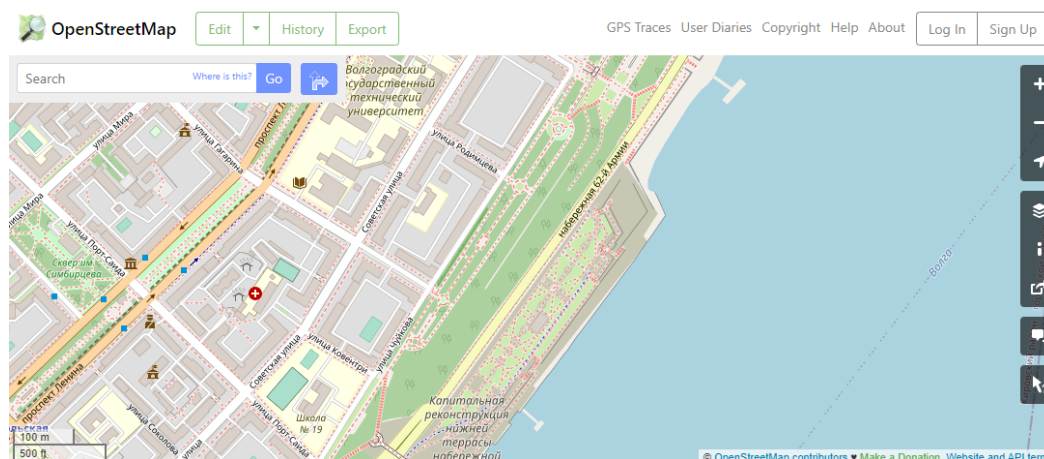


Рисунок 6 - Карта OpenStreetMap

Wikimapia - проект географической онлайн-энциклопедии. Проект реализует интерактивную «интерактивную» веб-карту, которая использует Карты Google с географически привязанной вики-системой, с целью отметить и описать все географические объекты в мире (см. Рисунок 7).

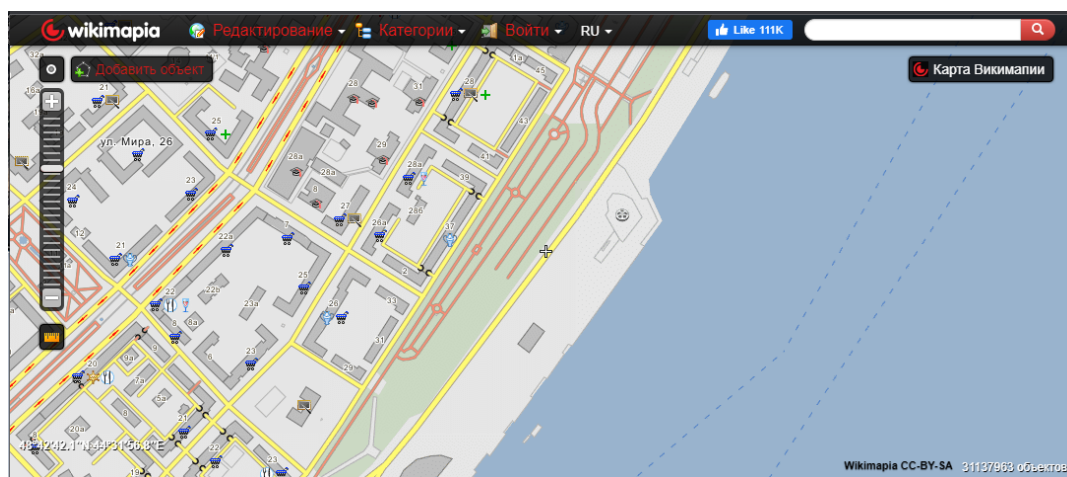


Рисунок 7 - Карта Wikimapia

Qwant Maps - Картографический сервис от Qwant. Относительно недавно Qwant закрыл свои поисковик и карты для ряда регионов [1] (см. Рисунок 8).

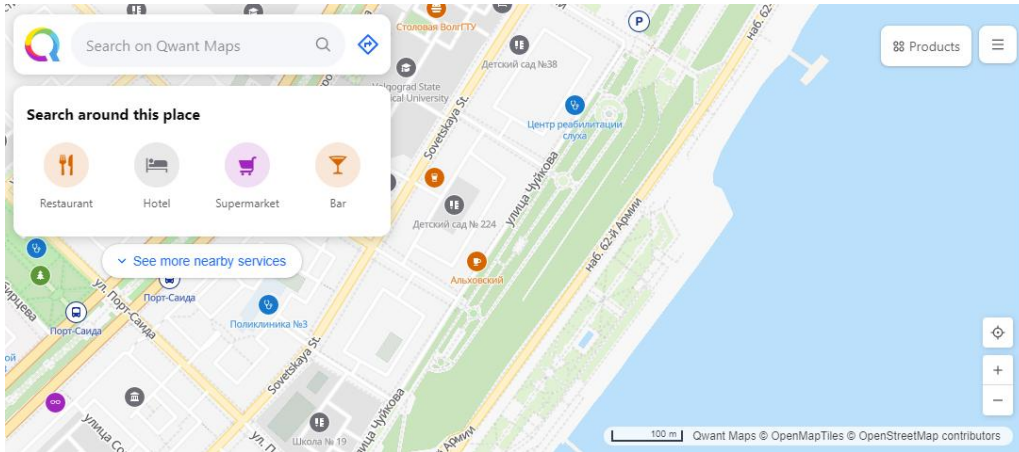


Рисунок 8 - Карта Qwant Maps

HERE WeGo - Веб-служба картографии и навигации, управляемая Here Technologies . Первоначально носившие имена «smart2go», «Ovi Maps», «Nokia Maps» и «HERE Maps». Разрабатывались Nokia и долгое время предустанавливались по умолчанию в устройства от Nokia. Учитывая прошлое, карты очень хорошо развиваются с уклоном в автомобильную навигацию. В ряде городов Европы детализированы даже пешеходные дорожки и дорогие, которые отсутствуют на других картах [1] (см. Рисунок 9).

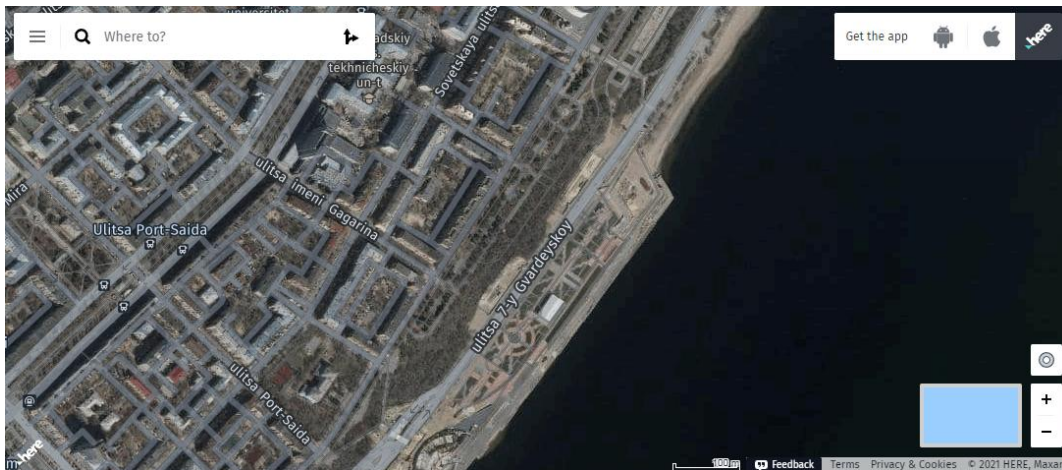


Рисунок 9 - Карта HERE WeGo

1.6 Управление данными в ГИС (на примере OpenStreetMap)

Базовым элементом структуры данных OSM является точка (node) с географическими координатами – широтой (latitude) и долготой (longitude). Точка может быть отдельным объектом или входить в состав других объектов (линий, полигонов, мультиполигонов).

Линия (way) – это последовательности точек. Менять последовательность нельзя. Несколько линий логически могут представлять один объект. Например, длинная дорога состоит из нескольких линий.

Полигон – это замкнутая линия, у которой совпадают первая и последняя точки. Полигон не является самостоятельным элементом OSM.

Отношение – это логическое объединение точек, линий и других отношений в единый объект. Каждый объект имеет набор тегов, описывающих его. Тег (tag) определён как $k = \text{«ключ»}$ $v = \text{«значение»}$. Если элемент не имеет тегов, то он не является объектом, а входит в состав других объектов, как и некоторые элементы с тегами.

В целом, всю структуру данных OSM можно представить схематично (см. Рисунок 11).

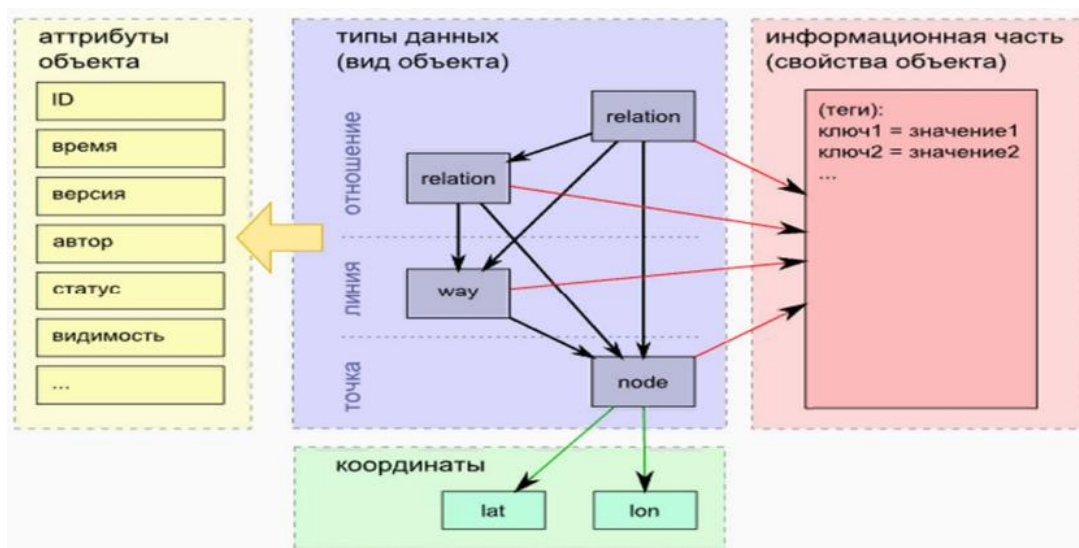


Рисунок 11 – Структура данных OpenStreetMap [5]

Все данные можно разбить на три группы:

1) типы данных (вид объекта) – основная группа, описывающая геометрические типы объектов на карте и их иерархию по отношению друг к другу, где основная единица точка (node) обязательно включает в себя координаты (lat, lon) и может входить во все остальные типы объектов;

2) атрибуты объектов – описывает формальные данные об авторах, времени добавления/правки,

статусе, видимости, организует структуру типов объектов за счет присваивания к ним ID;

3) информационная часть (свойства объектов) – описательная группа, формирующая представление об объекте и его свойствах путем добавления определенных тегов к объекту.

Существует ряд решений для парсинга данных из osm-файла. Например, библиотека «BasicOSMParser» (доступна на GitHub в репозитории аккаунта PanierAvide/BasicOSMParser) – это набор классов Java, позволяющий анализировать необработанные XML-файлы OSM. Также библиотека позволяет экспортировать данные в виде файлов CSV.

Он использует Java по умолчанию SAX parser (org.xml.sax). Тесты приложения используют платформу JUnit 4 framework.

Библиотека «osm4routing» (доступна на GitHub в репозитории аккаунта Tristramg/osm4routing) – инструмент, предоставляющий парсер данных из OpenStreetMap, чтобы превратить их в узлы-ребра, приспособлено для применений трассы. Входными данными является XML-файл OpenStreetMap. Выходными данными библиотеки могут быть CSV-файл или база данных (postgres, mysql, sqlite, postgres).

Библиотека «Libosmium» (доступна на GitHub в репозитории аккаунта osmcode/libosmium) – быстрая и гибкая библиотека C++ для работы с данными OpenStreetMap. Библиотека может прочитать OSM XML данные или в двоичном формате (PBF) и может вызывать различные обработчики для каждого OSM-объекта. Libosmium доступен под лицензией на программное обеспечение Boost.

Информационным обеспечением основной программы и других модулей является файл формата OSM XML (экспортированные данные участка карты OpenStreetMap). В основном файле содержится список экземпляров примитивных данных, таких как узлы, пути и отношения («nodes», «ways» и «relations»), а также их описание и связи.

Для удобной работы с данными файла часто обращаются к библиотеке NetTopologySuite и приводят данные из файла к структуре (точек, линий и отношений).

Для преобразования данных из объектов типа Geometry в GeoJSON используется GeoJsonWriter – средство библиотеки NetTopologySuite.

Соотношение структур полученных объектов из файла OSM XML и приведенных к структуре Geometry при помощи библиотеки NetTopologySuite представлено в таблице 3.

Таблица 3 – Соотношение структур объектов

Объект OSM XML	Наследуемый объект NetTopologySuite.Geometries
Точка (node)	Point
Линия (way)	LineString
Замкнутая линий (closed way)	LinearRing
Отношение (relation)	GeometryCollection

Для каждого из типов данных необходимо хранить его теги, описывающие объект карты и его свойства. Для типа «relation» помимо тегов необходимо хранить ещё и роль каждого из объектов.

Структура данных, которая должна быть получена после преобразования и использоваться в дальнейшем другими модулями и базовыми компонентами платформы, представлена на рисунке 12.

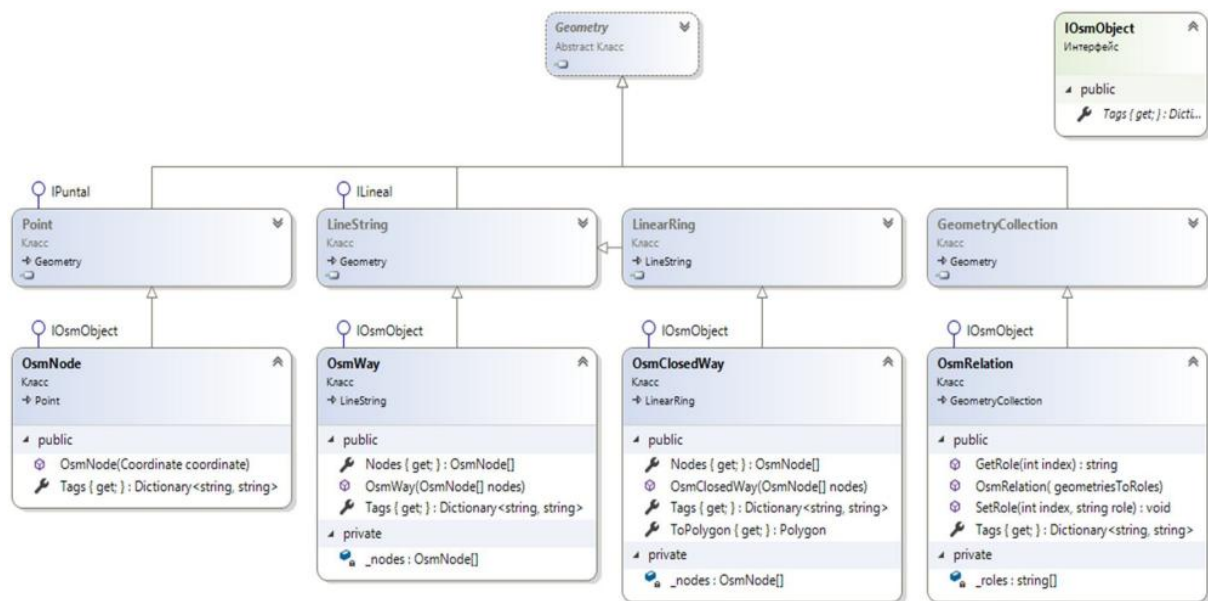


Рисунок 12 – Диаграмма классов OSM

1.7 Моделирование на основе пространственных данных

Оптимизация условий для осуществления процессов обеспечения жизнедеятельности с использованием сети городской инфраструктуры является сегодня одним из важнейших вопросов для разрастающихся

городов. Принимая решение о развитии инфраструктуры и распределения ресурсов, городские администрации и планировщики ориентируются на исследования данных статистики, социологических опросов, данные специализированных сенсоров-счётчиков, камер видеофиксации и другие способы учёта перемещения людей в пространстве (пешее передвижение, поездки на личных автомобилях или же общественном транспорте). На основе этих данных строятся модели и прогнозы развития ситуации. Однако проведение подобных исследований и их последующий анализ являются крайне дорогостоящими и времязатратными проектами [1]. Поэтому зачастую между проведением комплексного анализа городской транспортной системы проходит несколько лет, что делает неадекватной созданные модели и в итоге приводит к множественным локальным издержкам. Попытки сократить затраты на мониторинг путем только опросов или датчиков в общественном транспорте создают фрагментарную картину, т.к. охватывают лишь часть населения или территории.

Широкий спектр задач, связанных с исследованием потоков в городской среде, и высокая стоимость данных для их исследования при неизбежной необходимости их экстраполяции для получения полной картины ситуации в каждый момент времени требуют создания решений для комплексного прогнозного моделирования [2].

Изучение законов развития сложных систем лежит в основе формирования стратегий, позволяющих целенаправленно изменять их поведение и создавать новые способы управления их функционированием, обеспечивающие достижение требуемых значений показателей качества. Единственно возможным инструментом изучения этих законов является моделирование. Моделирование не гарантирует полную защиту от возможных ошибок при принятии решений, но позволяет выявить различные проблемы, проанализировать последствия принятия решений, прогнозировать развитие ситуаций.

В современных условиях появляется все больше возможностей для получения актуальной информации о городских процессах и поведении людей.

С 2015 года Правительство Москвы потратило более 500 млн. рублей на закупку геоданных о передвижениях людей у мобильных операторов. Информация используется для оптимизации обеспечения процессов жизнедеятельности и улучшения городской среды [10].

Многоагентные системы интегрируют в себе самые передовые достижения в области программного обеспечения, систем искусственного интеллекта и распределенных информационных систем, демонстрируя принципиально новые качества самоорганизации и интеллектуального поведения.

При наличии неполных данных об окружающей среде, агентах и их взаимодействии друг с другом, моделирование с использованием агентов является наиболее перспективным вариантом. Описывая базовые правила взаимодействия и необходимые свойства агентов мы получаем адекватную в некотором приближении модель, которая соответствует моделируемой части города и служит для выявления его слабых мест и закономерностей.

2 Существующие решения на основе мультиагентного подхода

Мультиагентная система представляет собой систему из нескольких взаимодействующих интеллектуальных агентов (программ) [6, 7, 11]. Сущность мультиагентного подхода состоит в сведении исходной сложной задачи в совокупность простых задач. При этом решение каждой «простой» задачи осуществляется специальной программой, называемой агентом.

Агенты как программные сущности способны воспринимать окружающую среду и реагировать в зависимости от ситуации, в которой

они находятся [12]. Они ориентированы на цель, т.е. они получают задачи и выполняют их, взаимодействуя между собой и с окружающей средой.

Применение мультиагентных систем широко распространено в самых разных отраслях науки и техники. Математика, биология, транспортная система, социально-экономические науки, планирование, прогнозирование и мн. др. представляют собой сферы применения различных мультиагентных систем.

Одним из направлений использования мультиагентных решений являются социальные науки. Таким образом может решаться проблема моделирования поведения человека в различных ситуациях. К примеру, для выявления, по каким причинам возникает скопление людей в общественных местах, профессором Такахаши (руководителем кафедры разработки промышленных систем и систем управления Университета Васэда) и Fujitsu Laboratories была разработана технология, основанная на мультиагентном подходе. Система автоматически анализирует факторы, которые привели к образованию большого скопления людей, на основании результатов моделирования поведения человека [5].

Для оценки поведения людей в случае локальных непредвиденных обстоятельств была разработана агентная модель TOXI-CITY. Ее цель – вычислить минимальное количество людей, которые знают, что нужно делать в случае непредвиденной ситуации и максимальное количество людей, которое может спастись. Моделирование показывает, что начальное количество агентов и пространственная конфигурация окружения сильно влияют на конечные показатели выживаемости. В результате авторы модели установили нелинейную закономерность: показатели выживаемости увеличиваются, когда показатели хорошо информированных агентов низкие (ниже 30%), тогда как увеличение доли информированных агентов (с 70%), по-видимому, лишь незначительно повышает шансы на выживание [3].

Рассматриваемый подход также применяется для моделирования динамики мнений в виртуальном сообществе. Модель, разработанная профессором ИТМО (Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики) Бухановским А.В., имитирует распространение информации о политических компаниях, социальных сетях и обучающих онлайн-сообществах. Модель воспроизводит ожидаемую динамику мнений с учетом влияния различных источников информации в течение жизненного цикла агентов. В работе было представлено два сценария, иллюстрирующие влияние противоречивых стратегий информационного воздействия на популяцию и поляризацию мнений на актуальную тему, «Информационная война» и «Мнение по актуальным темам» соответственно [13].

Мультиагентная реализация широко используется при планировании проектной деятельности. Одним из таких проектов является система для поддержки комплексного управления программными проектами. Она включает в себя мультиагентную систему и механизм аукциона для динамического планирования сроков и ресурсов проектов. Проекты имеют задачи, которые должны быть выполнены, и конкурируют за ресурсы и возможности. Система динамически выделяет ресурсы для проектов и решает, какие проекты принять или отклонить с учетом стоимости проекта, прибыльности и обратной связи о проекте [4].

Агентный подход очень часто находит свое применение в задачах моделирования и симуляции различных событий. Первым примером является подход к моделированию эвакуации пассажиров аварийного судна в штормовых условиях, основанный на силовой модели передвижения агентов, учитывающей влияние перемещений, скоростей и ускорений качки, разработанный профессором Бухановским. Используя данный подход, можно на основе имитационного моделирования оценить

время эвакуации в зависимости от условий шторма (интенсивность волнения, скорость хода, курсовой угол). Предложенный подход может использоваться в системах поддержки принятия решений по обеспечению безопасности мореплавания с целью планирования мероприятий, уменьшающих число возможных жертв как за счет внешних факторов (позиционирование судна относительно направления волн), так и внутренних (открыть или заблокировать двери, забалластировать отсеки для уменьшения крена и пр.) [14].

Другое использование мультиагентных систем для моделирования представлено в виде системы суперкомпьютерного моделирования критических явлений в сложных социальных системах. Данная система включает в себя агрегированную аналитическую модель для идентификации критических состояний, детализированную модель динамики сети на индивидуальном уровне, а также модель, уточняющую особенности топологической структуры комплексной сети. Разработанная применяется для моделирования распада криминальных сетей, быстрого распространения слухов в социальных сетях Интернет, эволюции финансовых сетей и распространения эпидемий [9].

Так же симуляция на основе агентного подхода реализована в системе, моделирующей распространение инфекционных заболеваний. Система позволяет в интерактивном режиме изучать различные альтернативные сценарии для поддержки принятия решений и предотвращения, прогнозирования и восстановления вспышки [15].

Еще одним применением мультиагентных систем является межъязыковое обогащение онтологий на основе многоагентной архитектуры. Прототип для предложенного подхода был реализован, чтобы обогатить несколько онтологий, используя текст на английском, арабском и немецком языках. Каждый агент ответственен за обогащение онтологии для определенного языка. Самой важной особенностью

является то, что каждый агент может обучать друг друга, используя predetermined схему коммуникации [16].

Теперь рассмотрим наиболее важную область применения в данном исследовании – транспорт.

Tangramob – система, основанная на мультиагентном моделировании, которая позволяет предсказать, как изменится движение транспорта в городе при изменении маршрутов перемещений. Это позволяет лицам, принимающим решения, оценить эффективность изменения мобильности и жизни жителей с учетом нынешней городской системы и введенными пользователем приоритетами. Система выполняет сравнительные эксперименты до, после и между изменениями в мобильности. Пользователи системы могут измерить воздействие смоделированного изменения в отношении: уровня городского движения, выбросов углекислого газа, пройденного расстояния, времени в пути, уровня землепользования, стоимости перемещения и уровня использования ресурсов [1]. Необходимым развитием проекта является увеличение учитываемых критериев.

SimMobility Freight – система, позволяющая моделировать городские грузовые операции. В основе системы лежит мультиагентная структура моделирования, учитывающая неоднородность городских грузовых агентов и их взаимодействие. К агентам относятся учреждения (поставщики, перевозчики и получатели) и водители грузовых транспортных средств. Решения агентов структурированы в трех временных решениях: стратегическом, тактическом и оперативном. Представление поставки имеет два разных уровня разрешения (микро или мезо), что позволяет проводить детальное или вычислительно эффективное моделирование [2]. Для улучшения проекта предполагается добавление новых моделей и альтернатива при их выборе.

На основе мультиагентного подхода было реализовано моделирование дорожного движения Амстердама. Для процесса моделирования требуются следующие данные: время отправления (продолжительность действия), режим путешествия (автомобиль / общественный транспорт), маршрут и пункт назначения; также необходимо составить ежедневный план действий для каждого агента [8]. В дальнейшем планируется расширение существующего набора принимаемых во внимание характеристик.

Еще одним транспортным проектом Нидерландов является эмпирическая агентная система моделирования для городского грузового транспорта (MASS-GT). С помощью данной системы специалисты по городскому планированию исследуют проблемы, связанные с повышением устойчивости городского грузового транспорта: снижением загруженности городов, обеспечением надежных окон доставки, снижением затрат на логистику, сокращением выбросов, повышением безопасности [17]. Текущим недостатком, который планируется исправить, является небольшое количество вариантов выбора моделей поведения.

Был разработан многоагентный подход к моделированию для оценки динамики городской логистики в условиях возможного наводнения и проведено предварительное исследование в Белу-Оризонти (Бразилия). Процесс планирования в реальном времени делают городское распределение товаров более эффективным с меньшим воздействием на городскую среду. Таким образом, авторы пришли к выводу, что городские логистические меры важны для снижения негативного воздействия городского грузового транспорта. [18].

Так же с помощью исследуемого подхода было разработано моделирование перемещения велосипедистов в Берлине. Были выбраны атрибуты инфраструктуры, которые влияют на поведение велосипедистов – время прохождения, склоны, тип велосипеда и поверхность тротуара.

Исследователи обнаружили, что поведение человека за велосипедом сильно зависит от личных и других качеств, например, женщины склонны избегать склонов больше, чем мужчины, и жители пригородной зоны сильнее придерживаются кратчайшего маршрута. Авторы предполагают, что подход также может быть расширен для учета взаимодействия между велосипедистами и автомобилистами [19].

Еще одним применением мультиагентного решения является интеллектуальная система обучения на основе нескольких агентов для беспилотных наземных транспортных средств (USV). Используя данную систему группового обучения, мы предоставляем подходы для изучения базы правил для мультиагентных систем, разработки учебной среды для имитации поведения кооперативных и конкурентных агентов и сбора данных о конкурентах из наблюдений агентов для задач автономного обучения. Для оценки результатов был использован сценарий завоевания островов, где две команды беспилотных наземных транспортных средств соревнуются в завоевании островов в окружающей среде. Результаты показывают, что подход к изучению базы знаний системы с несколькими USV применительно к обученной команде смог составить базу знаний для достижения лучшей производительности [20].

Исследователями из Китая разработан проект «умного туризма», который отчасти объединяет в себе транспортную и социально-экономическую сферы и представляет собой систему рекомендаций туристических предложений, а именно: дорожные карты, система бронирования отелей, бронирование авиабилетов и т. д. Предложенная система была разработана для того, чтобы рекомендовать туристический пакет клиентам, используя данные в реальном времени и методы гибридной фильтрации [21].

Также мультиагентная система была использована для моделирования на микроуровне процесса транспортировки топлива морем

в Китае. Агентами являются танкеры, пираты и тропические циклоны. Была построена многофункциональная модель основного энергетического канала Китая (M Eastdle East Rote) с взаимодействием на основе агентов. Результат симуляции показал, что для увеличения оборота нефти, необходимо увеличить число танкеров [22].

Рассмотрим реализацию перечисленных ранее проектов, базирующихся на основе мультиагентного подхода.

Использование готовых систем мультиагентного моделирования:

- MATSim (платформа с открытым исходным кодом для реализации крупномасштабного агентного моделирования транспорта.) [23] – Tangramobe, моделирование дорожного движения Амстердама, моделирование движения велосипедистов из Берлина и моделирование логистики в Белу-Оризонти;

- NetLogo (многоагентная программируемая среда моделирования) [24] – Toxi-city и моделирование перевозок топлива по морю в Китае;

- CLAVIRE (многопрофильная инструментально-технологическая платформа) [25] – моделирование критических явлений в сложных социальных системах и моделирование эвакуации пассажиров аварийного судна в штормовых условиях;

Разработка собственной системы:

- язык программирования Python – система моделирования для грузового транспорта в Нидерландах;

- язык программирования Java с использованием фреймворка JADE [26] – «умный туризм» и расширение онтологий;

- язык программирования Java с использованием библиотеки C++ Fuzzylite (A Fuzzy Logic Control Library) [27] – система обучения для беспилотных наземных транспортных средств;

- язык программирования C++ – SimMobility Freight.

Вопросы

1. Понятие геоинформационной системы (ГИС).
2. Базовые возможности ГИС.
3. Методы искусственного интеллекта в ГИС
4. Технологии сбора и анализа пространственных данных
5. Картографические сервисы.
6. Особенности растровой модели представления пространственных данных в ГИС.
7. Особенности векторной модели представления пространственных данных в ГИС.
8. Перечислите средства анализа данных в ГИС.
9. Проекционные преобразования в ГИС.
10. Приведите основные принципы разграфки топографических карт.
11. Охарактеризуйте основные принципы построения номенклатуры топографических карт.
12. Охарактеризуйте основные принципы построения триангуляционной модели местности.
13. Раскройте содержание технологии ведения атрибутивных данных ГИС.
14. Охарактеризуйте основные этапы развития ГИС.
15. Методы распознавания образов в ГИС.
16. Методы кластеризации в ГИС
17. Методы классификации в ГИС
18. Структура интеллектуальной ГИС
19. Визуализация в геоинформатике
20. Методы извлечения пространственных данных.
21. Особенности существующих картографических сервисов.
22. Структура данных в OpenStreetMap.

2. МЕТОДИЧЕСКИЕ УКАЗАНИЯ К ЛАБОРАТОРНЫМ РАБОТАМ

2.1 Лабораторная работа № 1. Модели пространственных данных

Векторные данные

Цель: Понимание векторных данных и их использования в ГИС

Ключевые слова: Вектор, Точка, Полилиния, Полигон, Вершина, Геометрия, Масштаб, Качество Данных, Условные Обозначения, Источник Данных.

В **векторных данных** местоположение объектов задается с помощью координат и математических формул. В геоинформационных системах используются специализированные форматы векторных данных, позволяющие хранить как **информацию о местоположении объектов**, так и их **атрибутивные (непространственные) характеристики**, состоящие из текстовой и числовой информации, **описывающей** каждый объект.

Любой географический объект может быть представлен набором графических примитивов: точек, линий, полигонов, выраженных соответствующими картографическими символами. Таким образом, векторная модель представляет собой коллаж графических объектов, физически присутствующих в изображении. Пространственные данные обычно хранятся в виде взаимосвязанных объектов, атрибутивная информация о которых позволяет составить целостную картину картографируемого пространства.

Дискретные объекты, каждый из которых может занимать в любой момент времени только одну определенную точку в пространстве, представляются в виде **точек** (рисунок 1). Считается, что такие объекты не имеют пространственной протяженности и ширины (внемасштабные, 0-

мерные пространственные объекты) и каждый из них может быть обозначен парой координат: X и Y (широта и долгота). Примерами могут служить мысы, отдельно стоящие деревья, маяки, отметки высот и т. д. Совокупность точечных объектов образует **точечный слой**. (Дополнительно в описание точечного объекта может быть включена координата Z . Такие объекты часто называются **2.5D объектами**, т.к. они описывают только высоту или только глубину точки, но не оба параметра одновременно).

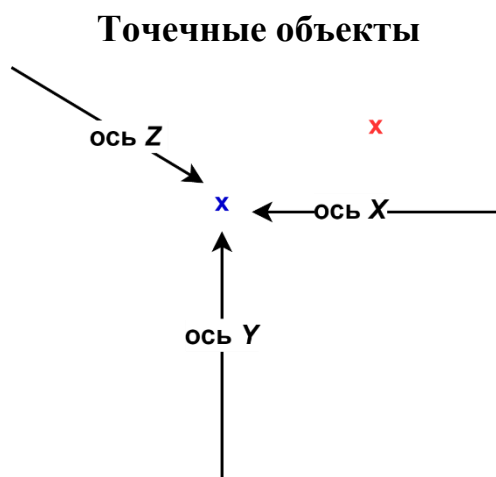


Рисунок 1: Точечный объект, описываемый координатами X , Y и (возможно) Z . Атрибуты описывают каждую точку.

Первая вещь, которую надо отметить, говоря о точках, – это условность выбора точечного представления объекта и его зависимость от масштаба. В качестве примера рассмотрим города. Если у Вас мелкомасштабная карта (т.е. она покрывает большую площадь), целесообразно будет представить города в виде точек. Тем не менее, в случае увеличения масштаба (приближения) лучше показать границы городов в виде полигонов. При выборе точек для представления объектов необходимо руководствоваться масштабом карты (как мелко показаны объекты), удобством (проще и быстрее поставить точку, нежели нарисовать полигон) и типом объектов (такие объекты, как телефонные

столбы, не имеет смысла представлять в виде полигонов даже в крупном масштабе).

Объекты, внемасштабные по ширине, но имеющие протяженность, отображаются в виде **линий** (1-мерный пространственный объект) (рисунок 2). Это могут быть дороги, линии электропередач, реки, тектонические разломы и другие. В ГИС они представлены последовательностью пар координат. Сложные наборы линий называются **сетями**. Они содержат дополнительную информацию о пространственных взаимоотношениях этих линий. Например, линии, отражающие транспортные магистрали, могут содержать сведения об особенностях передвижения по ним (направления, скорость и т. д.), при этом информация присваивается каждому отрезку до изменения атрибутов. Точки, в которых отрезки связываются, называются **узлами**, также кодируются, каждый узел свидетельствует о смене ситуации. Все эти атрибуты определяются по всей сети без пропусков, что необходимо для обеспечения связанности и пространственных отношений. Совокупность линий образует **линейный слой**.

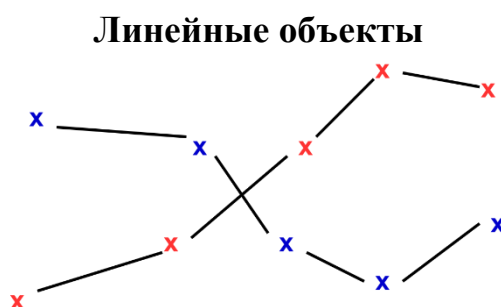


Рисунок 2: Полилиния – это последовательность связанных вершин. Каждая вершина имеет координаты X , Y и (возможно) Z . Атрибуты описывают каждую полилинию.

Если точечный объект состоит из одной вершины, то **полилиния** имеет две и более вершины. Полилиния – это непрерывная линия, соединяющая последовательность вершин, как показано на Рисунке 2.

Когда соединяются две вершины, создается линия. Когда к ним добавляются последующие вершины, получается «линия из линий», то есть **полилиния**. Полилинии используются для хранения геометрии линейных объектов, таких как дороги, реки, изолинии, маршруты и др. Иногда в дополнение к основной геометрии для полилиний устанавливаются специальные правила. Например, горизонтали высот могут касаться друг друга (в случае отвесного склона), но никогда не должны пересекаться, а полилинии, используемые для хранения данных о дорожной сети, должны быть связаны в местах перекрестков. В некоторых ГИС-приложениях Вы можете устанавливать набор подобных правил для определенных типов объектов (т.е. дорог), и программа будет проверять полилинии на соответствие этим правилам.

Полигоны используются для тех объектов, которые на карте сохраняют свои очертания (имеют длину и ширину) (2-мерный пространственный объект) (рисунок 3). Отображаются в виде замкнутой последовательности пар координат, где начальная и конечная точки имеют одинаковое значение. Совокупность полигонов образует **полигональный слой**.

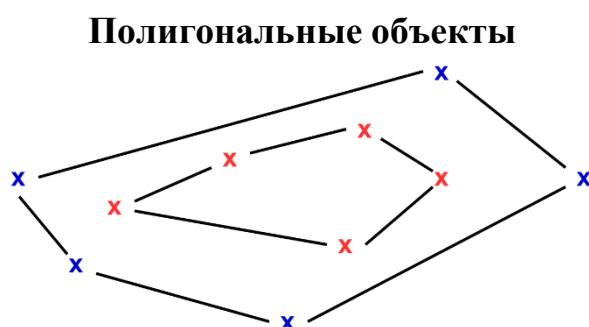


Рисунок 3: Полигон, как и полилиния, является последовательностью вершин. При этом, первая и последняя вершины всегда совпадают.

Поверхности выражены обычно высотами точек либо контурами рельефа. Модель поверхности обычно образована через

последовательность точек, распределенных с различной регулярностью, причем каждая точка отражает определенное значение высоты. Через три ближайших точки можно провести плоскость в виде треугольника, внутри которого будет зафиксирован постоянный уклон. Совокупность таких треугольников будет представлять модель поверхности, напоминающую кристалл. Создаваемая таким образом модель получила название TIN - нерегулярная триангуляционная сеть (Triangulated Irregular Network) (рисунок 4).

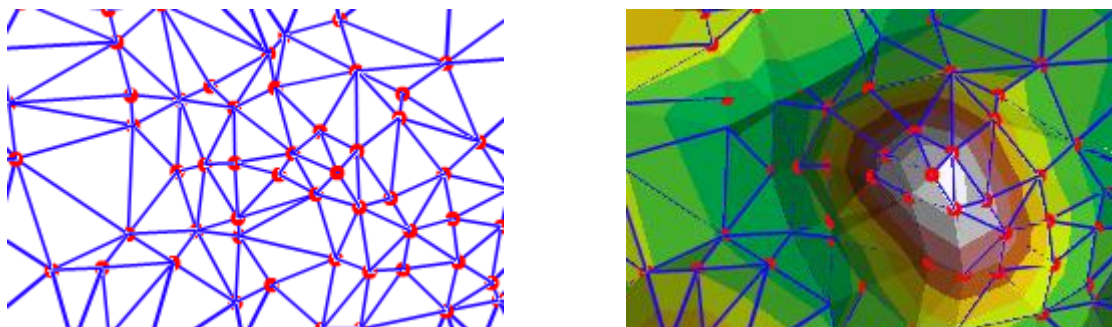


Рисунок 4: Модель данных TIN

Современные ГИС поддерживают одновременно множество моделей, однако векторные получили большую популярность, поскольку обеспечивают хорошую визуализацию картографируемых объектов, возможность детально представлять реальные объекты и более оперативную обработку информации в процессе анализа и геообработки.

Что мы можем делать с векторными данными в ГИС?

На самом простом уровне мы можем использовать векторные данные в ГИС-приложении так же, как мы используем обычные топографические карты. Реальные возможности ГИС начинают проявляться, когда вы начинаете задавать вопросы вроде «какие дома находятся в 100-летней зоне затопления близлежащей реки?», «где лучше разместить больницу, чтобы она была легко доступна как можно большему количеству людей?»,

«какие учащиеся проживают в определенном пригороде?» и т.д. ГИС является отличным инструментом для ответа на подобные вопросы с помощью векторных данных. Процесс ответа на такие вопросы принято называть **пространственным анализом**.

Задание.

1. Используя лист топографической карты, изображенной на Рисунке 5, определите различные типы векторных данных и выделите их на карте.

2. Подумайте, как Вы создали бы векторные объекты в ГИС для представления реальных объектов вокруг Вашего учебного заведения или дома. Создайте таблицу объектов и попросите учеников решить, какой тип геометрии лучше всего подойдет каждому объекту – точка, линия или полигон. Пример таблицы приведен ниже (Таблица 1).

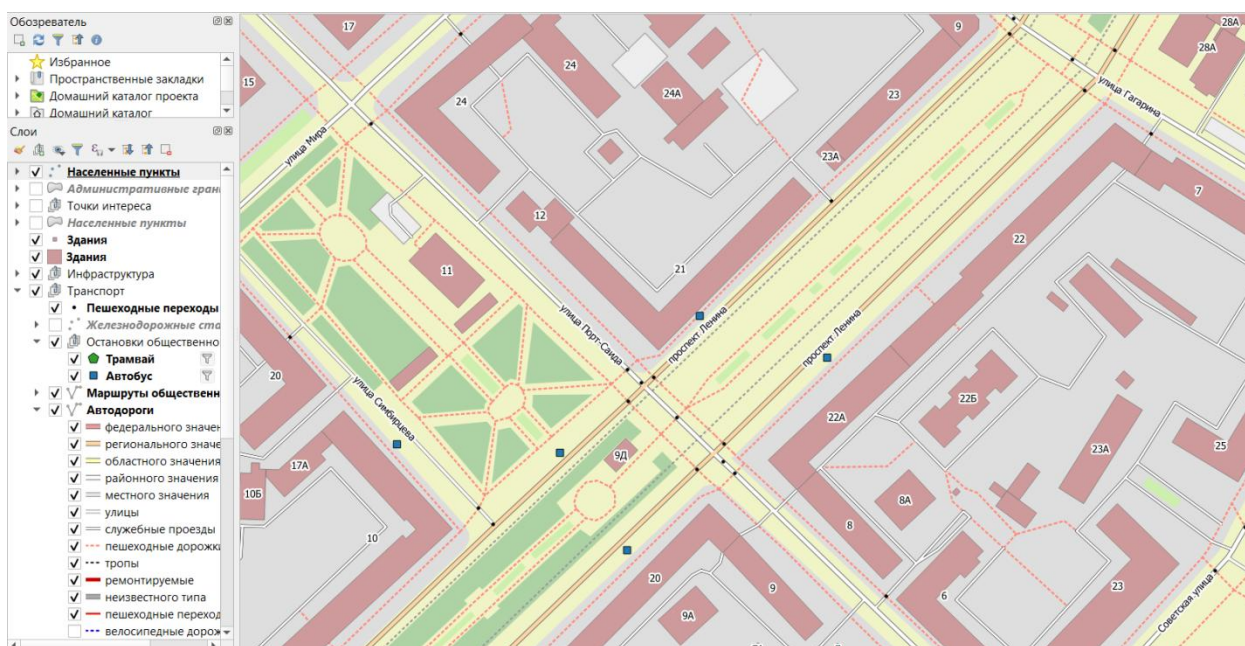


Рисунок 5: Определите различные типы векторных данных на этой карте

Объект реального мира	Подходящий тип геометрии
Футбольное поле	

Питьевые фонтаны	
Тропинки на территории	

Таблица 1: Создайте подобную таблицу и заполните её

2.2 Лабораторная работа № 2. Отображение пространственных данных

2.2.1 Цель работы

Получить навыки работы с двумерной визуализацией данных с использованием средств ГИС.

2.2.2 Порядок выполнения работы

- 1) Изучить теоретический материал по теме.
- 2) Установить QGIS Desktop версии 3.2 и выше.
- 3) Выполнить практические задания, указанные в п. 4 лабораторной работы.
- 4) Оформить отчет.

2.2.3 Теоретический материал

Данные в ГИС описывают, как правило, реальные объекты, такие как автомагистрали, различные виды строений, массивы растительности и многое другое. Реальные объекты можно разделить на две абстрактные категории: дискретные (дома, территориальные зоны) и непрерывные (рельеф, уровень осадков, среднегодовая температура). В зависимости от типа категории используют векторные и/или растровые данные.

Картографическая визуализация — это способ наглядно отобразить большое количество геоинформации и поместить имеющиеся данные в определенный географический контекст. Многочисленные социальные исследования, связанные с городским пространством, часто прибегают к

картографированию данных. Однако существующие современные цифровые технологии позволяют значительно расширить диапазон объектов, с которыми можно работать и при этом добавить интерактивности во взаимодействие с ними, и создать новые способы взаимодействия благодаря совмещению разных слоев данных.

2.2.3.1 ДОБАВЛЕНИЕ ТАЙЛОВЫХ КАРТ В QGIS

QGIS позволяет использовать сторонние слои карт, полученных из множества картографических сервисов. Мы рассмотрим, как осуществляется импортирование тайловых карт на примере сервиса Mapbox (<https://www.mapbox.com/>).

Для этого необходимо сформировать ссылку следующего вида: https://api.mapbox.com/styles/v1/YOUR_USERNAME/YOUR_STYLE_ID/wmts?access_token=<YOUR_ACCESS_TOKEN>.

Для этого нужно проделать следующие действия:

1. В качестве YOUR_USERNAME подставить название вашего аккаунта в Mapbox. Его можно узнать в Mapbox Studio (<https://studio.mapbox.com/>).
2. В качестве YOUR_STYLE_ID необходимо создать новый стиль в сервисе Mapbox Studio (<https://studio.mapbox.com/>), после чего подставить сформированный ID.
3. В качестве YOUR_ACCESS_TOKEN подставить ваш персональный токен, который доступен в профиле пользователя.

Полученная ссылка позволит загрузить определенную подложку карты.

Для добавления карты в QGIS Desktop необходимо добавить слой через меню Layer → AddLayer → Add WMS/WMTS layer... Далее

необходимо убедиться, что вы находитесь на вкладке «Layers» и создать новый слой, нажав на кнопку «New». После чего необходимо заполнить поле названия слоя и указать сформированную выше ссылку (рис 3.2.1). После чего нужно нажать на кнопку «OK».

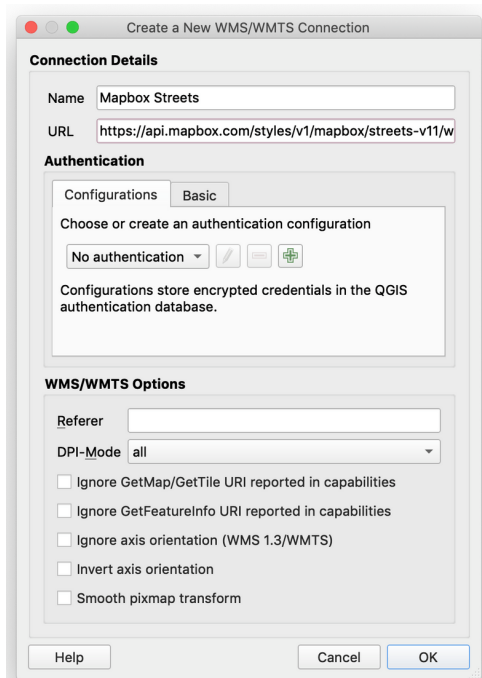


Рисунок 3.2.1 – Окно создания нового слоя WMS/WMTS Connection

Далее необходимо выбрать ваш слой Марбох и нажать на кнопку «Connect» в соответствии с рис. 3.2.2.

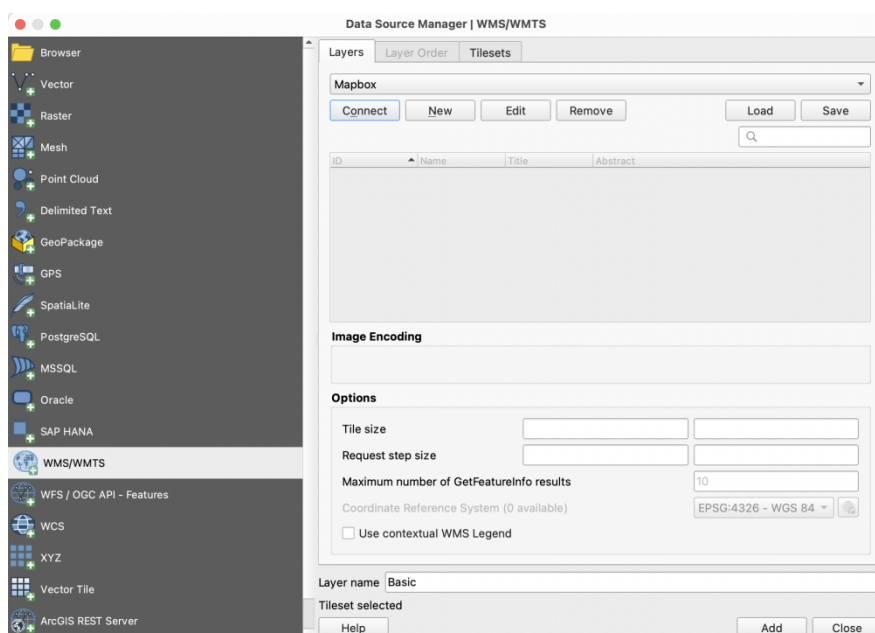


Рисунок 3.2.2 – Окно интерфейса WMS/WMTS Layers

Далее необходимо выбрать созданный слой, нажать на кнопку «Add», в соответствии с рис. 3.2.3, после чего закрыть диалоговое окно.

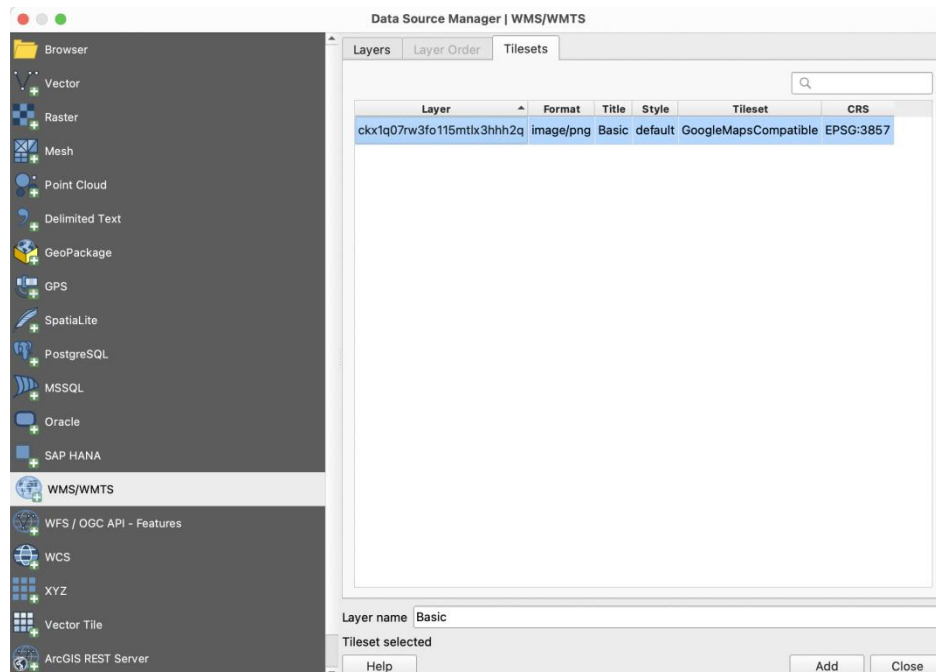


Рисунок 3.2.3 – Окно интерфейса WMS/WMTS Tilesets

2.2.3.2 ПОДГОТОВКА ДАННЫХ С ПОМОЩЬЮ СЕРВИСА ВВВІКЕ

Сервис ВВВіке позволяет извлекать области из Planet.osm в таких форматах, как: OSM, PBF, ESRI Shapefile и так далее.

Для выгрузки данных с использованием сервиса ВВВіке (<https://extract.bbbike.org>) (рис. 3.2.4) необходимо проделать следующую последовательность шагов:

- 1) переместить карту к нужной локации;
- 2) создать полигон, захватывающий выбранную область на карте;
- 3) в качестве формата файла для экспорта выбрать Esri Shapefile;

4) экспортировать данные кнопкой «Extract»;

5) скачать готовый архив с данными на ПК.

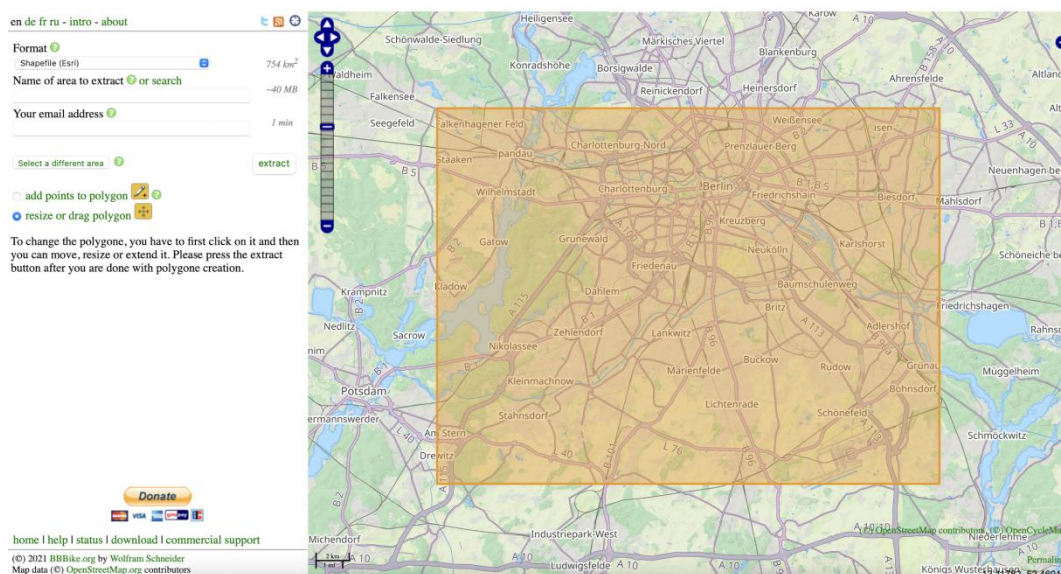


Рисунок 3.2.4 – Интерфейс сервиса BVBike

2.2.3.3 ГЕКСАГОНАЛЬНАЯ СЕТКА

Добавить слои из полученного zip-архива с локацией в QGIS Desktop, можно с помощью меню Layer → AddLayer → Add Vector Layer... Необходимо выбрать о точках интереса (файл points.shp). Из данного слоя нужно выбрать только те данные, которые относятся к кафе (type = 'cafe'). Для этого нужно кликнуть правой кнопкой мыши по слою и выбрать пункт меню «Filter...». Для построения сеток понадобится плагин MMQGIS, в случае отсутствия нужно установить его через менеджер плагинов QGIS. Далее необходимо создать слой сетки Hexagon, используя меню MMQGIS → Create → Create Grid Layer. Убедитесь, что проекции сетки и исходного слоя совпадают, в противном случае необходимо отредактировать и создать слой заново.

Необходимо подсчитать, какое количество точек попадает в полигон. Для этого нужно воспользоваться инструментом анализа векторных

данных: Processing Toolbox → Vector Analysis Tool → Count points in polygon.

Далее необходимо произвести фильтрацию полигонов, которые не содержат точек, с помощью инструмента Select features, с условием $NUMPOINTS > 0$. После чего нужно настроить цвет и прозрачность полигонов на карте, в соответствии (рис. 3.2.5).

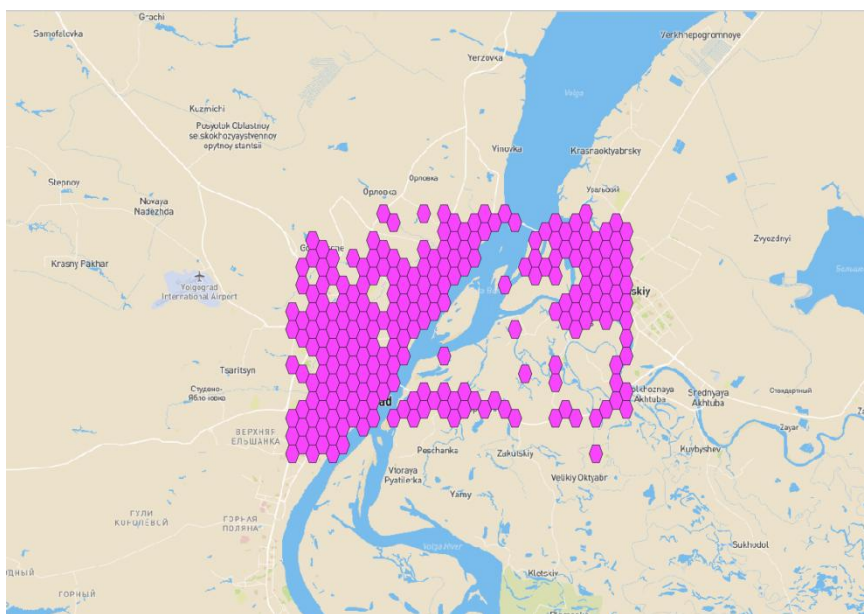


Рисунок 3.2.5 – Гексагональная сетка

2.2.3.4 ТЕПЛОВЫЕ КАРТЫ

Тепловая карта (англ. heatmap) – это графическое представление данных, где значения интересующей переменной выделяются цветом.

Одной из задач использования тепловых карт – это изучение плотности застройки города. Из полученного ранее zip-архива с локацией необходимо открыть слой buildings (файл buildings.shp). Далее нужно извлечь центроиды из каждого полигона с помощью команды Processing Toolbox → Polygon centroids. После этого открыть панель «Layer Styling», в соответствии с рис. 3.2.6.

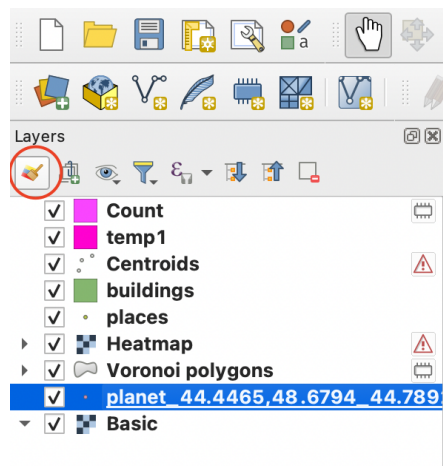


Рисунок 3.2.6 – Панель слоев

Далее в выпадающем меню нужно выбрать опцию «Heatmap», как показано на рис. 3.2.7.

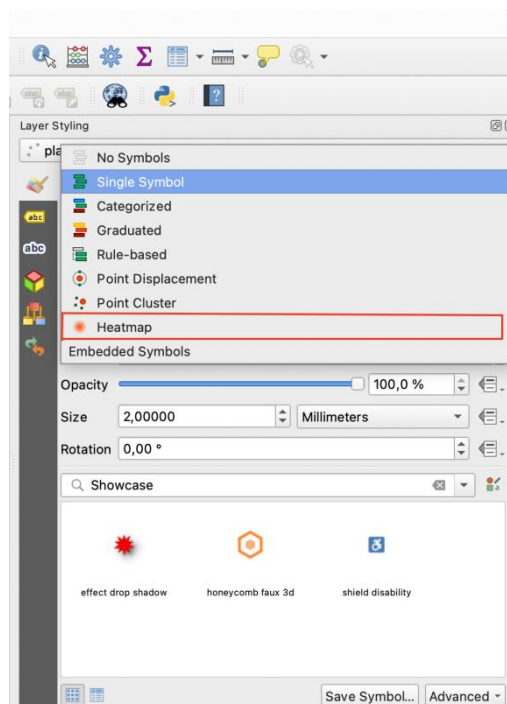


Рисунок 3.2.7 – Панель «Layer Styling»

После чего необходимо установить цвет и прозрачность слоя тепловой карты. В результате получится тепловая карта, показанная на рис 3.2.8.

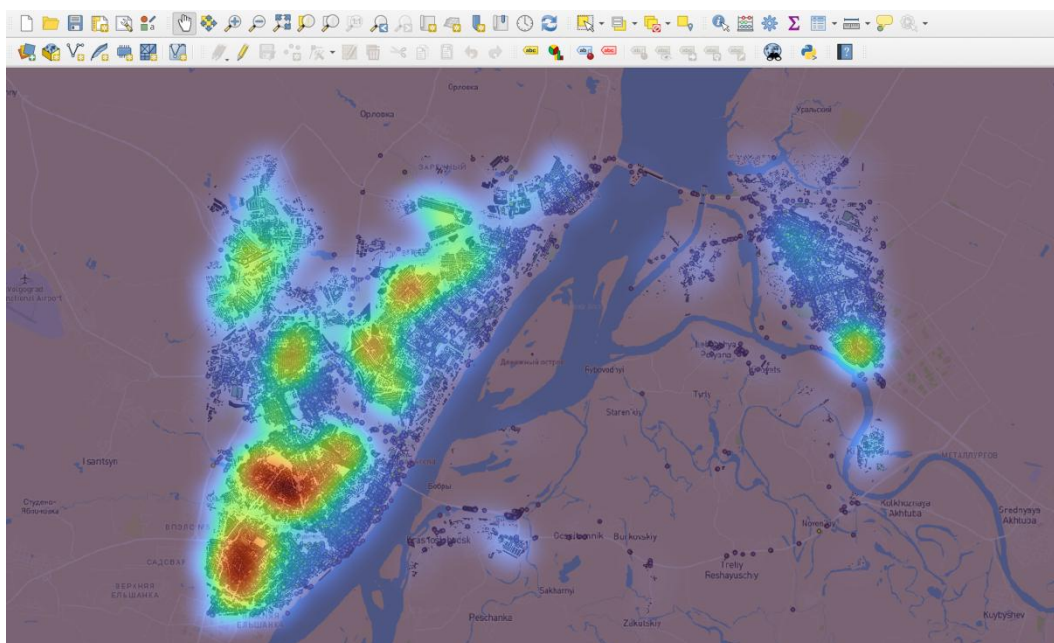


Рисунок 3.2.8 – Тепловая карта

2.2.3.5 ДИАГРАММА ВОРОНОГО

Диаграмма Вороного (названа в честь российского ученого Георгия Феоодосьевича Вороного) представляет собой компактную структуру данных, хранящую всю необходимо информацию для решения множества задач о близости. В результате разбиения пространства диаграммой Вороного по данным будет визуализирован набор областей, каждая из которых содержит ровно одну точку.

В качестве примера, изучим плотность остановок общественного транспорта в Волгограде. Для этого необходимо выбрать значения (`type = 'bus_stop'` OR `type = 'tram_stop'`) из слоя точек интереса. Для создания диаграммы в основном меню нужно выбрать пункт `Vector → Geometry Tools → Voronoi polygons` (либо `MMQGIS → Create → Voronoi diagram`). В итоге получится диаграмма, как на рис. 3.2.9.

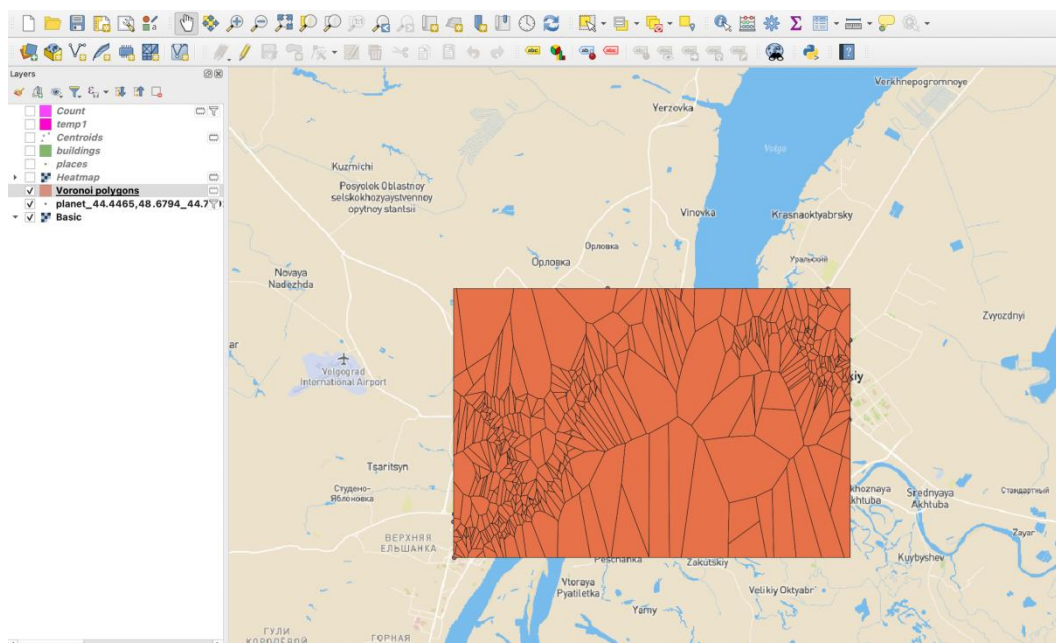


Рисунок 3.2.9 – Диаграмма Вороного

Получившееся изображение плохо отображает ситуацию на карте. Один из способов сделать изображение более информативным — это окрасить полигоны в зависимости от их площади. Для этого необходимо открыть панель «Layer styling», выбрать тип стилизации «Graduated», далее в поле «Value» установить значение «\$area» и подобрать подходящую цветовую схему. В итоге получится диаграмма, как на рис. 3.2.10.

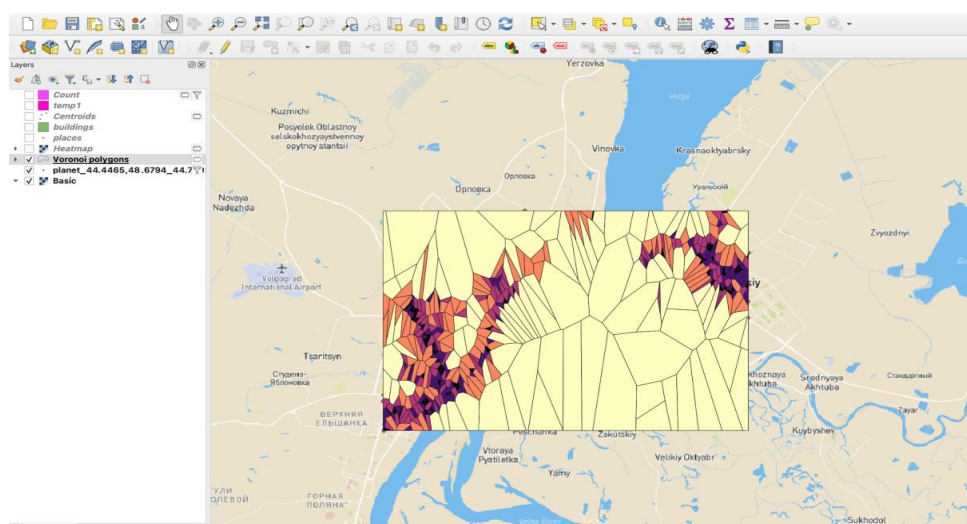


Рисунок 3.2.10 – Диаграмма Вороного с настроенными стилями

2.2.3.6 ВЫВОД ИЗОБРАЖЕНИЙ НА ПЕЧАТЬ

QGIS Desktop имеет функционал подготовки карты для вывода на печать. В качестве исходных данных будем использоваться точки интереса из ранее выбранной локации.

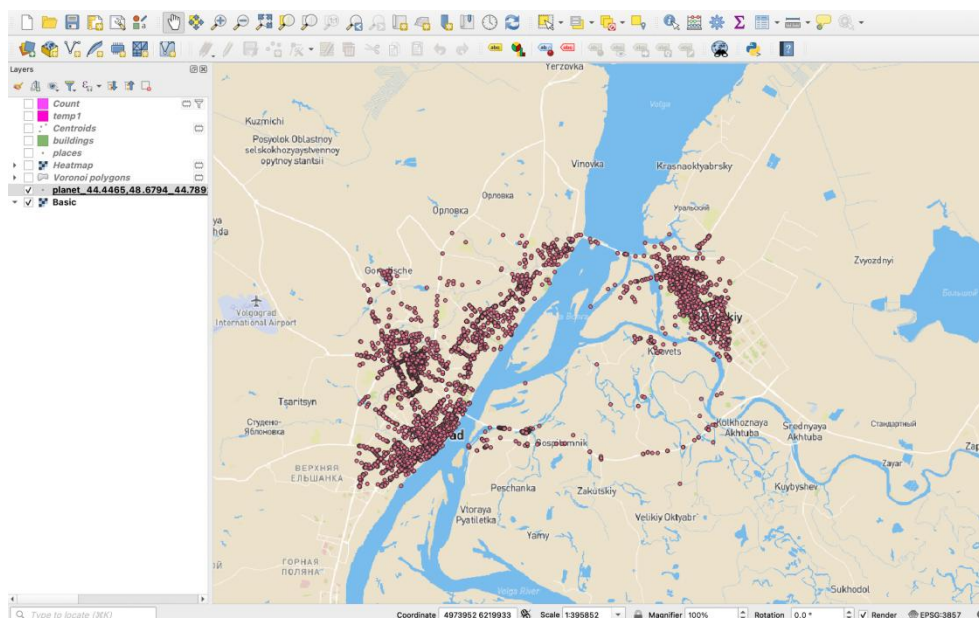


Рисунок 3.2.11 – Данные о точках интереса г. Волгограда

Далее необходимо в основном меню перейти в Project → New Print Layout. Интерфейс окна подготовки печати изображения представлен на рис. 3.2.12.

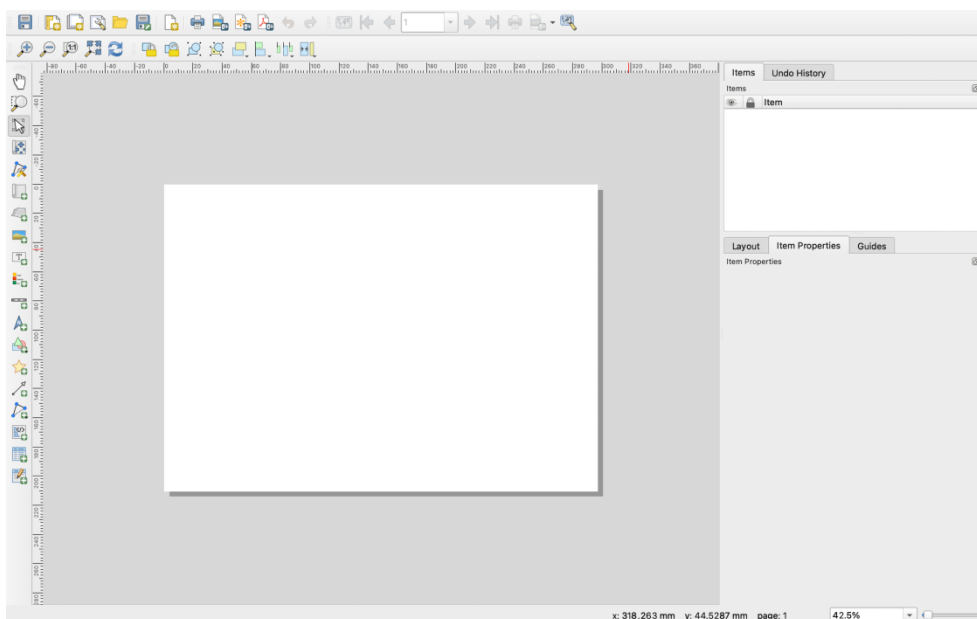


Рисунок 3.2.12 – Интерфейс окна Print Layout

Используя кнопку «Add Map» на панели инструментов слева, необходимо нарисовать прямоугольник в произвольной области листа, где вы желаете разместить карту. Однако следует учесть, что карта должна содержать заголовок и легенду, тем самым необходимо заранее выделить для этого место на листе.

Обычно карта отображает область идентичную той, что отображена в QGIS Desktop. Если необходимо изменить отображаемую область на карте или изменить ее масштаб, то нужно перейти в QGIS и выполнить необходимые действия. После чего вернуться в Print Layout и воспользоваться кнопкой «Refresh».

Для добавления заголовка необходимо воспользоваться инструментом «Add label». Чтобы задать свойства заголовка, нужно выбрать текстовый слой во вкладке Items. Также в окне свойств (рис. 3.2.13) можно настроить параметры шрифта и многое другое.

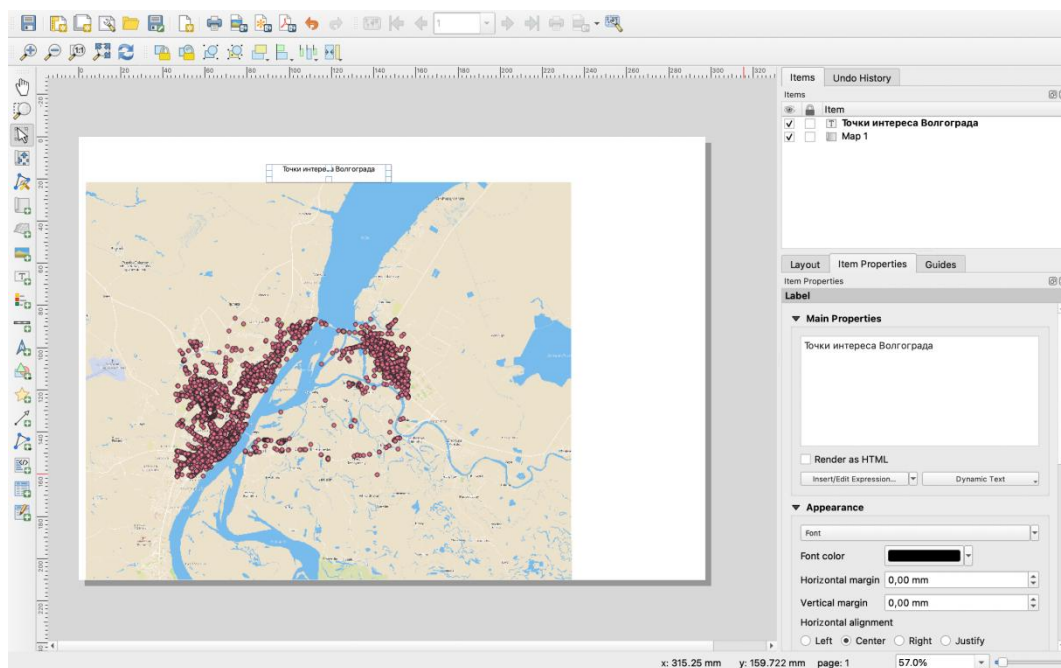


Рисунок 3.2.13 – Добавление заголовка

Для добавления легенды карты используется инструмент «Add legend». Необходимо поместить легенду в правую часть карты в свободное для этого место. Настройка параметров отображения легенды осуществляется через окно свойств. Для возможности редактирования элементов легенды, необходимо отключить автообновление кнопкой «Auto update». Итоговая карта представлена на рис. 3.2.14 ниже.

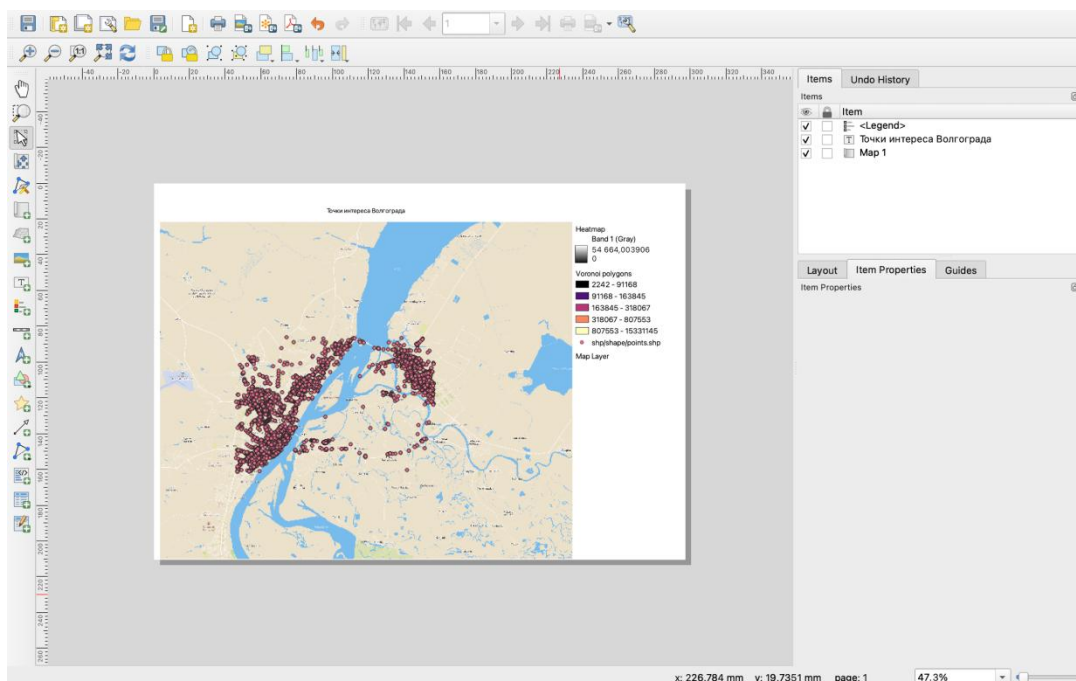


Рисунок 3.2.14 – Карта с легендой

По завершении работы необходимо сохранить получившуюся карту, для этого нужно воспользоваться кнопкой экспорта «Export as image». Обязательно необходимо задать разрешение не ниже 350 DPI, так мы сохраним все детали при печати.

3.2.4 Практические задания

- 1) Изучить теоретический материал по теме.
- 2) Определить территорию на карте для анализа.
- 3) Построить гексагональную сетку для точек интереса выбранного района.
- 4) Построить тепловую карту плотности застройки вашего района.

- 5) Построить диаграмму Вороного для плотности общественного транспорта в выбранном районе.
- 6) Сохранить полученную карту в файле.
- 7) Оформить отчет.

3.2.5 Рекомендуемая литература

1. *Берлянт А. М.* Теория геоизображений. М.: Геос, 2006. С. 181–187.
2. ДеМерс Майкл Н. Географические информационные системы. Основы.: Пер. с англ. – М.: Дата+, 1999. – 490 с.
3. Геоинформатика: Учеб. для студ. вузов / *Е. Г. Капралов, Г35 А. В. Кошкарев, В. С. Тикунов и др.*; Под ред. *В. С. Тикунова*. М.: Издательский центр «Академия», 2005. С. 199–230.

3.2.6 Вопросы к отчету

1. Назовите существующие способы представления географически привязанной информации?
2. Какие существуют особенности картографической визуализации?
3. Назовите существующие способы районирования пространства и их особенности?
4. Какие данные можно описать с помощью ГИС?

2.3 Лабораторная работа № 3. Создание карты субъекта Российской Федерации на основе данных OpenStreetMap

OpenStreetMap

OpenStreetMap (www.openstreetmap.org) – это картографический сервер, на котором карта создается пользователями (как в Википедии). Карта охватывает весь мир. Есть регионы, где карта очень подробна, есть регионы, где она практически отсутствует. Любой пользователь, в том числе и вы, может принять участие в создании этой карты.

Карту *OpenStreetMap* можно использовать и для создания своих карт на определенных лицензионных условиях. На сайте NextGIS из данных OpenStreetMap и других открытых источников создаются обновляемые наборы слоев по любой точке мира. Данные наборы слоев доступны в форматах ESRI Shapefile, ESRI Geodatabase, Mapinfo TAB, GeoJSON, PBF, OSM (XML), SQL (PostgreSQL), CSV и PDF, что позволяет использовать их практически в любой ГИС. Например, можно заказать выгрузку данных OpenStreetMap по Волгоградской области (data.nextgis.com/ru/region/RU-VGG/).

В данной лабораторной работе мы рассмотрим, как выгрузить данные *OpenStreetMap* самостоятельно, используя плагин *QuickOSM* (https://docs.qgis.org/3.16/en/docs/training_manual/qgis_plugins/plugin_examples.html), который следует установить из официального репозитория плагинов QGIS.

Напомним, что для работы с модулями на языке Python (а это подавляющее большинство модулей) служит специальный модуль - «Установщик модулей», предназначенный для загрузки дополнительных расширений из официального и/или авторского репозитория. «Управление модулями» тоже является модулем ядра QGIS, поэтому устанавливать и

включать его не нужно. Вызвать его можно сразу после установки и запуска QGIS (рисунок 1).

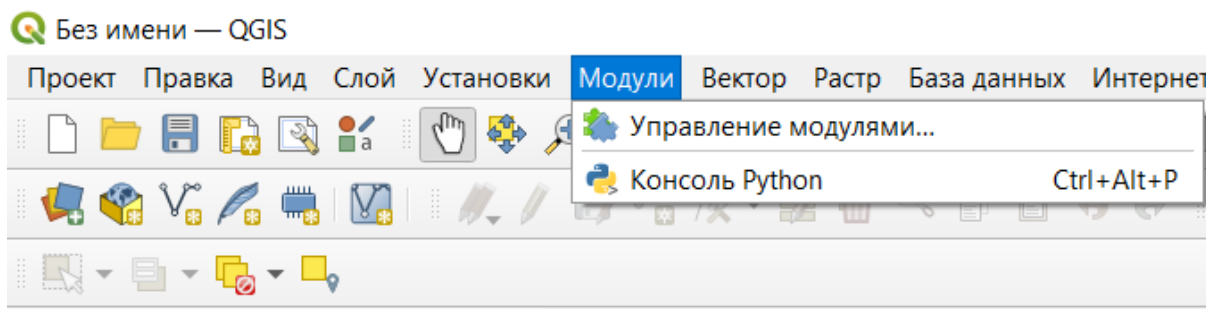


Рисунок 1 – Вызов модуля Управление модулями в меню Модули

Для установки модуля переключимся на закладку «Не установленные» и, либо отсортировав по имени, либо введя часть названия модуля в строку поиска найдем интересующий нас модуль.

После того как он найден - выберем его и нажмем «Установить модуль» (рисунок 2).

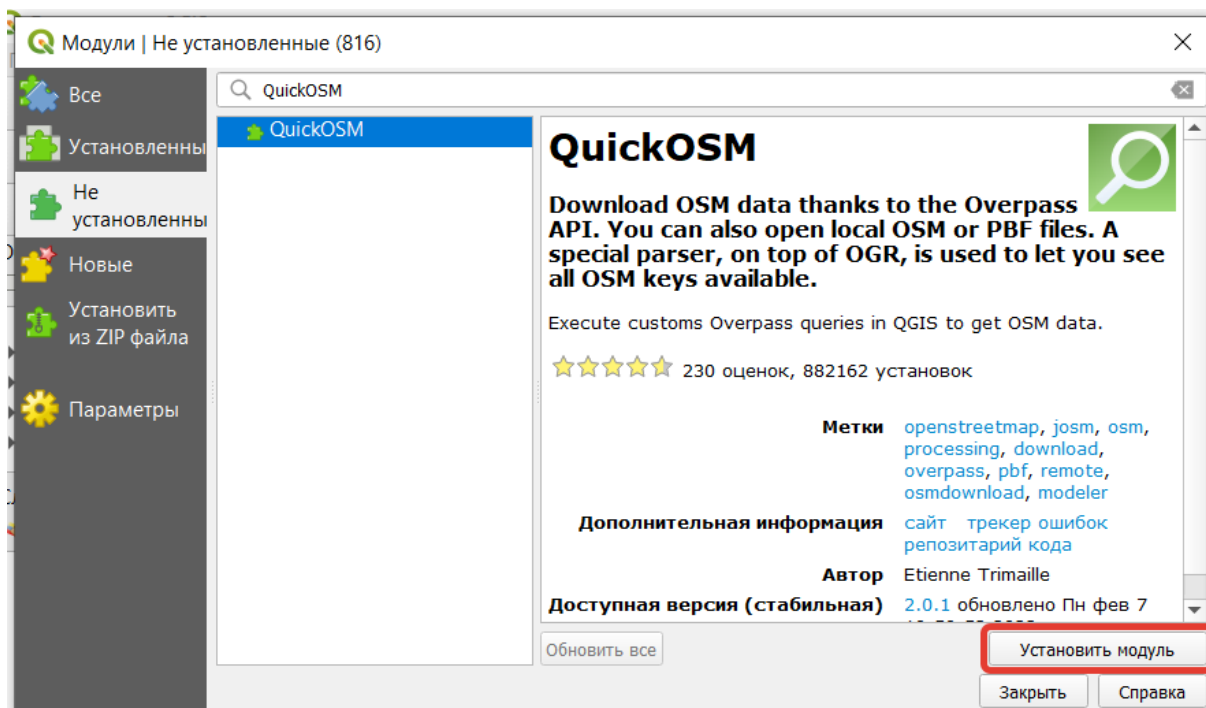


Рисунок 2 – Установка модуля QuickOSM

После загрузки модуля он будет автоматически активирован.

После того, как модуль *QuickOSM* установлен, в меню Вектор появится соответствующий пункт *QuickOSM* (рисунок 3).

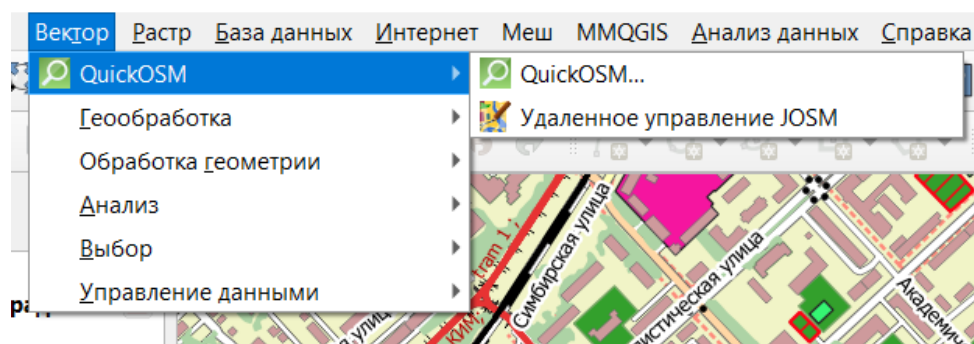


Рисунок 3 - QuickOSM в меню Вектор

Выгрузка данных OpenStreetMap с использованием плагина QuickOSM.

Создадим пустой проект и дадим ему имя *VolgogradRegion*.

В качестве подложки установим карту *Google Road*, используя модуль *QuickMapServices* и отмасштабируем её до Волгоградской области.

Далее загрузим данные о районах Волгоградской области и их границах.

Семантика OSM состоит из набора свойств (называемых «тегами»), описывающих географические классы, использование которых определяется участниками проекта на специальном веб-сайте Wiki (https://wiki.openstreetmap.org/wiki/RU:Объекты_карты). Модель данных проста и состоит из узлов, путей и отношений. Каждый сопоставленный объект сопровождается тегом, который представляет из себя соединение ключа и его значения.

Для загрузки данных о районах Волгоградской области и их границах воспользуемся тегом с парой ключ-значение *admin_level* и *6*, указав в поле «В» *Волгоградская область*. Укажем, что результат нужно сохранить в рабочем каталоге, как ESRI Shapefile (рисунок 4).

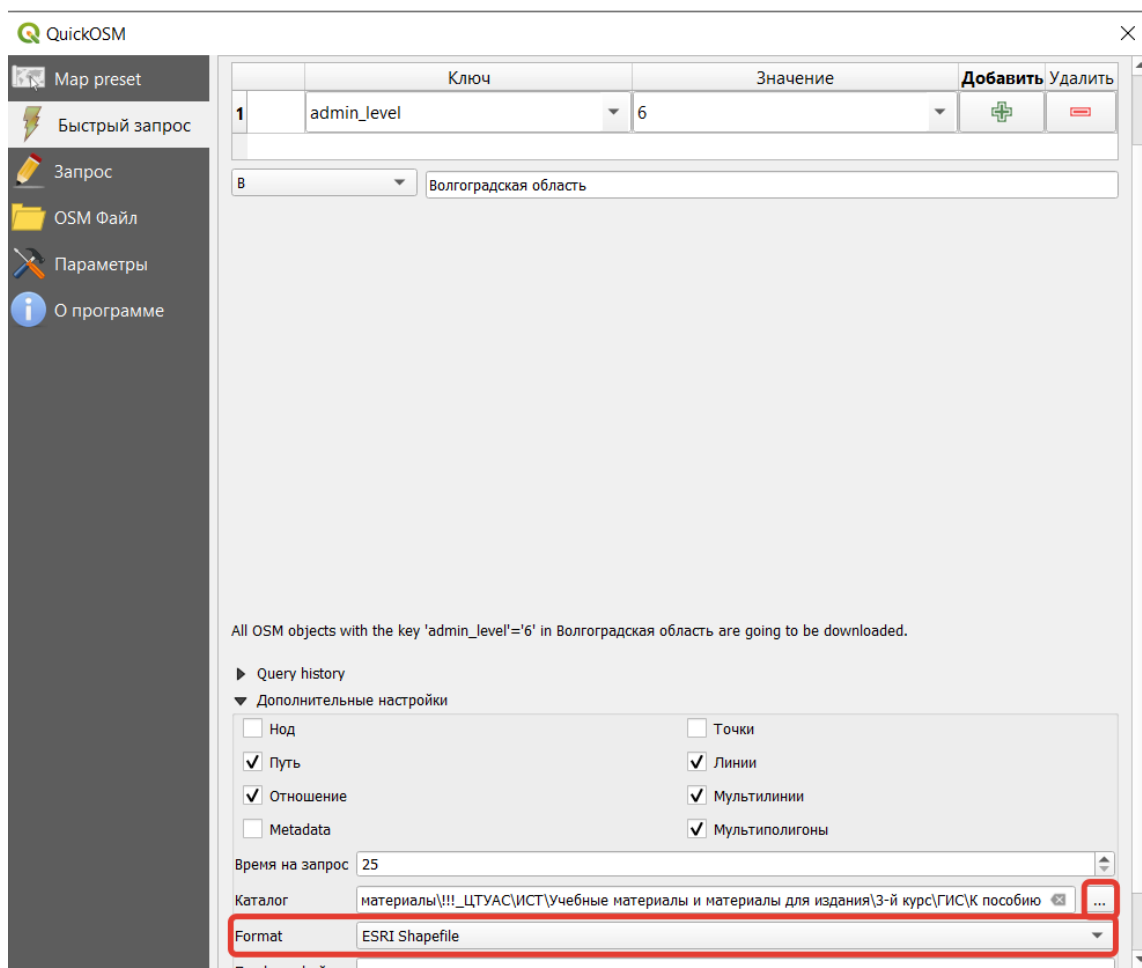


Рисунок 4 - Загрузка данных о районах Волгоградской области и их границах

Выполнив запрос, мы получим два векторных слоя: полигональный с территориями районов и линейный с их границами.

Для корректного отображения кириллических символов в атрибутах элементов, нужно в свойствах слоев изменить значение кодировки с *System* на *UTF-8* (рисунок 5).

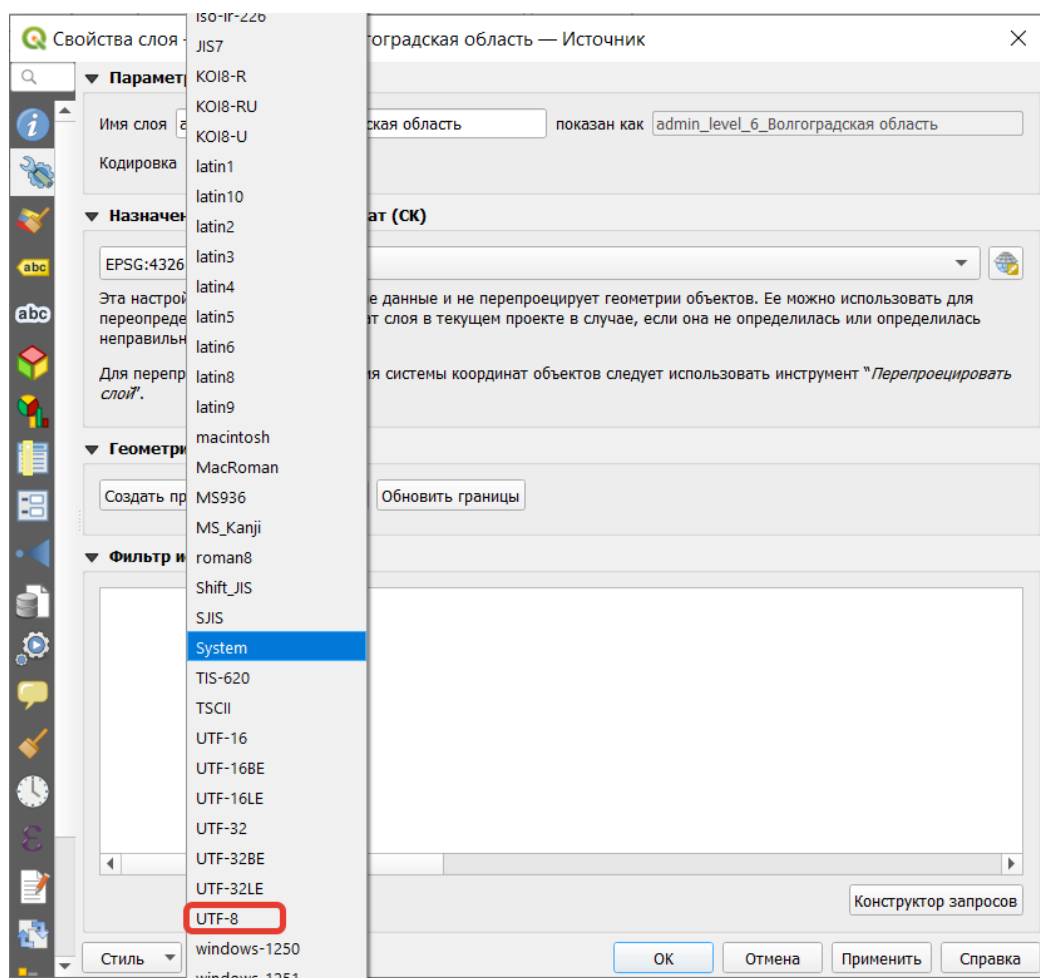


Рисунок 5 - Изменение значения кодировки свойстве слоя

Задание:

1. Для полигонального слоя, содержащего данные о районах Волгоградской области, укажите тип легенды *Уникальные значения* – поле классификации "*name*" и сделайте скриншот результата.
2. По ключу *place* со значениями *city* и *town*, указав в дополнительных настройках только *Нод* и *Точки*, получите точечный слой с основными населенными пунктами.

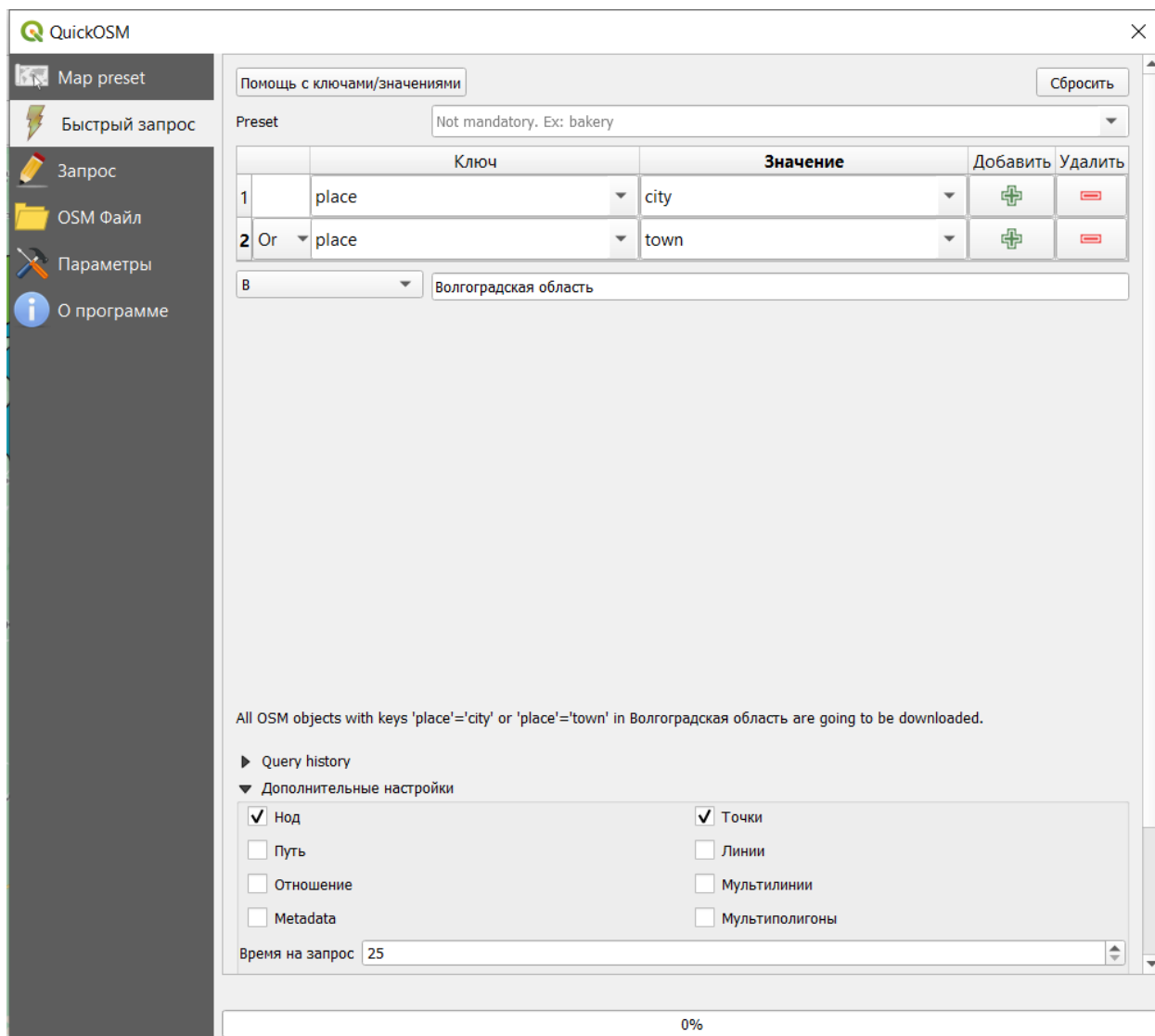


Рисунок 6 – Параметры запроса для получения точечного слоя с основными населенными пунктами

3. Для полученного слоя:

- Задайте тип легенды **Уникальные значения** – поле классификации "**place**".

- Для значения "**city**" выберите в библиотеке символов символ **diamond red**. Укажите размер 4,5 мм.

- В легенде укажите "**Крупные города**".

- Для значения "**town**" укажите размер маркера 3 мм, и задайте для него синий цвет.

- В легенде укажите "**Крупные населенные пункты**".
- Задайте подписи для слоя по полю надписи "**name**", шрифт "**Arial Rounded MT Bold**", размер 8.
- Установите буфер размером 1 мм.
- Сделайте скриншот результата.

2.4 Лабораторная работа № 4. Анализ транспортных сетей

Задание посвящено знакомству с сетевым анализом. Задачи, предлагаемые в задании, связаны с определением оптимальных маршрутов, построением зон обслуживания, определением ближайших сервисных точек, размещением сервисных точек. Данные задачи активно используются в логистике — оптимизации перевозок, а также в геомаркетинге и оптимизации местоположения пунктов обслуживания (магазинов, складов, пожарных депо и т.д.).

В основе решения этих задач лежит сетевая модель данных, являющаяся частным случаем векторной модели. В сетевой модели дорожная сеть представляется в виде графа.

Добавление исходных данных

Создадим пустой проект и дадим ему имя **Volgograd**.

Загрузим данные о районах Волгограда и их границах, воззвавшись для этого модулем **QuickOSM**.

Создадим быстрый запрос по тегу с парой ключ-значение **admin_level** и **9**, указав в поле «В» **Волгоград**. Укажем, что результат нужно сохранить в рабочем каталоге, как ESRI Shapefile (рисунок 1).

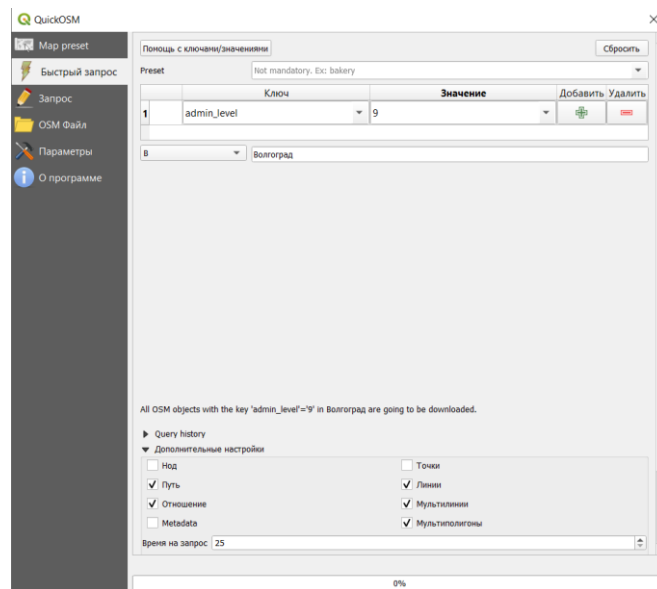


Рисунок 1 – Параметры запроса для получения данных о районах Волгограда и их границах

Для полигонального слоя, содержащего данные о районах Волгограда, укажем тип легенды *Уникальные значения* – поле классификации "*name*". (рисунок 2). Непрозрачность слоя зададим 35,8%.

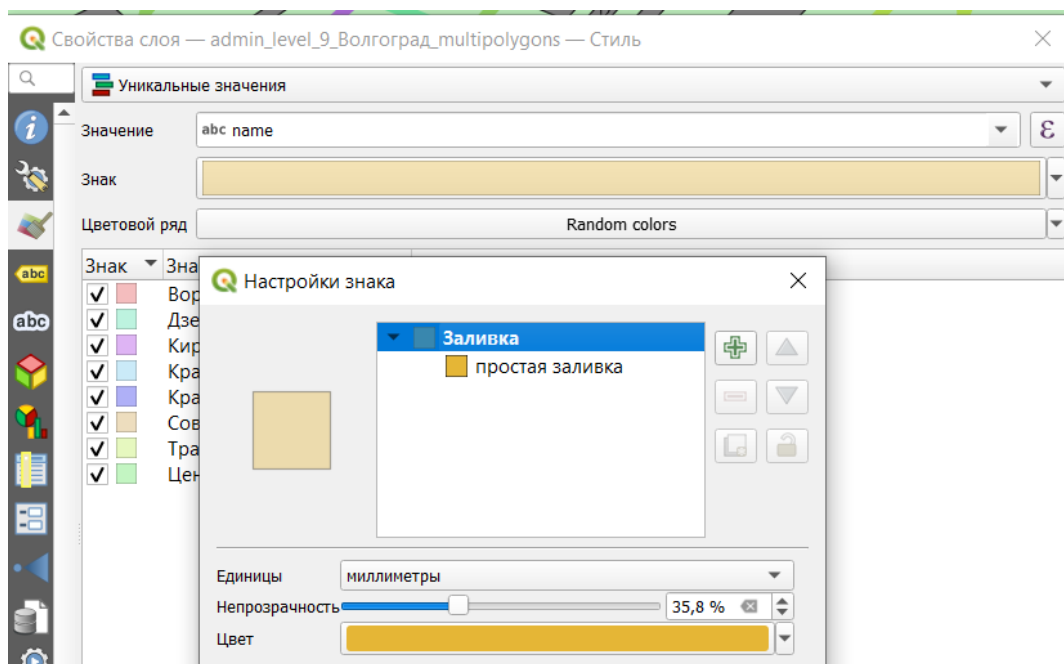


Рисунок 2 – Классификация данных о районах

Далее добавим в проект слои *RoadNetworkVolgograd* (улично-дорожная сеть) и *VolgogradBuildings* (здания и сооружения). Оба этих слоя подготовлены в формате **GeoPackage** на основе данных **OpenStreetMap**.

Зададим для слоя *Здания и сооружения* следующие настройки отображения:

- полигоны серого цвета (#8a8a8a) без обводки;
- подписи для слоя по полю "A_HSNMBR", размер 6.

Для слоя *Улично-дорожная сеть* — линии синего цвета толщиной 0,26 мм. С подписями для слоя по полю "NAME", размер 83

Фрагмент фрейма карты после увеличения масштаба изображения до 1:5000 показан на рисунке 3.



Рисунок 3 – Фрагмент фрейма карты после увеличения масштаба изображения до 1:5000

Выбор элементов дорожной сети

В рамках этого упражнения мы будем решать задачи логистики на примере движения автомобилей. Однако набор данных об улично-

дорожной сети, предоставленный вам в качестве исходных данных, содержит не только действующие автомобильные дороги, но и другие типы объектов. Тип объекта хранится в поле "HIGHWAY". В таблице 1 перечислены значения атрибута "HIGHWAY" для тех объектов, которые не должны участвовать в анализе.

Таблица 1. Значения атрибута "HIGHWAY" для объектов, которые не должны участвовать в анализе

Значения атрибута "HIGHWAY"	Пояснение
construction	строящиеся дороги
cycleway	велодорожки
footway, pedestrian	пешеходные дороги и тротуары
path	тропы
proposed	планируемые дороги
raceway	гоночные треки
steps	лестницы

Составим запрос, чтобы выбрать в слое *RoadNetworkVolgograd* все объекты, **кроме** перечисленных выше.

Для этого выполним следующую последовательность действий (рисунок 4):

1. Выберем слой *RoadNetworkVolgograd* на панели **Слои**.
2. Включим таблицу атрибутов в меню.
3. Откроем её во встроенной панели.
4. Включим редактор для построения запросов.
5. Построим запрос.
6. Выделим объекты, удовлетворяющие условиям этого запроса.

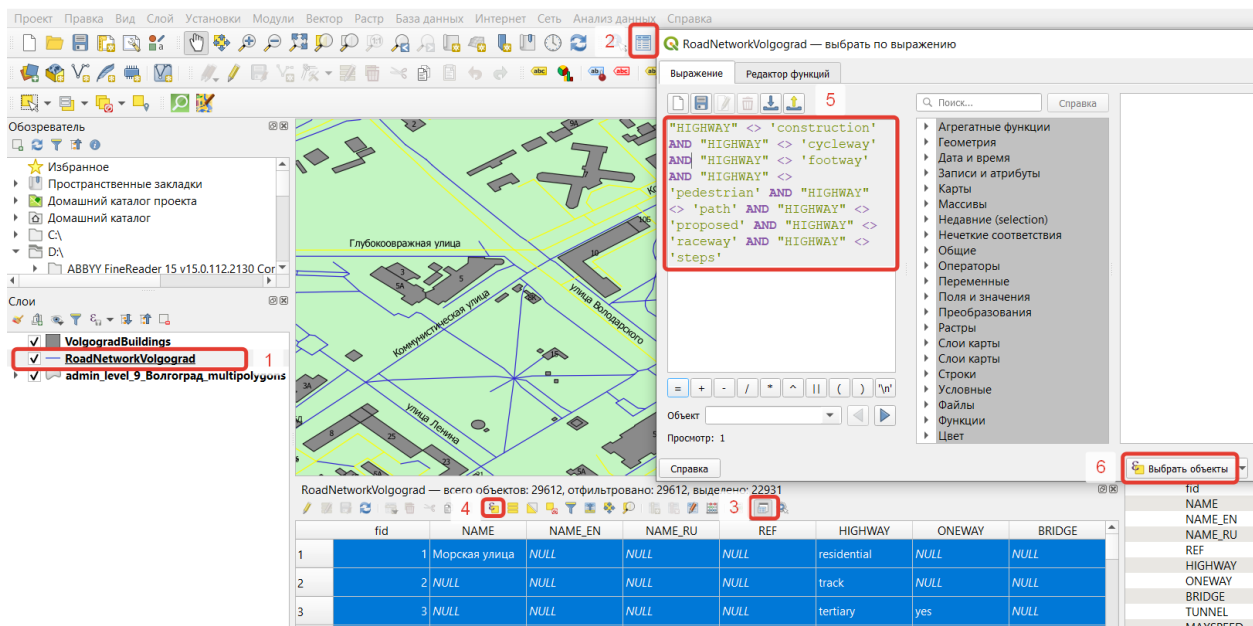



Рисунок 4 – Последовательность действий для создания запроса

Сохраним выборку в новый набор данных формата **GeoPackage**, как *FilteredRoadNetworkVolograd* (правой кнопкой мыши на слой — «Экспорт...» — «Сохранить выбранные объекты как...»).

Закроем дополнительные окна, и снимем выделение объектов, нажав на иконку  на панели инструментов.

Когда новый слой будет добавлен к проекту, изменив настройки его визуализации: установите толщину линий равной 0,26 мм, а цвет — любой, кроме того, который уже используется для улично-дорожной сети.

Фрагмент фрейма карты после сохранения выбранных дорог в отдельный набор показан на рисунке 5.



Рисунок 5 – Фрагмент фрейма карты после сохранения выбранных дорог в отдельный набор

Построение маршрута

Существует несколько подключаемых модулей для **QGIS**, которые позволяют решать задачи сетевого анализа. Мы воспользуемся одним из самых простых — модулем **QNEAT3**.

В отличие от «продвинутых» программных продуктов, большинство модулей для сетевого анализа в **QGIS** не требуют представления графа дорожной сети в виде отдельной сущности. Вместо этого такой граф создаётся непосредственно в процессе работы инструментов на основе векторного линейного набора объектов.

Установим модуль **QNEAT3**. Откроем панель инструментов анализа и убедимся, что в неё добавилась группа инструментов **QNEAT3** (рисунок 6).

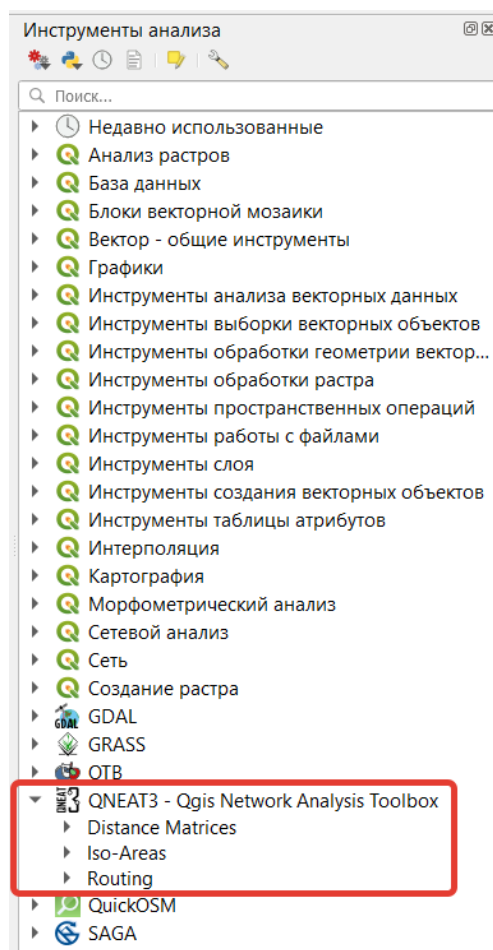


Рисунок 6 – Панель инструментов анализа с группой инструментов **QNEAT3**

Теперь построим маршрут между двумя точками. Для решения этой задачи используется инструмент **Shortest path (point to point)** из группы **Routing** набора **QNEAT3**

Найдем здание главного корпуса ВолгГТУ (проспект Ленина, 28) и здание главного корпуса ИАиС ВолгГТУ (улица Академическая Ленина, 1). Запомним расположение этих зданий, создав пространственные закладки.

Запустим инструмент **Shortest path (point to point)**. Откроется интерфейс настройки инструмента.

В качестве слоя дорожной сети (**Network Layer**) укажем набор данных, созданный на предыдущем шаге.

Чтобы указать начальную точку маршрута, нажмем многоточие справа от поля ввода **Start point**. Интерфейс настройки инструмента будет свёрнут, а курсор в окне **QGIS** примет вид мишени. Найдем здание главного корпуса ВолгГТУ и щёлкнем левой кнопкой мыши в произвольном месте внутри здания. Координаты курсора будут считаны и введены в поле **Start point**.

Аналогичным образом укажем конечную точку маршрута в пределах здания главного корпуса ИАиС ВолгГТУ.

Проверим, что в качестве критерия оптимизации указано **Shortest Path** (кратчайшее расстояние), а выходной набор данных сохраняется во временный файл (рисунок 7).

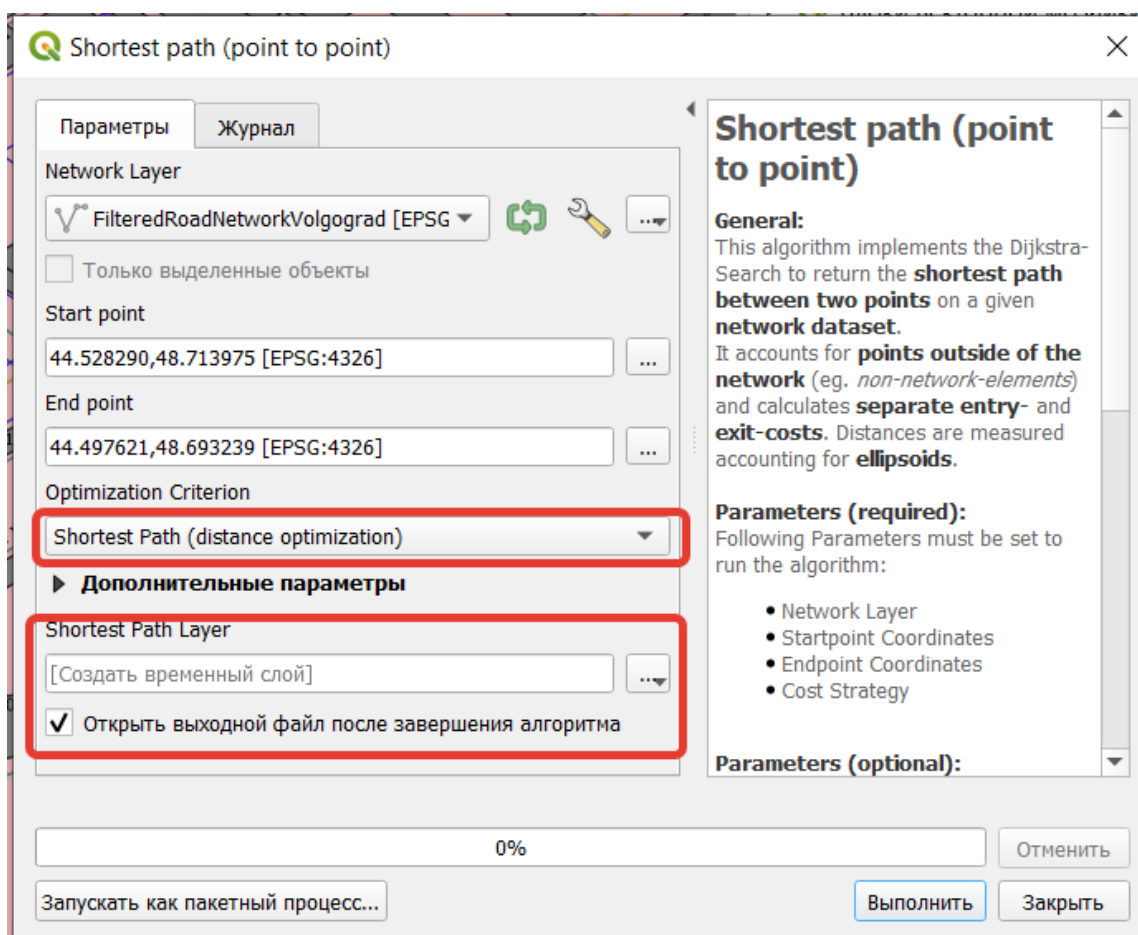


Рисунок 7 – Интерфейс настройки инструмента **Shortest path (point to point)**

Запустим инструмент и дождемся, пока будет рассчитан кратчайший маршрут. Интерфейс инструмента закрывать не будем, т.к. мы воспользуемся им снова, изменив параметры.

Построенный кратчайший маршрут появится в таблице слоёв как временный слой с названием *Shortest Path Layer*. Изменим стиль этого слоя на линию красного цвета толщиной 1 мм. Отобразим карту в охвате слоя *Shortest Path Layer* (рисунок 8).



Рисунок 8 – Карта в охвате слоя *Shortest Path Layer*

Обратите внимание, что при построении этого маршрута учитывались только длины сегментов пути, но не учитывались характерные скорости и разрешённые направления движения. На следующих шагах мы внесём соответствующие коррективы.

Откроем таблицу атрибутов слоя дорожной сети и изучим её содержимое.

Произведем реорганизацию этой таблицы (рисунок 9), оставив только те столбцы, которые могут быть использованы для дальнейшей работы (**fid**, **HIGHWAY**, **HIGHWAY**, **ONEWAY**, **MAXSPEED**).

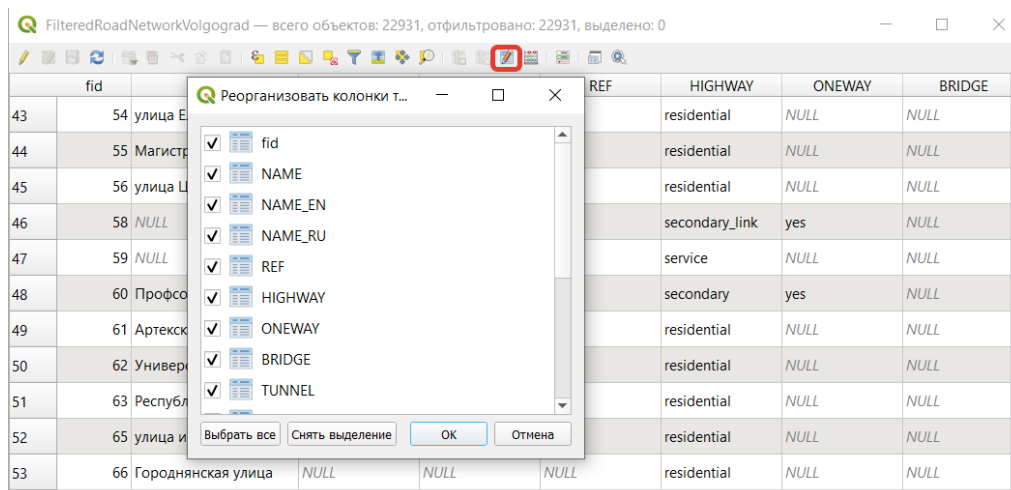


Рисунок 9 – Реорганизация таблицы атрибутов слоя дорожной

В столбцах **ONEWAY** и **MAXSPEED** должна содержаться информация о разрешённых направлениях движения и разрешённой скорости движения. Однако мы видим, что они заполнены далеко не везде.

Поэтому создадим два вычисляемых поля: **CalcOneway** и **CalcMaxSpeed**.

Значения поля **CalcOneway** будем вычислять на основе значений записей в поле **ONEWAY** (рисунок 10).

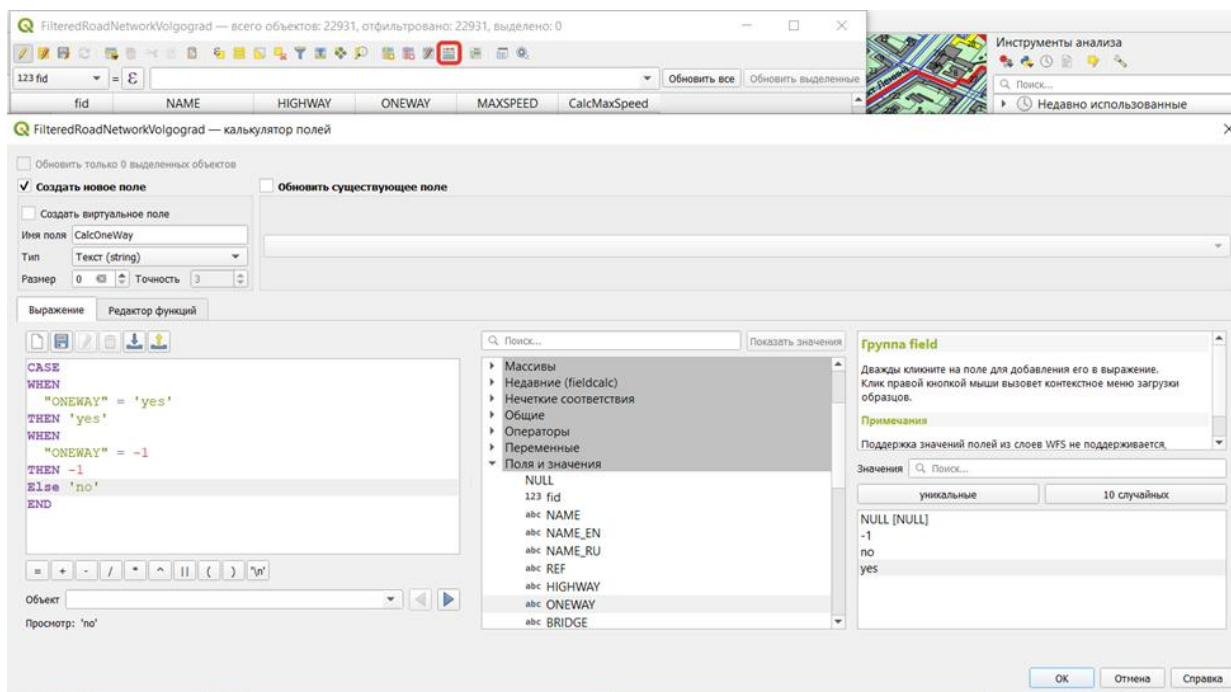


Рисунок 10 – Вычисление значений поля **CalcOneway**

Значения поля **CalcMaxSpeed** будем вычислять на основе значений записей в поле **HIGHWAY** и соответствующих им максимальным значениям скоростей, взятым из таблицы 2 (рисунок 11).

Таблица 2. Значения атрибута "**HIGHWAY**" и соответствующие им максимальные скорости

Значения атрибута "HIGHWAY"	Максимальная скорость (км/ч)
living_street, service	10
residential	20
road, tertiary, tertiary_link, track, unclassified	40
primary, primary_link, secondary, secondary_link, trunk, trunk_link	60

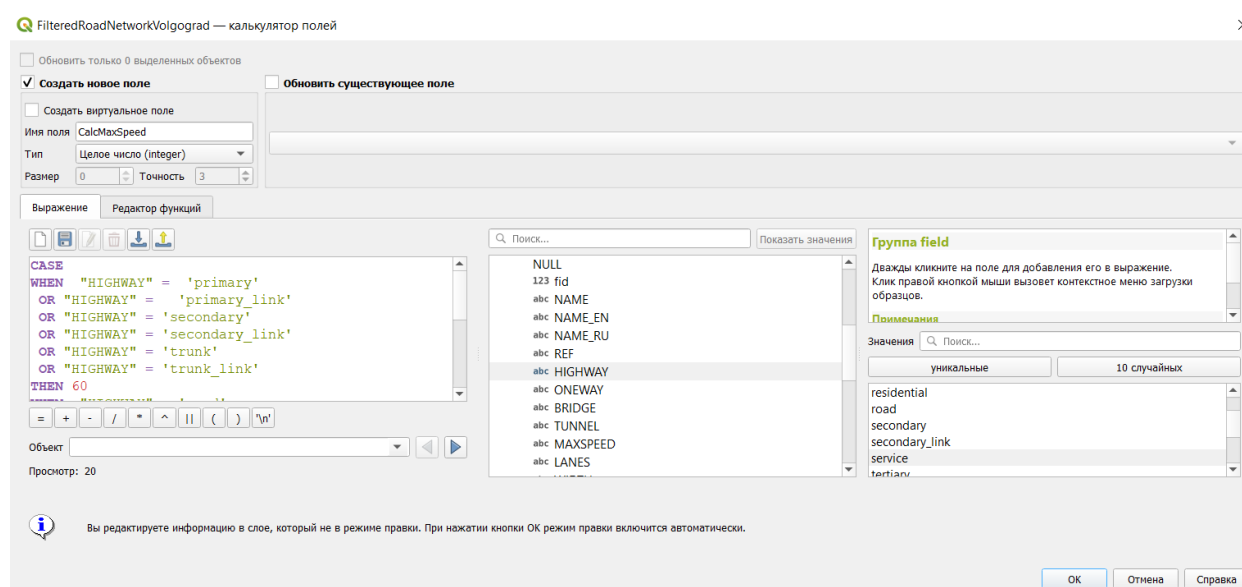


Рисунок 11 – Вычисление значений поля **CalcMaxSpeed**

Вернемся в интерфейс инструмента **Shortest path (point to point)** (напомним, что его не нужно было закрывать).

Изменим критерий оптимизации с кратчайшего пути на наиболее быстрый путь.

В настройке **Direction field** укажите, в каком поле содержится информация о разрешённых направлениях движения (**CalcOneway**).

В следующих полях введем значения, соответствующие разрешённым направлениям: **yes** для движения только в прямом

направлении, **-1** для движения только в обратном направлении, **no** для разрешения движения в обоих направлениях.

В настройке **Speed field** укажем, из какого поля следует взять разрешённую скорость движения (**CalcMaxSpeed**).

Снова запустим расчёт пути (рисунок 12).

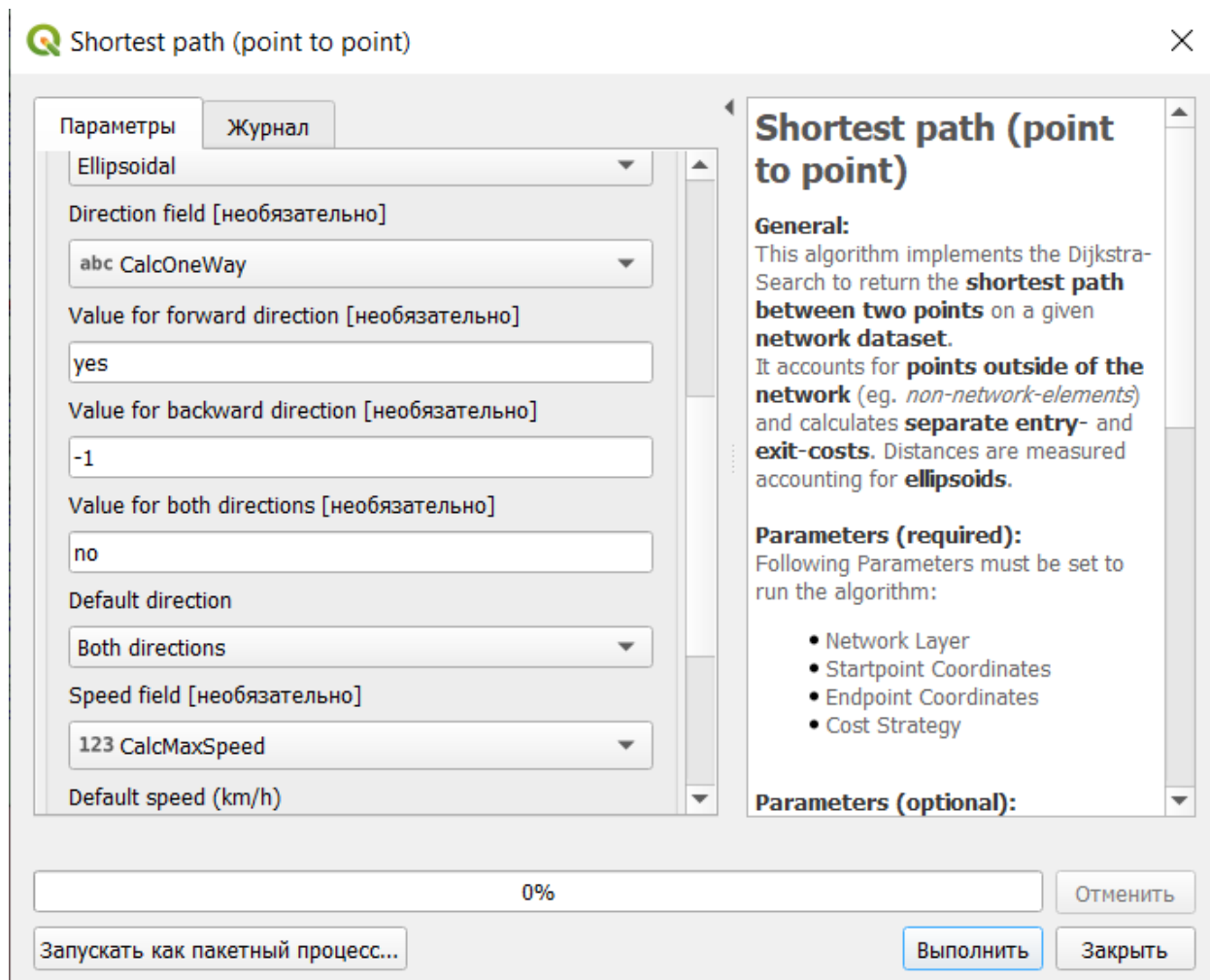


Рисунок 12 – Интерфейс настройки инструмента **Shortest path (point to point)** с новыми параметрами

Дождемся, пока инструмент завершит работу, и новый временный слой будет добавлен в проект. После этого окно инструмента можно закрыть.

Переименуем добавленный слой в **Fastest Path Layer** и изменим его отображение на линию зеленого цвета толщиной 1 мм (рисунок 13).



Рисунок 13 – Карта со слоями *Shortest Path Layer* и *Fastest Path Layer*

Самостоятельно повторите расчёт маршрутов по кратчайшему расстоянию и скорейшему времени для пары «источник-назначение», взяв в качестве «источника» вокзал Волгоград-1, а в качестве «точки назначения» место Вашего проживания. Сохраняйте построенные пути в «постоянные» наборы данных.

2.5 Лабораторная работа № 5. Методы и средства визуализации данных в ГИС

2.5.1 Цель работы:

В ходе данной работе студенты получают навыки работы с программным обеспечением для визуализации географически привязанной информации; навыками работы с видами слоёв электронных карт, узнают способы получения, обработки географически привязанной информации, представленной в векторном виде.

2.5.2 Порядок выполнения работы

Получение данных о рельефе. Одним из источников данных о рельефе является ресурс USGS EarthExplorer (рисунок 6.1), который предоставляет информацию для исследований бесплатно. В случае если у вас нет аккаунта на данном ресурсе – зарегистрируйтесь.

Для получения DEM файла перейдите на сайт EarthExplorer (<http://earthexplorer.usgs.gov>) и выберите интересующую вас область, примеры окон представлены на рисунках 3.5.1 - 3.5.3. Также необходимо будет пройти регистрацию. Для отображения данного формата через QGIS Desktop понадобится плагин QGIS2Threejs. Установите его, если необходимо.

1. Enter Search Criteria

To narrow your search area: type in an address or place name, enter coordinates or click the map to define your search area (for advanced map tools, view the [help documentation](#)), and/or choose a date range.

Geocoder KML/Shapefile Upload

Select a Geocoding Method
Address/Place

Search Limits: The search result limit is 100 records; select a Country, Feature Class, and/or Feature Type to reduce your chances of exceeding this limit.

US Features World Features

Feature Name
(use % as wildcard)

State
All

Feature Type
All

Show Clear

Polygon **Circle** Predefined Area

Circular polygons are created using a center point defined by decimal latitude and longitude values and a radius.

Center Latitude: 49.49239300333961 Center Longitude: 43.570854818510384

Radius: 50 Kilometers

Apply Clear Circle

Рисунок 3.5.1. Установка критериев поиска области исследования.

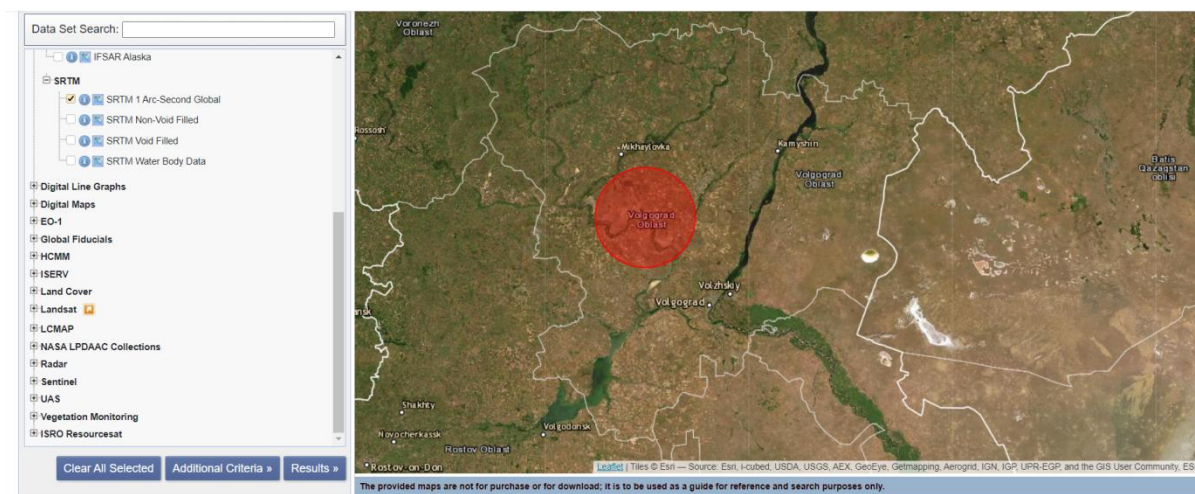


Рисунок 3.5.2. Установка фильтров по типу карт для выбора SRTM.

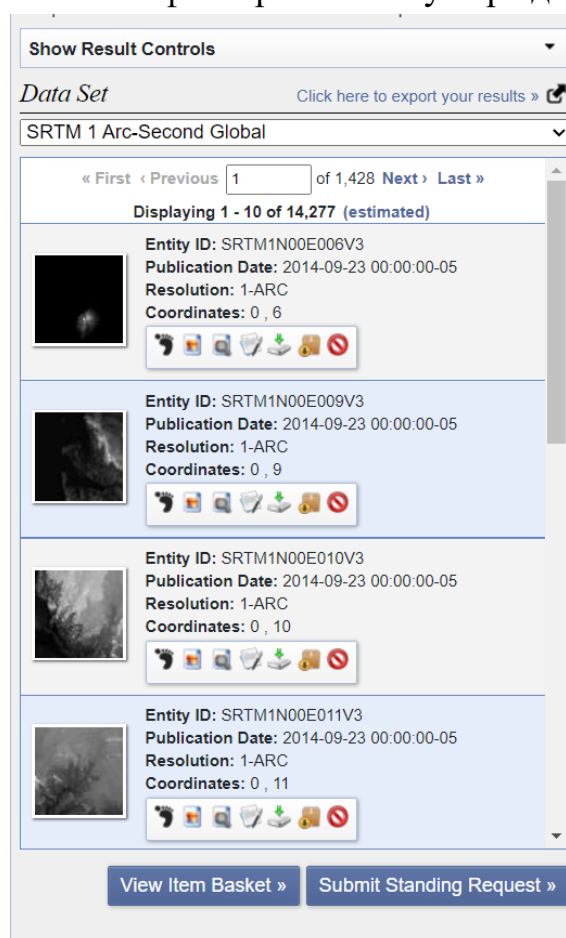


Рисунок 3.5.3. Выбор датасета для трехмерной визуализации данных.

Создадим трехмерную визуализацию. Откроем QGIS, далее импортируем файл, который мы получили ранее. В данном формате содержится геопривязка. На рисунках 3.5.4 - 3.5.5 показано как установить плагины. На рисунках 3.5.6 - 3.5.7 показано как добавить дефолтную карту к которой будет привязываться карта SRTM.

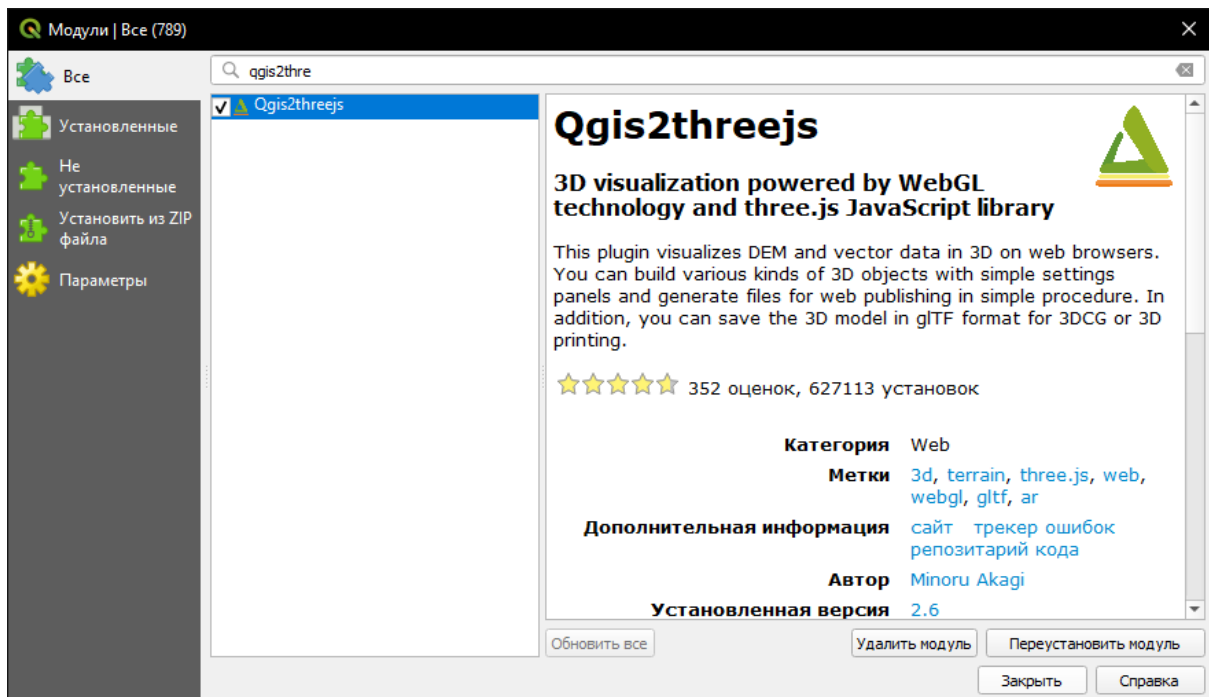


Рисунок 3.5.4. Установка плагина Qgis2threejs.

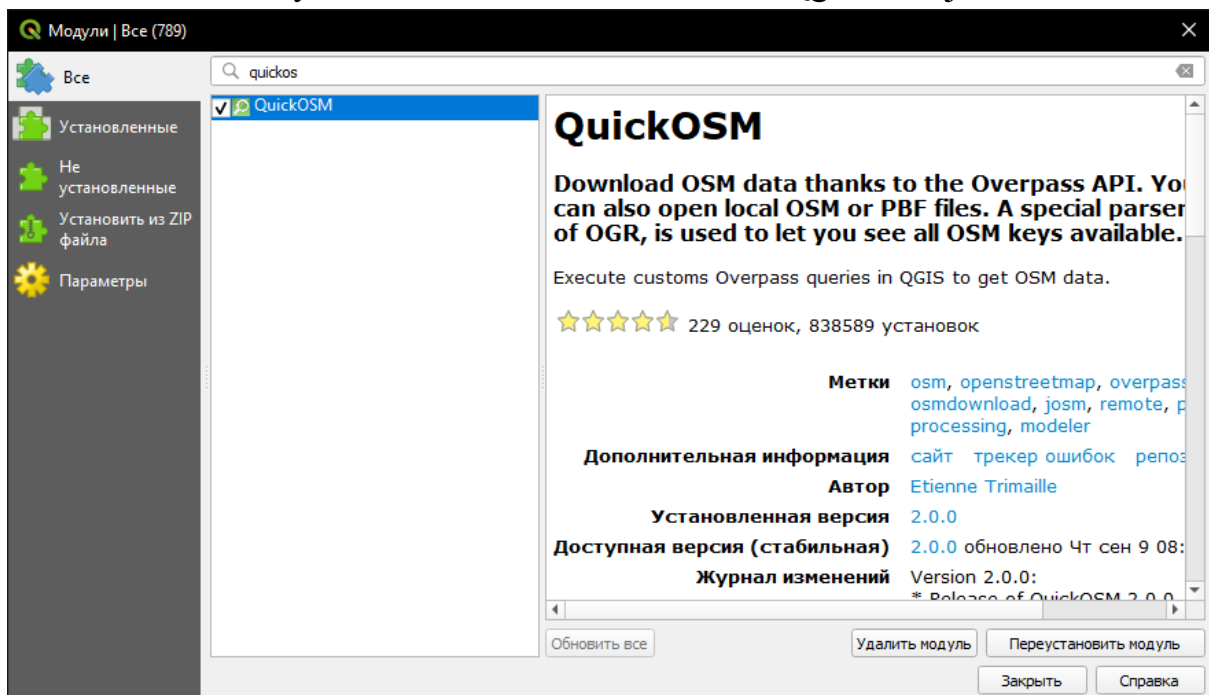


Рисунок 3.5.5. Установка плагина QuickOSM.

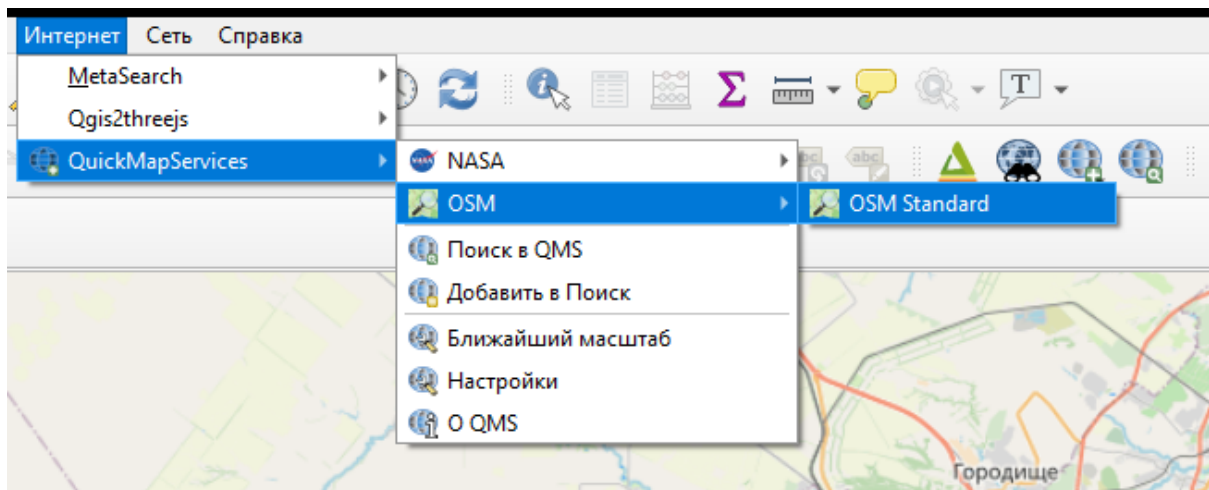


Рисунок 3.5.6. Добавление на канвас дефолтной карты.

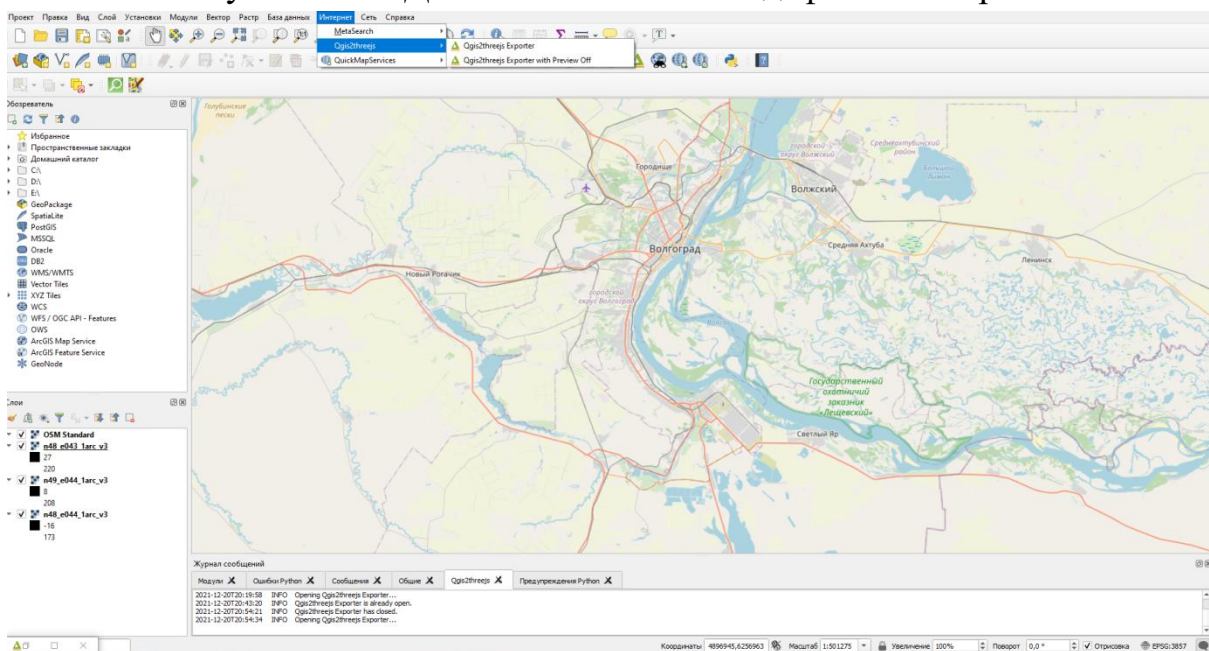


Рисунок 3.5.7. Карта из плагина QuickOSM.

После вывода дефолтной карты необходимо добавить карту рельефа. Для этого перенесем файл с картой в проект. Далее перейдем в плагин Qgis2threejs и изменим настройки так чтобы рельеф совпадал с действительными пропорциями местности. На рисунке 3.5.8 представлены настройки используемые для файлов с сайта <https://earthexplorer.usgs.gov/>, данные настройки могут отличаться при выборе другого датасета. На рисунке 3.5.9 представлена получившаяся карта, которая отображает рельеф местности.

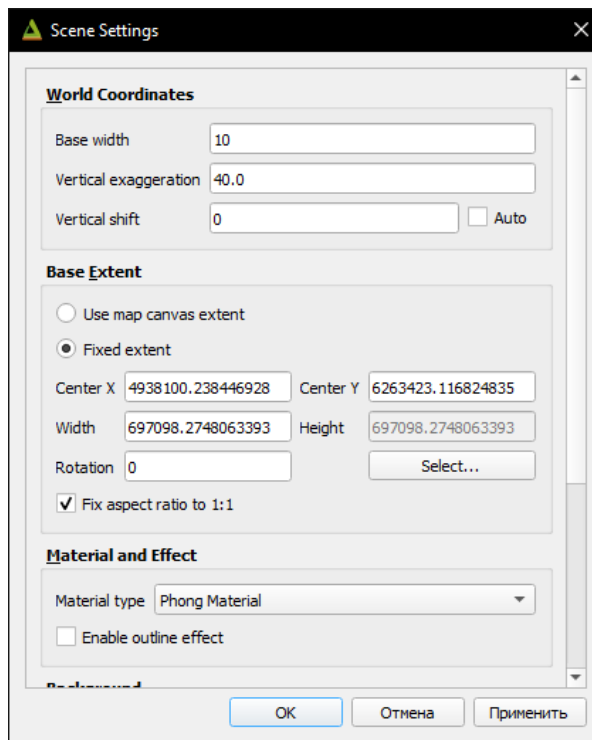


Рисунок 3.5.8. Настройка плагина Qgis2threejs.

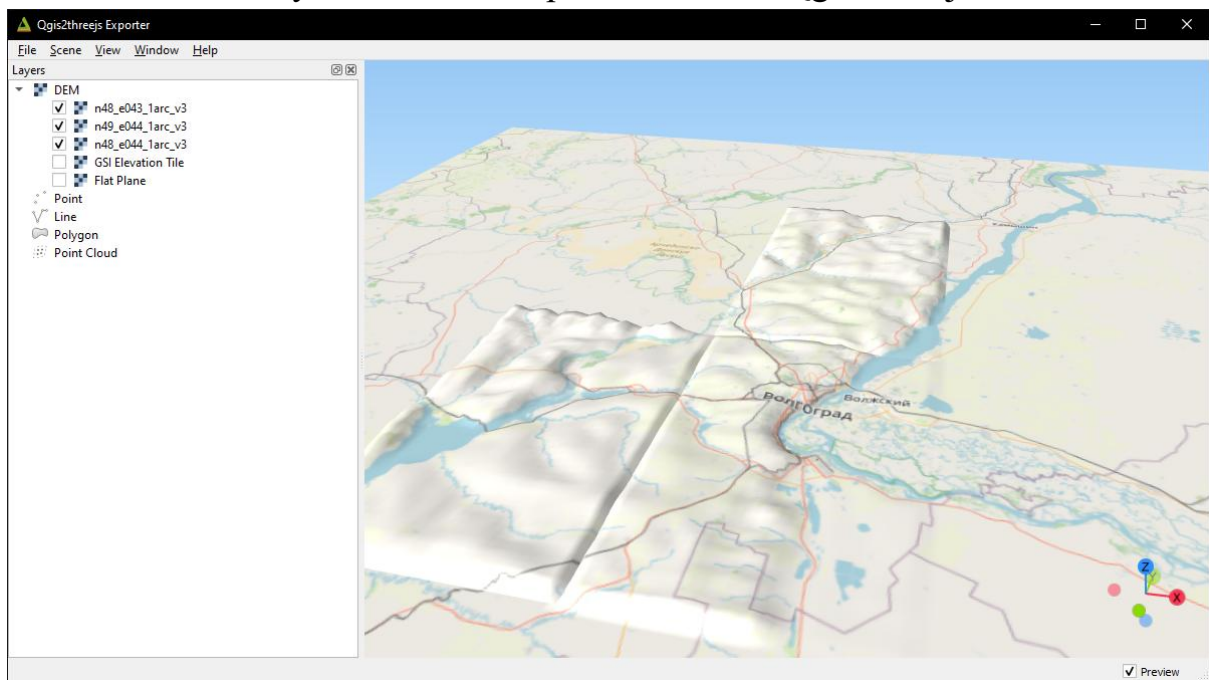


Рисунок 3.5.9. Визуализация рельефа местности.

Чтобы иметь возможность демонстрировать сделанную работу вне зависимости установлен QGIS или нет, необходимо экспортировать полученную карту в web. Для этого выберем “Export to Web”, После этого выберем директорию куда необходимо выгрузить необходимый файлы и нажмем Export.

При экспорте мы получим сгенерированную папку, которая определяет web-страницу, при открытии файла с расширением .html, мы

перейдем в браузер, в котором сможем детальнее рассмотреть полученную карту. На рисунке 3.5.11 отображена карта в web-версии.

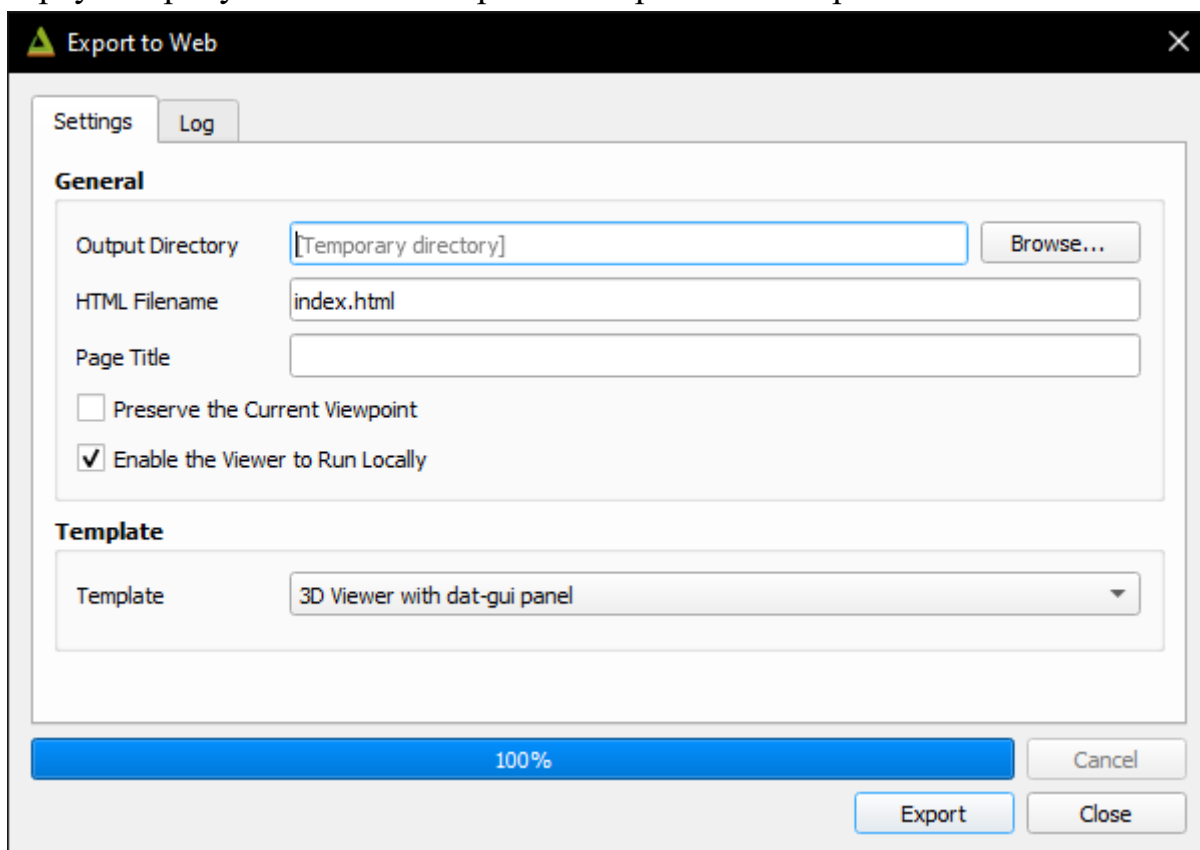


Рисунок 3.5.10. Экспорт в web-версию.

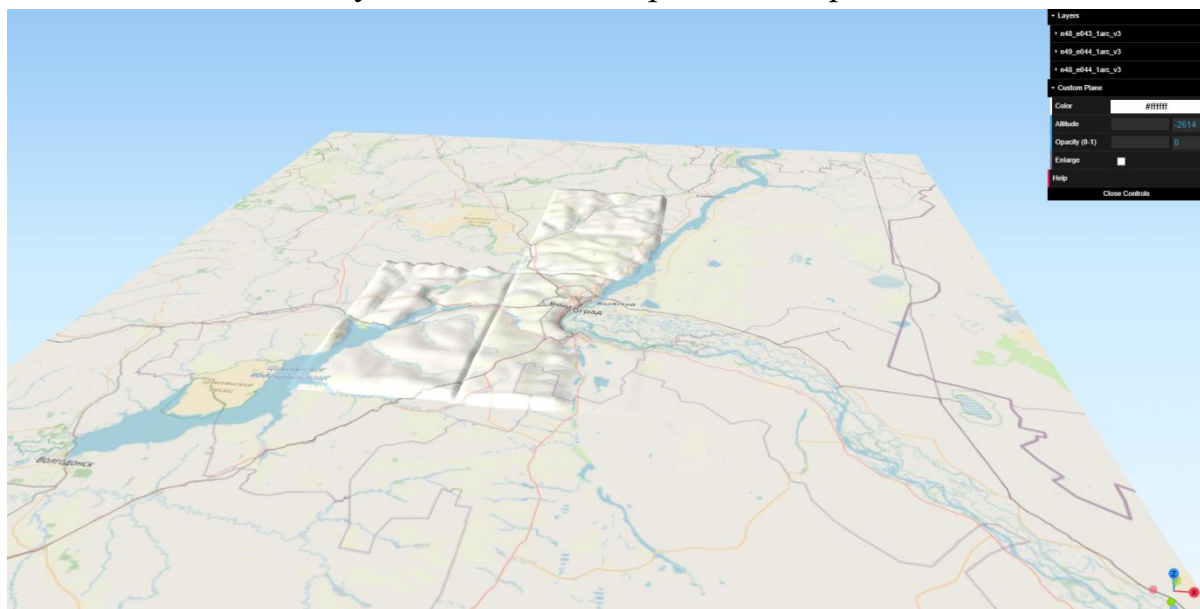


Рисунок 3.5.11. Визуализация рельефа местности в web-версии.

2.5.3 Теоретический материал

Рельеф – это важнейший и главный элемент ландшафта. Данные о рельефе преимущественно хранятся в растровом формате в виде спутниковых фотоснимков с информацией о высотности. DEM (digital elevation model) – представляет собой цифровую модель поверхности рельефа. Наличие дополнительного измерения в рамках трехмерной визуализации позволяет более эффективно представлять данные, имеющие несколько параметров в рамках одной карты. Однако при использовании объемных визуализаций следует учесть особенности рельефа изучаемой области, либо отказаться от отображения рельефных особенностей территории.

2.5.4 Практические задания

Отчет должен быть подготовлен в соответствии с шаблоном (Приложение 1) и содержать следующие снимки браузера: трехмерную карту с рельефом, карты с трехмерными данными всех типов.

Самостоятельно подготовьте трехмерную карту любого района или ограниченной территории (например, острова) Волгоградской области. Полученная веб страница должна быть запакована и приложена к отчету. В отчете должны содержаться снимки экрана с данной картой.

2.5.5 Рекомендуемая литература

1. Брынь М. Я., Богомолова Е. С., Коугия В. А., Левин Б. А. Инженерная геодезия и геоинформатика. Краткий курс. СПб: Лань, 2015. 288 с.
2. Берлянт А. М. Картография: Учебник для вузов. М.: Аспект Пресс, 2002. С. 71–92.3.
3. Берлянт А. М. Теория геоизображений. М.: Геос, 2006. С. 181–187.
4. Геоинформатика: Учеб. для студ. вузов / Е. Г. Капралов, А. В. Кошкарев, В. С. Тикунов и др.; Под ред. В. С. Тикунова. М.: Издательский центр «Академия», 2005. 480 с.

2.5.6 Вопросы к отчету

1. Что такое DEM формат? Что будет если открыть файл такого формата в графическом редакторе?
2. В чем преимущества и недостатки трехмерной визуализации данных?
3. В каких случаях использование трехмерной визуализации является целесообразным? Приведите не менее трех примеров.

2.6 Лабораторная работа № 6. Классификации мультиспектральных снимков со спутника

Это пошаговая инструкция по классификации мультиспектральных снимков со спутника Landsat 5. Сегодня в ряде сфер глубокое обучение доминирует как инструмент для решения сложных проблем, в том числе геопространственных. Надеюсь, вы знакомы с датасетами спутниковых снимков, в частности, Landsat 5 TM. Если вы немного разбираетесь в работе алгоритмов машинного обучения, то это поможет вам быстро освоить это руководство. А для тех, кто не разбирается, будет достаточно знать, что, по сути, машинное обучение заключается в установлении взаимосвязей между несколькими характеристиками (набором признаков X) объекта с другим его свойством (значением или меткой, — целевой переменной Y). Мы подаём на вход модели много объектов, для которых известны признаки и значение целевого показателя/класса объекта (размеченные данные) и обучаем ее так, чтобы она могла спрогнозировать значение целевой переменной Y для новых данных (неразмеченных).

В чём заключается основная проблема со спутниковыми снимками?

Два и более класса объектов (например, застройки, пустыри и котлованы) на спутниковых снимках могут иметь одинаковые спектральные характеристики значения, поэтому в последние лет двадцать их классификация представляет собой трудную задачу.

Из-за этого возможно использовать классические модели машинного обучения с учителем и без, но их качество будет далеко от идеала. Они всегда обладают одними и теми же недостатками.

Если в качестве классификатора использовать вертикальную линию и двигать её вдоль оси X, то классифицировать изображения домов будет непросто. Данные распределены так, что разделить их на классы с помощью одной вертикальной линии невозможно (в таких случаях говорят, что «объекты разных классов не являются линейно разделимыми»). Но это не означает, что дома вообще нельзя классифицировать!

Эта проблема домов и деревьев аналогична проблеме с застройкой, пустырями и котлованами. Приоритет метрик классификации спутниковых снимков может меняться в зависимости от задачи. Например, если вам нужно удостовериться, что все застроенные территории без исключений классифицированы как застройка, и вы готовы смириться с наличием пикселей других классов с аналогичными сигнатурами, которые тоже будут классифицированы как застройка, тогда вам потребуется модель с высокой полнотой. А если вам важнее классифицировать именно застройку, без добавления пикселей других классов, и вы готовы отказаться от классификации смешанных территорий, тогда выбирайте классификатор с высокой точностью. В случае с домами и деревьями обычная модель будет использовать красную линию, выдерживая баланс между точностью и полнотой

Используемые данные

В качестве признаков мы будем использовать значения шести диапазонов (band 2 — band 7) изображения из Landsat 5 TM, и попытаемся спрогнозировать двоичный класс застройки. Для обучения и тестирования будут использованы многоспектральные данные (изображения и слой с бинарным классом застройки) с Landsat 5 за 2011 года для Бангалора. А для прогнозирования будут использованы многоспектральные данные Landsat 5, полученные в 2005-м в Хайдарабаде.

Поскольку мы используем для обучения размеченные данные, это называется обучение с учителем.

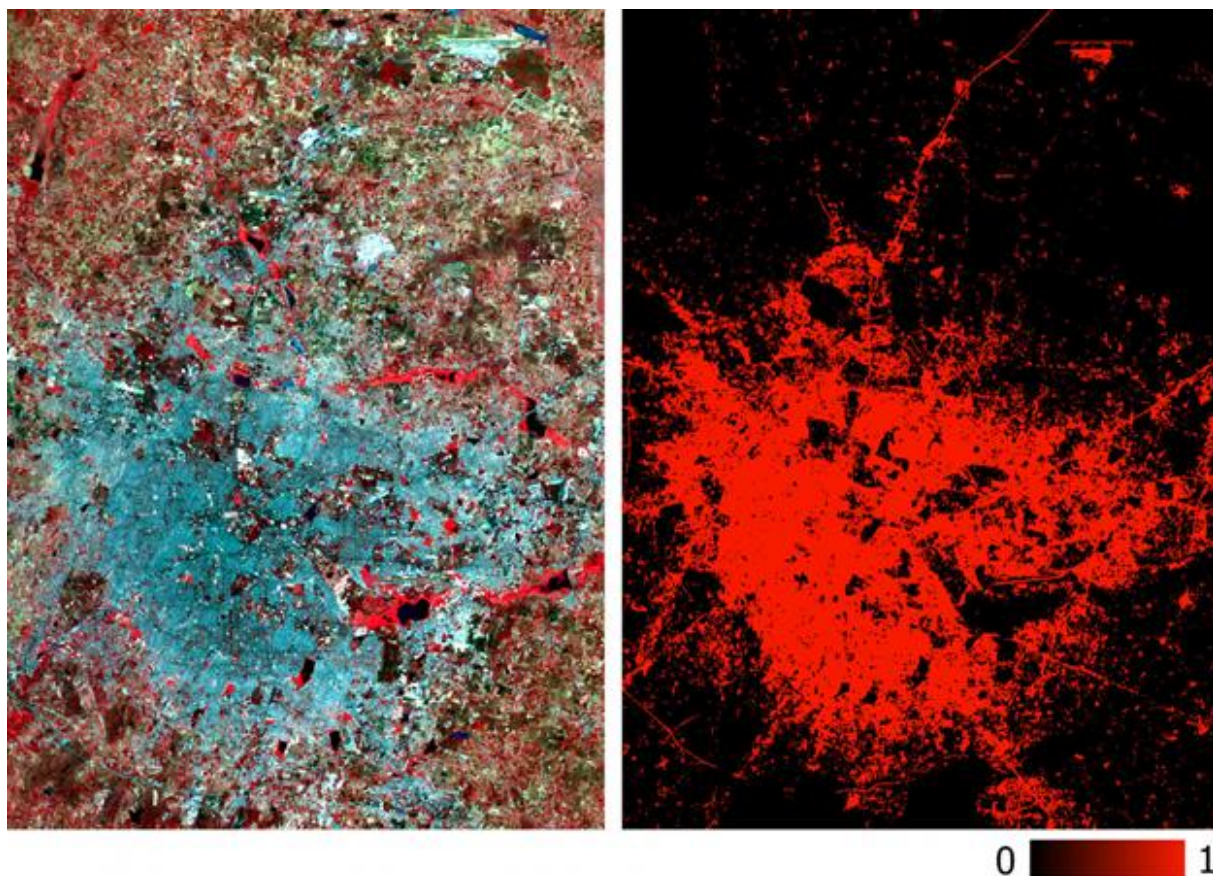


Рис 1. Многоспектральные обучающие данные и соответствующий двоичный слой с застройкой.

Для создания нейросети воспользуемся Python—библиотекой Google Tensorflow. Также нам потребуются эти библиотеки:

- `pyrsgis` — для чтения и записи GeoTIFF.
- `scikit-learn` — для предобработки данных и оценки точности.
- `numpy` — для базовых операций с массивами.

А теперь без дальнейших проволочек давайте писать код.

Положите все три файла в директорию, пропишите в скрипте путь и названия входных файлов, а затем прочитайте файлы GeoTIFF.

```
import os
from pyrsgis import raster

os.chdir("E:\\yourDirectoryName")
mxBangalore = '15_Bangalore2011_raw.tif'
builtupBangalore = '15_Bangalore2011_builtup.tif'
mxHyderabad = '15_Hyderabad2011_raw.tif'

# Read the rasters as array
ds1, featuresBangalore = raster.read(mxBangalore, bands='all')
ds2, labelBangalore = raster.read(builtupBangalore, bands=1)
ds3, featuresHyderabad = raster.read(mxHyderabad, bands='all')
```

Модуль `raster` из пакета `pyrsgis` считывает геолокационные данные GeoTIFF и значения цифровых номеров (DN) в виде отдельных NumPy-массивов.

Теперь выведем на экран размер прочитанных данных.

```
print("Bangalore multispectral image shape: ", featuresBangalore.shape)
print("Bangalore binary built-up image shape: ", labelBangalore.shape)
print("Hyderabad multispectral image shape: ", featuresHyderabad.shape)
```

Результат

Bangalore multispectral image shape: 6, 2054, 2044

Bangalore binary built-up image shape: 2054, 2044

Hyderabad multispectral image shape: 6, 1318, 1056

Как видите, в изображениях Бангалора такое же количество строк и столбцов, как и в бинарном слое (соответствующем застройке). Количество слоёв в многоспектральных изображениях в Бангалоре и Хайдарабаде тоже совпадает. Модель будет учиться решать, какие пиксели относятся к застройке, а какие нет, на основе соответствующих значений по всем 6-ти спектрам. Поэтому многоспектральные изображения должны иметь одинаковое количество признаков (диапазонов), перечисленных в одном и том же порядке.

Теперь превратим массивы в двумерные, где каждая строка представляет отдельный пиксель, потому что это нужно для работы большинства алгоритмов машинного обучения. Сделаем мы это с помощью модуля `convert` из пакета `pyrsgis`.

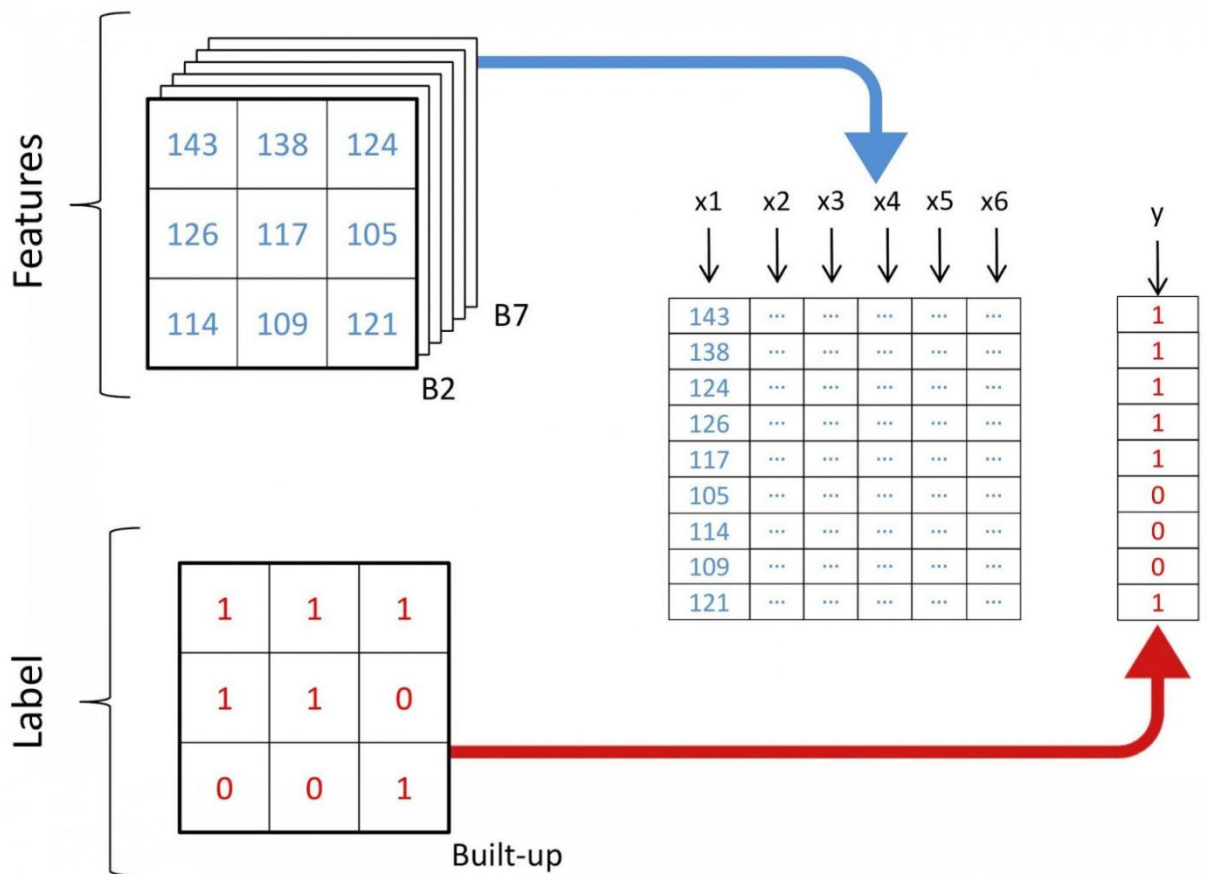


Рис 1. Схема реструктуризации данных.

```
from pyrsgis.convert import changeDimension
```

```
featuresBangalore = changeDimension(featuresBangalore)
```

```
labelBangalore = changeDimension (labelBangalore)
```

```
featuresHyderabad = changeDimension(featuresHyderabad)
```

```
nBands = featuresBangalore.shape[1]
```

```
labelBangalore = (labelBangalore == 1).astype(int)
```

```
print("Bangalore multispectral image shape: ", featuresBangalore.shape)
```

```
print("Bangalore binary built-up image shape: ", labelBangalore.shape)
```

```
print("Hyderabad multispectral image shape: ", featuresHyderabad.shape)
```

Результат

Bangalore multispectral image shape: 4198376, 6

Bangalore binary built-up image shape: 4198376

Hyderabad multispectral image shape: 1391808, 6

В седьмой строке мы извлекли все пиксели со значением 1. Это помогает избежать проблем с пикселями без информации (NoData), которые часто имеют крайне высокие или низкие значения.

Теперь разделим данные на обучающую и валидационную выборки. Это нужно для того, чтобы модель не видела тестовых данных и работала так же хорошо с новой информацией. Иначе модель переобучится и будет хорошо работать только на обучающих данных.

```
from sklearn.model_selection import train_test_split
```

```
xTrain, xTest, yTrain, yTest = train_test_split(featuresBangalore, labelBangalore, test_size=0.4, random_state=42)
```

```
print(xTrain.shape)
```

```
print(yTrain.shape)
```

```
print(xTest.shape)
```

```
print(yTest.shape)
```

Результат

```
(2519025, 6)
```

```
(2519025,)
```

```
(1679351, 6)
```

```
(1679351,)
```

```
test_size=0.4
```

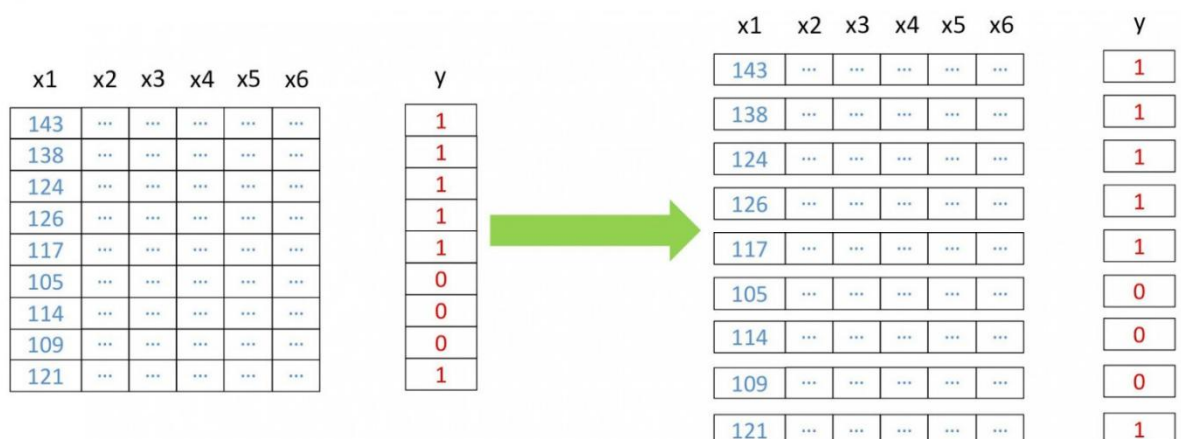
Данные разделены на обучающие и валидационные в соотношении 60/40.

Многим алгоритмам машинного обучения, включая нейросети, нужны нормализованные данные. Это означает, что они должны быть распределены в рамках заданного диапазона (в данном случае от 0 до 1). Поэтому, чтобы выполнить это требование, мы нормализуем признаки. Сделать это можно с помощью извлечения минимального значения с

последующим делением на разброс (разницу между максимальным и минимальным значениями). Поскольку датасет Landsat является восьмибитным, минимальное и максимальное значение будут равны 0 и 255 ($2^8 = 256$ значений).

Обратите внимание, что для нормализации всегда лучше вычислять минимальные и максимальные значения на основе данных. Для упрощения задачи мы будем придерживаться восьмибитного диапазона по умолчанию.

Ещё один этап предварительной обработки заключается в преобразовании матриц признаков из двумерных в трёхмерные, чтобы модель воспринимала каждую строку как отдельный пиксель (отдельный объект обучения).



```
# Normalise the data
```

```
xTrain = xTrain / 255.0
```

```
xTest = xTest / 255.0
```

```
featuresHyderabad = featuresHyderabad / 255.0
```

```
# Reshape the data
```

```
xTrain = xTrain.reshape((xTrain.shape[0], 1, xTrain.shape[1]))
```

```

xTest = xTest.reshape((xTest.shape[0], 1, xTest.shape[1]))

featuresHyderabad = featuresHyderabad.reshape((featuresHyderabad.shape[0], 1,
featuresHyderabad.shape[1]))

# Print the shape of reshaped data

print(xTrain.shape, xTest.shape, featuresHyderabad.shape)

```

Результат

```
(2519025, 1, 6) (1679351, 1, 6) (1391808, 1, 6)
```

Всё готово, давайте соберём нашу модель с помощью keras. Для начала воспользуемся sequential-моделью, добавляя слои один за другим. У нас будет один входной слой с количеством узлов, равным числу диапазонов (nBands) — в нашем случае их 6. Также будем использовать один скрытый слой с 14 узлами и функцией активации ReLu. Последний слой состоит из двух узлов для определения двоичного класса застройки с функцией активации softmax, которая подходит для вывода категоризированного результата.

```

from tensorflow import keras

# Define the parameters of the model

model = keras.Sequential([

    keras.layers.Flatten(input_shape=(1, nBands)),

    keras.layers.Dense(14, activation='relu'),

    keras.layers.Dense(2, activation='softmax')])

# Define the accuracy metrics and parameters

model.compile(optimizer="adam", loss="sparse_categorical_crossentropy", metrics=["accuracy"])

# Run the model

model.fit(xTrain, yTrain, epochs=2)

```

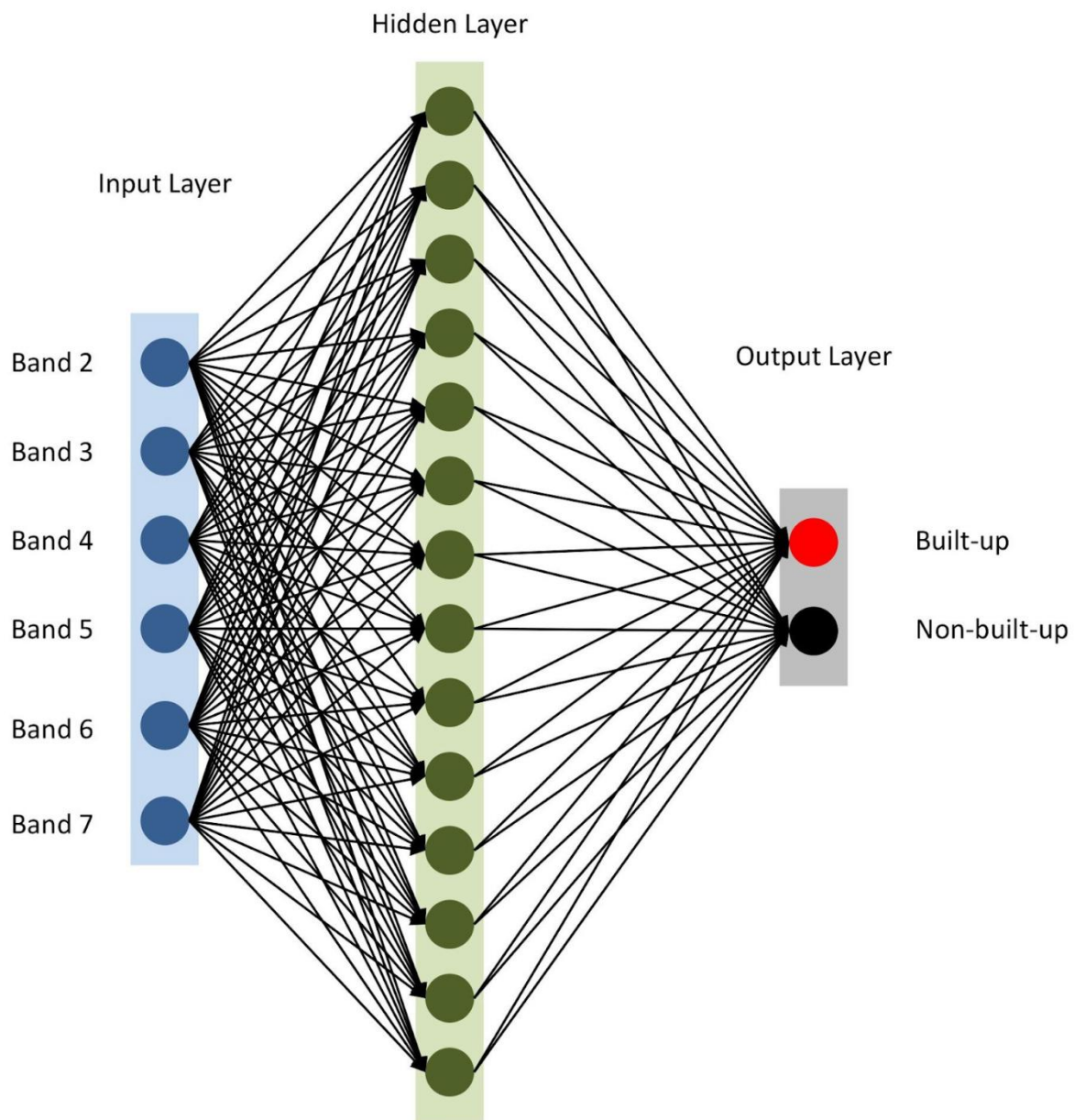


Рис Архитектура нейросети

Как упоминалось в строке 10, мы задаем `adam` в качестве оптимизатора модели (есть и несколько других). В данном случае будем использовать в качестве функции потерь перекрестную энтропию. Для оценки качества работы модели будем использовать метрику `assigasy`.

Наконец, запустим обучение нашей модели в течение двух эпох (или итераций) на `xTrain` и `yTrain`. Это займёт какое-то время, в зависимости от

размера данных и вычислительной мощности. Вот что вы увидите после компиляции

```
2019-08-28 19:41:09.299466: I tensorflow/core/platform/cpu_feature_guard.cc:142] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX2
Epoch 1/2
2519025/2519025 [=====] - 52s 21us/sample - loss: 0.1285 - accuracy: 0.9515
Epoch 2/2
2519025/2519025 [=====] - 52s 21us/sample - loss: 0.1076 - accuracy: 0.9593
```

Давайте спрогнозируем значения для валидационных данных, которые мы храним отдельно, и рассчитаем различные метрики точности.

```
from sklearn.metrics import confusion_matrix, precision_score, recall_score
```

```
# Predict for test data
yTestPredicted = model.predict(xTest)
yTestPredicted = yTestPredicted[:,1]

# Calculate and display the error metrics
yTestPredicted = (yTestPredicted>0.5).astype(int)
cMatrix = confusion_matrix(yTest, yTestPredicted)
pScore = precision_score(yTest, yTestPredicted)
rScore = recall_score(yTest, yTestPredicted)

print("Confusion matrix: for 14 nodes\n", cMatrix)
print("\nP-Score: %.3f, R-Score: %.3f" % (pScore, rScore))
```

Функция `softmax` генерирует отдельные столбцы для значений вероятности для каждого класса. Мы используем только значения для первого класса («есть застройка»), как видно по шестой строке приведенного выше кода. Оценивать работу моделей геопространственного анализа не так просто, в отличие от других классических задач машинного обучения. Будет нечестно полагаться на обобщенную суммарную ошибку. Ключом к успешной модели является пространственное расположение. Таким образом, матрица ошибок (`confusion matrix`), точность и полнота могут дать более корректное представление о качестве работы модели.

```
Confusion matrix: for 14 nodes
[[1443164  37712]
 [ 32062 166413]]
P-Score: 0.815, R-Score: 0.838
```

Как видно из confusion matrix, есть тысячи пикселей, которые относятся к застройке, но классифицированы иначе, и наоборот. Однако их доля от общего объёма данных не слишком велика. Точность и полнота на тестовых данных превысили порог в 0,8.

Вы можете потратить больше времени и выполнить несколько итераций для поиска оптимального количества скрытых слоев, количества узлов в каждом скрытом слое, а также количества эпох для достижения желаемой точности. По мере необходимости, в качестве признаков можно использовать индексы дистанционного зондирования вроде NDBI или NDWI. При достижении желаемой точности используйте модель для прогнозирования застройки на основе новых данных и экспортируйте результат в GeoTIFF. Для подобных задач можно использовать аналогичную модель с небольшими изменениями.

```
predicted = model.predict(feature2005)
predicted = predicted[:,1]

#Export raster
prediction = np.reshape(predicted, (ds.RasterYSize, ds.RasterXSize))
outFile = 'Hyderabad_2011_BuiltupNN_predicted.tif'
raster.export(prediction, ds3, filename=outFile, dtype='float')
```

Доступ к данным: <https://github.com/PratyushTripathy/Landsat-Classification-Using-Neural-Network>

3. МЕТОДИЧЕСКИЕ УКАЗАНИЯ К КОНТРОЛЬНОЙ РАБОТЕ “МОДЕЛИРОВАНИЕ НА ОСНОВЕ ПРОСТРАНСТВЕННЫХ ДАННЫХ”

3.1 Цель работы

Целью работы: получить опыт в моделировании перемещений и взаимодействий объектов на основе пространственных данных.

3.2 Порядок выполнения работы

3.2.1 Настройка и запуск системы моделирования OSMLS

Для запуска инфраструктуры системы моделирования требуется:

- 1) установить Docker;
- 2) переключить Docker на работу с linux контейнерами (должно быть по-умолчанию);
- 3) установить Git;
- 4) клонировать репозиторий основной программы, используя команду `“git clone --branch 21.03b https://bitbucket.org/BUROVIC/osmlifesimulation”`;
- 5) выполнить переход в директорию программы командой `“cd osmlifesimulation”`;
- 6) в файле Dockerfile (в последней строке) клонированного репозитория изменить путь к файлу карты на любой другой, чтобы изменить участок карты, на котором будет происходить моделирование (опционально);
- 7) создать docker image, выполнив команду: `“docker build --pull -t osmls .”`;

8) запустить docker контейнер с помощью команды “docker run --rm -it -p 80:80 osmls” (для выбора другого порта требуется указать, например 8000:80);

9) перейти в браузере по адресу “http://localhost/index.html”, может понадобиться также указать порт, если он был изменен на предыдущем шаге;

10) для сборки модулей требуется скачать и установить .NET Core SDK (версия 3.1 LTS рекомендуется для создания новых модулей);

11) выполнить сборку требуемых модулей из исходных файлов стандартными средствами .NET.

3.2.2 Создание и запуск тестового модуля

Для создания модулей можно использовать любую IDE или редактор кода, например: JetBrains Rider, Visual Studio или Visual Studio Code.

В данном примере создание модуля будет продемонстрировано в среде Visual Studio 2019 (в 2017 версии также не должно возникнуть проблем). Перед созданием модуля необходимо убедиться, что был установлен .NET Core SDK. Также, в Visual Studio Installer может понадобиться добавить пункт “Кроссплатформенная разработка .NET Core” (см. Рисунок А.1), если шаблон проекта “Class Library (.NET Core)” отсутствует при создании проекта.

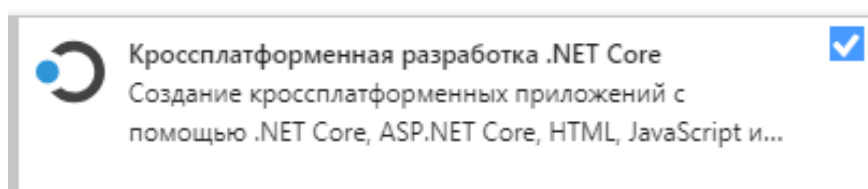


Рисунок А.1 - Добавление необходимых компонентов в Visual Studio Installer

Для создания модуля требуется:

1) создать новый проект типа Class Library (.NET Core), как показано на рисунке А.2;

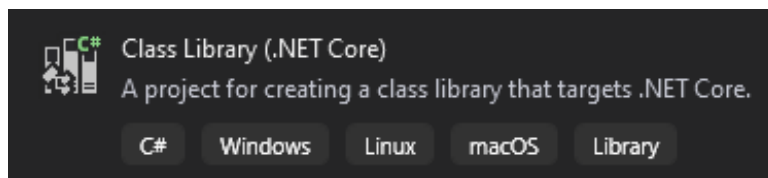


Рисунок А.2 - Добавление необходимых компонентов в Visual Studio Installer

2) задать произвольное имя проекта (в данном примере будет использоваться имя TestModule), как показано на рисунке А.3;

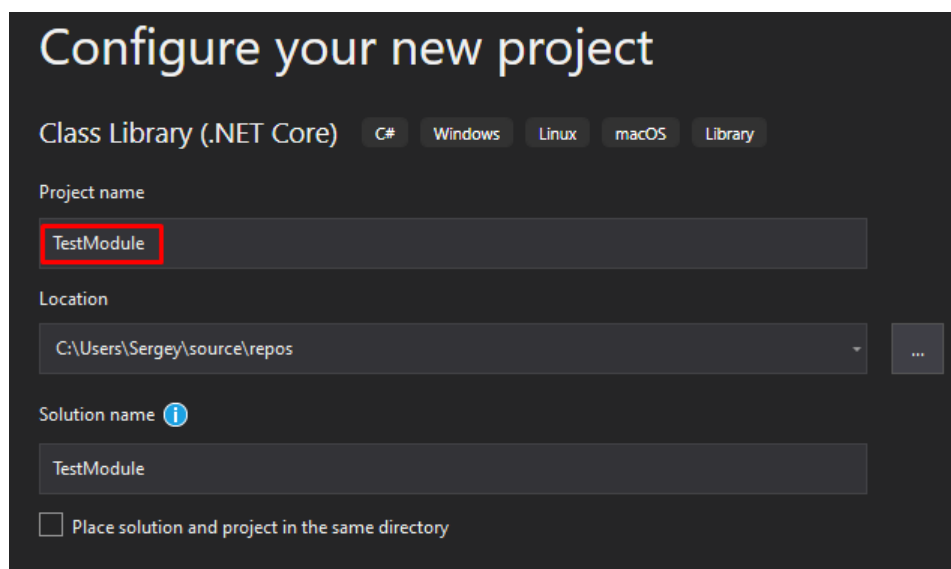


Рисунок А.3 - Назначение имени проекта

3) получить файл сборки глобальной библиотеки, выполнив команду “dotnet publish ./OSMLSGlobalLibrary -c release -o ./release” в корневой директории системы моделирования (osmlifesimulation), файл OSMLSGlobalLibrary.dll будет создан в директории release;

4) перенести файл сборки в директорию проекта модуля;

5) добавить в созданный проект модуля зависимость на файл OSMLSGlobalLibrary.dll, как показано на рисунках А.4, А.5.

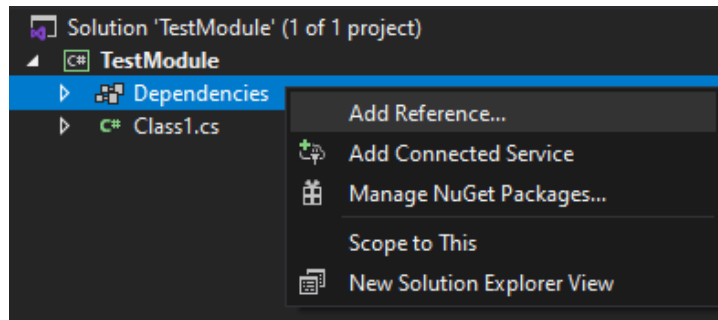


Рисунок А.4 - Добавление зависимости в проект

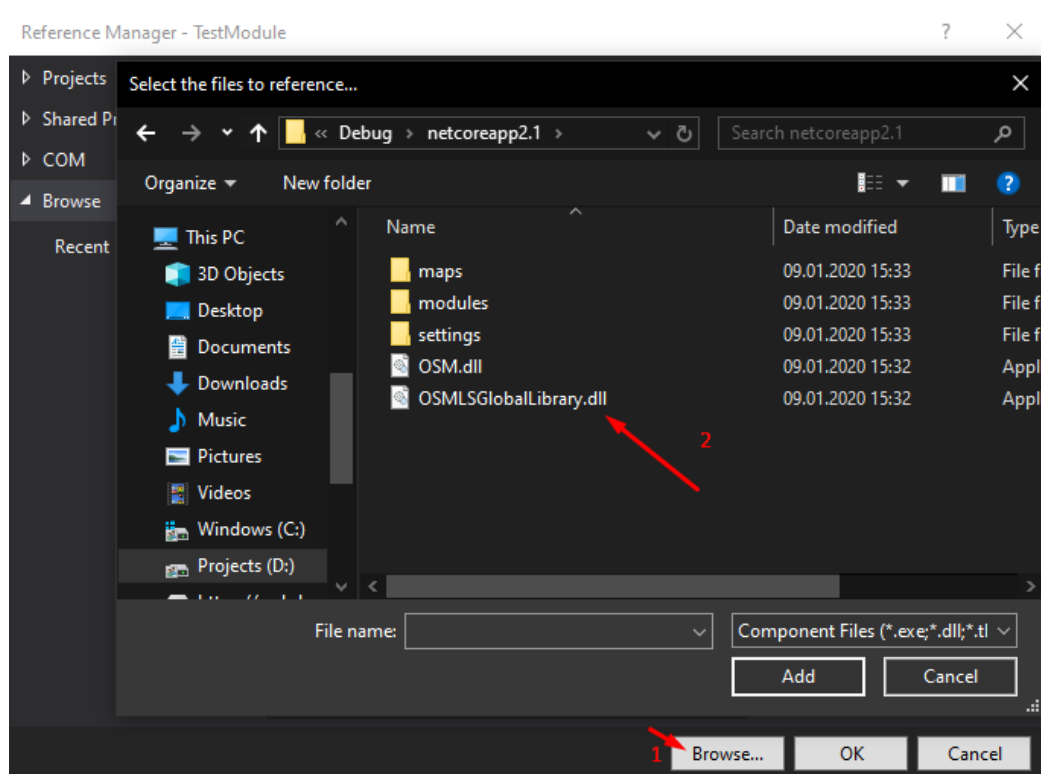


Рисунок А.5 - Выбор добавляемой в проект зависимости

б) в консоли, находясь в директории проекта, выполнить команды “dotnet add TestModule package NetTopologySuite --version 2.0.0” и “dotnet add TestModule package NetTopologySuite.Features --version 2.0.0”, чтобы установить библиотеку NetTopologySuite, необходимую для работы со всеми геометрическими объектами в проекте;

7) переименовать созданный по-умолчанию класс Class1.cs в TestModule.cs, как показано на рисунках А.6 и А.7;

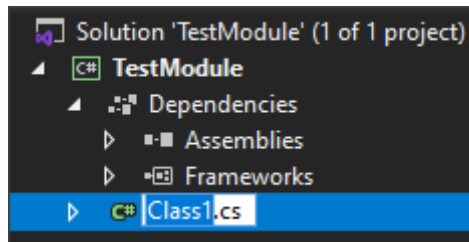


Рисунок А.6 - Переименование файла класса

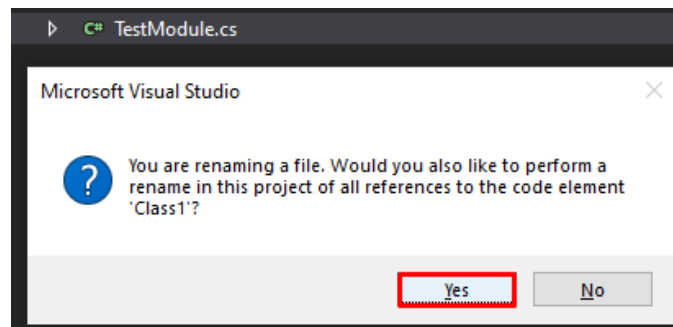


Рисунок А.7 - Подтверждение переименования

8) разместить в созданном классе код тестового модуля, доступный по ссылке <https://bitbucket.org/osmlifesimulation/testmodule/src/21.03b/TestModule/TestModule.cs>, проанализировать код;

9) выполнить сборку тестового модуля с помощью команды “dotnet publish -c release -o release” и найти в директории release файл сборки (TestModule.dll);

10) добавить файл тестового модуля в систему моделирования OSMLS (с помощью кнопки Add Assemblies и выбора модуля из выпадающего списка модулей, как показано на рисунке А.8) и перезапустить её (нажатием на кнопку Stop, а затем на кнопку Run), проанализировать результат.

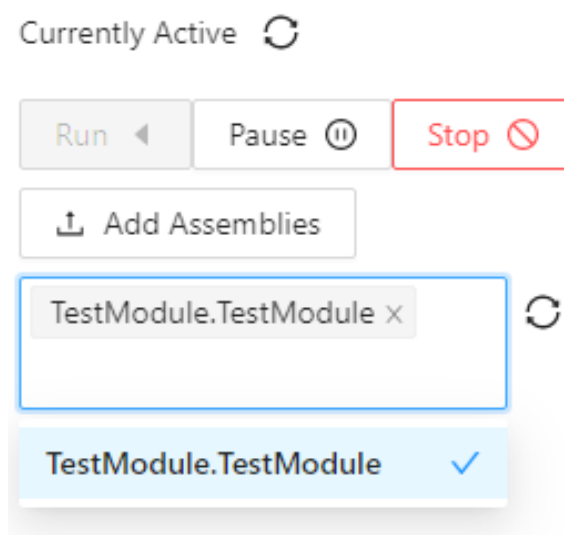


Рисунок А.8 - Интерфейс управления статусом моделирования

3.2.3 Создание и запуск модуля на основе индивидуального задания

Требуется разработать свой модуль в соответствии с индивидуальным заданием, для этого нужно:

- 1) выбрать вариант из раздела “практические задания”;
- 2) создать проект модуля с названием, соответствующим тематике варианта;
- 3) реализовать логику модуля;
- 4) протестировать работу модуля в связке с системой моделирования.

3.3 Теоретический материал

Все объекты в системе моделирования OSMLS наследуются от геометрических примитивов библиотеки NetTopologySuite. Данная библиотека имеет свой репозиторий и официальную документацию. Все взаимодействия между объектами (такие, как, например пересечение

линии и полигона) также можно выполнять с использованием данной библиотеки, вместо реализации своих дополнительных методов и функций.

Все объекты в системе моделирования OSMLS отображаются на клиенте с использованием библиотеки OpenLayers. Каждый тип (класс) объектов может иметь свой собственный стиль, задаваемый атрибутом CustomStyle. Если не задать объекту стиль, стиль по-умолчанию будет таким, как отображено на рисунке А.9.

```
new style.Style({
  image: new style.Circle({
    opacity: 1.0,
    scale: 1.0,
    radius: 3,
    fill: new style.Fill({
      color: 'rgba(255, 255, 255, 0.4)'
    }),
    stroke: new style.Stroke({
      color: 'rgba(0, 0, 0, 0.4)',
      width: 1
    }),
  }),
  fill: new style.Fill({
    color: 'rgba(255, 255, 255, 0.4)'
  }),
  stroke: new style.Stroke({
    color: 'rgba(0, 0, 0, 0.4)',
    width: 1
  })
});
```

Рисунок А.9 - Openlayers стиль объекта по-умолчанию.

3.4 Практические задания

В связи с тем, что для реализации взаимодействий между объектами городской среды требуется использование других модулей (таких как модуль поиска путей и модуль городских объектов), в данном разделе будут описаны задачи требующие несложных объектных взаимодействий и их отображения на карте.

Все объекты, создаваемые в рамках задач, должны быть описаны отдельными классами и иметь отличный от стандартного стиль отображения, заданный атрибутом CustomStyle.

В таблице А.1 представлены индивидуальные задания на разработку модулей, сгруппированные по индивидуальным вариантам.

Таблица А.1 - Индивидуальные задания на разработку модулей по вариантам

№	Объект	Базовый тип	Количество	Действия
1	дикая местность	Polygon	1	Постоянно генерирует волков и олений в случайной точке себя.
	волк	Point	много	Пытается догнать и съесть ближайших олений . Не может выйти за пределы дикой местности .
	олень	Point	много	Двигается хаотично. Не может выйти за пределы дикой местности .
2	дикая местность	Polygon	1	При инициализации генерирует внутри себя несколько природных барьеров . Постоянно генерирует волков и олений в случайной точке себя.
	природный барьер	Polygon	много	
	волк	Point	много	Пытается догнать и съесть ближайших олений . Не может выйти за пределы дикой местности . Не может пересечь природный барьер .
	олень	Point	много	Двигается хаотично. Не может выйти за пределы дикой местности . Не может пересечь природный барьер .
3	дикая местность	Polygon	1	Генерирует некоторое количество волков и олений при инициализации.
	волк	Point	много	Пытается догнать и съесть ближайших олений . Создает новых волков с волками противоположного пола. Не может выйти за пределы дикой местности.
	олень	Point	много	Создает новых олений с оленьями противоположного пола. Двигается хаотично. Не может выйти за пределы дикой местности .
4	дикая местность	Polygon	1	Постоянно генерирует волков и олений в случайной точке себя.
	волк	Point	много	Имеет постоянно растущий голод, при достижении определенной отметки умирает. Пытается догнать и

				съесть ближайших олений , восполнив тем самым голод. Не может выйти за пределы дикой местности .
	олень	Point	много	Двигается хаотично. Не может выйти за пределы дикой местности .
5	дикая местность	Polygon	1	Постоянно генерирует волков и олений в случайной точке себя.
	волк	Point	много	Пытается догнать и съесть ближайших олений . Не может выйти за пределы дикой местности . Может сбиваться с другими волками в стаю, в стае движется быстрее.
	олень	Point	много	Двигается хаотично. Не может выйти за пределы дикой местности .
6	аэропорт	Point	5	Создает самолеты , посылая их в другой случайный аэропорт .
	самолет	Point	много	Двигается до заданного аэропорта с заданной скоростью.
7	аэропорт	Point	5	Хранит количество своих самолетов . Посылает самолеты в другой случайный аэропорт , если самолеты у него есть.
	самолет	Point	много	Двигается до заданного аэропорта с заданной скоростью. При достижении цели, добавляется в аэропорт .
8	аэропорт	Point	5	Создает самолеты , посылая их в другой случайный аэропорт .
	самолет	Point	много	Двигается до заданного аэропорта с заданной скоростью. Должен избегать попадания в грозовую бурю .
	грозовая буря	Polygon	10	Создаются в начале работы модуля.
9	аэропорт	Point	5	Создает самолеты , посылая их в другой случайный аэропорт .
	самолет	Point	много	Двигается до заданного аэропорта с заданной скоростью. Должен избегать попадания в грозовую бурю .
	грозовая буря	Polygon	много	Появляются и исчезают в случайные моменты времени.
10	аэропорт	Point	5	Создает самолеты , посылая их в другой случайный аэропорт так, чтобы у них на пути не было грозовых бурь .

	самолет	Point	много	Двигается до заданного аэропорта с заданной скоростью.
	грозовая буря	Polygon	10	Создаются в начале работы модуля.
11	регион подводной лодки	Polygon	1	Постоянно генерирует объекты для исследования в в случайной точке себя .
	подводная лодка	Point	1	Двигается в сторону объектов для исследования , после достижения цели, объект для исследования исчезает.
	объект для исследования	Point	много	
12	регион подводной лодки	Polygon	1	Постоянно генерирует объекты для исследования в случайной точке себя.
	подводная лодка	Point	1	Двигается в сторону объектов для исследования , после достижения цели и прошествия времени, необходимого для исследования, объект для исследования исчезает.
	объект для исследования	Point	много	
13	регион подводной лодки	Polygon	1	Постоянно генерирует объекты для исследования в случайной точке себя.
	подводная лодка	Point	1	Двигается в сторону объектов для исследования , после достижения цели, объект для исследования исчезает.
	объект для исследования	Point	много	Может потерять актуальность и исчезнуть после прошествия определенного времени.
14	регион подводной лодки	Polygon	1	Постоянно генерирует объекты для исследования в случайной точке себя.
	подводная лодка	Point	1	Двигается в сторону объектов для исследования , после достижения цели, объект для исследования исчезает. Не может пересекать барьерные рифы .
	объект для исследования	Point	много	

	барьерный риф	Polygon	5	Создаются в начале работы модуля внутри региона подводной лодки .
15	регион подводной лодки	Polygon	1	Постоянно генерирует объекты для исследования в в случайных точках себя..
	подводная лодка	Point	1	Двигается в сторону объектов для исследования , после достижения цели, объект для исследования исчезает.
	объект для исследования	Polygon	много	(Следует обратить внимание, что в данном случае это полигон.)
16	порт	Point	5	Создает корабли , посылая их в другой случайный порт .
	корабль	Point	много	Двигается до заданного порта с заданной скоростью.
17	порт	Point	5	Хранит количество своих кораблей . Посылает корабли в другой случайный порт , если корабль у него есть.
	корабль	Point	много	Двигается до заданного порта с заданной скоростью. При достижении цели, добавляется в порт .
18	порт	Point	5	Создает корабли , посылая их в другой случайный порт .
	корабль	Point	много	Двигается до заданного порта с заданной скоростью. Должен избегать попадания в грозовую бурю .
	грозовая буря	Polygon	10	Создаются в начале работы модуля.
19	порт	Point	5	Создает корабли , посылая их в другой случайный порт .
	корабль	Point	много	Двигается до заданного порта с заданной скоростью. Должен избегать попадания в грозовую бурю .
	грозовая буря	Polygon	много	Появляются и исчезают в случайные моменты времени.
20	порт	Point	5	Создает корабли , посылая их в другой случайный порт так, чтобы у них на пути не было грозовых бурь .
	корабль	Point	много	Двигается до заданного порта с заданной скоростью.
	грозовая буря	Polygon	10	Создаются в начале работы модуля.
21	вокзал	Point	5	Создает поезда , посылая их в другой случайный вокзал .

	поезд	Point	много	Двигается до заданного порта с заданной скоростью.
22	вокзал	Point	5	Хранит количество своих поездов . Посылает поезда в другой случайный вокзал , если поезд у него есть.
	поезд	Point	много	Двигается до заданного вокзала с заданной скоростью. При достижении цели, добавляется в вокзал .
23	вокзал	Point	5	Создает поезд , посылая их в другой случайный вокзал .
	поезд	Point	много	Двигается до заданного вокзала с заданной скоростью. Должен дожидаться исчезновения преграды .
	преграда	Polygon	10	Создаются в начале работы модуля, исчезают через какое-то время.
24	вокзал	Point	5	Создает поезд , посылая их в другой случайный вокзал .
	поезд	Point	много	Двигается до заданного вокзала с заданной скоростью. Должен дожидаться исчезновения преграды .
	преграда	Polygon	много	Появляются и исчезают в случайные моменты времени.
25	вокзал	Point	5	Создает поезда , посылая их в другой случайный вокзал так, чтобы у них на пути не было преград .
	поезд	Point	много	Двигается до заданного вокзала с заданной скоростью.
	преграда	Polygon	10	Создаются в начале работы модуля.
26	комбайн	Polygon	1	Собирает урожай с поля .
	поле	Polygon	1	Полигон, на котором изначально есть урожай. Урожай может быть собран (для этого в этом и подобных задачах можно использовать <code>ILinearRing[] Holes</code> , свойство полигона, отвечающее за “дыры” в нем)
27	комбайн	Polygon	2	Собирает урожай с ближайшего поля .
	поле	Polygon	5	Полигон, на котором изначально есть урожай. Урожай может быть собран.
28	комбайн	Polygon	1	Собирает урожай с поля .
	поле	Polygon	1	Полигон, на котором изначально есть урожай. Урожай может быть собран. Урожай снова вырастает через какое-то время.
29	комбайн	Polygon	1	Собирает урожай с поля . Есть максимальная вместимость. При достижении максимальной

				вместимости должен отвезти урожай на склад и снова продолжить работу.
	поле	Polygon	1	Полигон, на котором изначально есть урожай. Урожай может быть собран.
	склад	Point	1	
30	комбайн	Polygon	1	Собирает урожай с поля , не может пересекать преграды .
	поле	Polygon	1	Полигон, на котором изначально есть урожай.
	преграда	Polygon	5	

РЕКОМЕНДУЕМАЯ ЛИТЕРАТУРА ПО КУРСУ

1. Системы поддержки принятия решений: учебник и практикум для бакалавриата и магистратуры / под ред. В. Г. Халина, Г. В. Черновой. – М. : Издательство Юрайт, 2016. – 494 с.

2. *Simon, H. A. Rationality as Process and as Product of Thought / H. A. Simon // American Economic Review. – 1978. – Vol. 68. – No. 2. – P. 1–16.*

3. *Петровский, А. Б. Теория принятия решений / А. Б. Петровский. – М. : Академия, 2009. – 400 с.*

