СТАТЬИ

УДК 004.42

МЕТОД КОНТРОЛЯ УРОВНЕЙ ДЕТАЛИЗАЦИИ В 3D СЦЕНАХ НА ОСНОВЕ АЛГОРИТМА ПОИСКА МАРШРУТА

Алпатов А.Н., Баранецкий Д.А., Синьбухов Д.С.

ФГБОУ ВО «МИРЭА – Российский технологический университет», Москва, e-mail: aleksej01-91@mail.ru

В современном мире средства виртуального изучения городских пространств набирают большую популярность. Для их создания всё чаще применяют технологии виртуальной или дополненной реальности. Существующие решения помогают создавать виртуальные города на основе грубых примитивов с довольно высокой скоростью, однако детализация таких пространств не соответствует требованиям ряда задач, для которых может решаться: виртуальные экскурсии и реалистичные копии зданий, архитектурных сооружений и памятников требуют проработанных моделей объектов. При этом хранение и одновременная обработка больших массивов данных, особенно в системах виртуальной реальности, – ресурсоёмкая задача, которая может решаться путём настройки уровней детализации. В данной работе исследуется возможность и метод контроля уровней детализации в сценах 3D с проложенным маршрутом с применением геоинформационных данных GIS-систем. В статье рассмотрен алгоритм получения необходимых геоинформационных данных из открытых источников, предложен алгоритм поиска маршрута с учётом критериев, а также алгоритм построения графа позиционирования объектов. С точки зрения уровней детализации рассматривается применимость геоинформационных данных для определения уровней детализации для различных секторов относительно проложенного маршрута.

Ключевые слова: уровни детализации, LOD, HLOD, игровые визуализаторы графики, игровые движки, маршрутизация, 3D, алгоритм поиска пути

A METHOD FOR CONTROLLING LEVELS OF DETAIL IN 3D SCENES BASED ON A ROUTE-FINDING ALGORITHM

Alpatov A.N., Baranetskiy D.A., Sinbukhov D.S.

MIREA – Russian Technological University, Moscow, e-mail: aleksej01-91@mail.ru

In today's world, means of virtual study of urban spaces are gaining in popularity. More and more often virtual or augmented reality technologies are used to create them. The existing solutions help to create virtual cities based on rough primitives at a fairly high speed, but the detailing of such spaces does not meet the requirements of a number of tasks for which it can be solved: virtual tours and realistic copies of buildings, architectural structures and monuments require elaborate models of objects. At the same time, the storage and simultaneous processing of large amounts of data, especially in VR, is a resource-intensive task, which can be solved by adjusting the levels of detail. This paper explores the possibility and method of controlling the level of detail in 3D scenes with a laid route using GIS-system geo-information data.In this paper we are going to obtain necessary geo-information data from open sources, propose an algorithm for finding a route taking into account criteria and an algorithm for constructing a graph for positioning objects. In terms of levels of detail the applicability of geographic information data to determine the levels of detail for different sectors in relation to the laid route is considered.

Keywords: levels of detail, LOD, HLOD, game graphics renderers, game engines, routing, 3D, pathfinding algorithm

В мире всё чаще возникают ситуации, при которых физическое посещение того или иного места становится проблематичным или вовсе невозможным. Это послужило более быстрому развитию и распространению альтернативных способов изучения и взаимодействия с окружающим миром. Большинство из них представляют собой фотографии или видеоролики, снятые с помощью панорамных камер. При таком подходе пользователи получают максимально фотореалистичную картинку, но при этом они остаются скованными материалом, предоставленным «режиссёром» сцены. Степени свободы пользователя ограничены минимальным функционалом интерактивного взаимодействия.

Параллельно с этим наблюдается существенный прогресс в сфере компьютерного 3D моделирования, средствах игровых визуализаторов графики (игровых движков) и технологиях виртуальной реальности.

В данной статье будет предложен и рассмотрен возможный метод использования данных из открытых GIS-систем для поиска маршрута внутри городских пространств в виртуальной реальности и их дальнейшего использования при настройке уровней детализации объектов на сцене.

Разрабатываемый метод может быть полезен при изучении реально существующих городских пространств, для изучения давно утраченных архитектурных сооружений, а также при создании совершенно новых объектов.

Обзор методов визуализации геоинформационных данных в компьютерной 3D графике

Проводимые в данной области исследования отмечали, что принципы работы игровых визуализаторов и геоинформационных систем очень сильно отличаются [1].

Детальное отображение объектов не является целью большинства GIS-систем. Во-первых, GIS-системы по своей концепции предоставляют текстовую и географическую информацию об объектах, в связи с этим они не требуют серьёзных визуальных мощностей. Во-вторых, так как данная составляющая не требует постоянных улучшений, то фундаментальные основы GIS систем меняются медленнее, чем у игровых движков.

В приведённой работе [1] автором были выделены основные проблемы, возникающие при переориентации графических визуализаторов к работе с данными GIS-систем:

1. Различие в используемых системах координат.

2. Использование различных видов и типов текстур. Так, в игровых движках в большинстве случаев используется подход, основанный на замещении элементов компьютерной 3D сцены повторяющимися текстурами, часто сгенерированных программно, в то же время в ГИС такой подход не распространен – используются реальные фотоснимки поверхности Земли, ландшафта и т.д. В то же время стоит добавить, что тенденции последних лет в области игровых движков показывают стремление разработчиков к фотореализму в играх, широкому распространению виртуальной реальности, поэтому в современных компьютерных играх могут использоваться и снимки реальных объектов.

3. Минимизация полигонов в сетке и использование высококачественных текстур в игровых движках с целью повышения их производительности. В то время как в ГИС количество полигонов обычно больше, что положительно влияет на точность используемых моделей, что важно для практического применения.

4. Использование точных механизмов позиционирования в ГИС-системах, что может не наблюдаться в игровых движках, так как такая точность зачастую не требуется в компьютерных играх.

Общеизвестна информация о разработках стартапа Blackshark.ai. Силами команды в 50 сотрудников была реализована фотореалистичная компьютерная сцена, на основе технологии искусственных нейронных сетей и высоких вычислительных мощностей облачных ресурсов [2]. Blackshark – ответвление игровой студии Bongfish, делающее упор на работу с искусственным интеллектом. В данном проекте использованы техники машинного обучения для реализации программной системы, обладающей свойствами обучения для построения игровых сцен и карт с высокой степенью детализации, что впоследствии было использовано в рамках совместного проекта с компанией Microsoft.

Существующие решения в совмещении GIS-систем и игровых движков

Такие методы используются в ряде решений: Vitruvio, TerraformPro, CityEngine VR Experience и подобных. Приведённые плагины используются в Unreal Engine 4 для упрощения разработки городских пространств в 3D сцене и решают задачи совмещения GIS-данных и игровых визуализаторов графики.

Однако графические возможности этих плагинов ограничены. Они предоставляют довольно примитивные грубые шаблоны 3D объектов. Подобные системы используются для воссоздания среднего района города игрового VR проекта. В случае работы с более серьёзными проектами, нацеленными на большую степень географической достоверности, требуются высокодетализированные модели, что делает плагины лишь вспомогательными средствами.

На основе проведённых исследований можно сделать вывод, что построение высокодетализированного городского пространства задача всё ещё актуальная. Существующие решения помогают воссоздавать на основе данных из GIS-системы ландшафты и «болванки» городов с довольно высокой скоростью, однако детализация созданных пространств не соответствует требованиям ряда задач, для которых может решаться: виртуальные экскурсии и реалистичные копии зданий, архитектурных сооружений и памятников требуют проработанные модели объектов. При этом хранение и одновременная обработка таких массивов данных, особенно в VR – ресурсоёмкая задача, которая может решаться путём настройки уровней детализации. В данной работе предлагается взглянуть на задачу контроля детализации с помощью данных GIS-систем.

Анализ моделей данных для построения графа позиционирования географических объектов для 3D графики

Существует обширный спектр GIS-средств для изучения географических пространств окружающего мира. Google Maps, Yandex Maps, 2GIS и прочие системы сейчас используются повсеместно и уже перестают быть просто средствами для навигации [3]. Поэтому, чтобы предоставлять весь имеющийся функционал для их работы, необходимо иметь достаточно полные и актуальные данные о местности, а также средства хранения и обработки информации.

Есть два возможных варианта хранения данных в подобных системах [4]:

− растровый – представление модели в виде совокупности ячеек и информации о них;

− векторный – цифровое представление точечных, линейных и полигональных объектов в виде наборов координатных пар.

Так как при работе с картами приходится взаимодействовать с объектами, лежащими на плоскости в топологическом пространстве, то для данной области более рациональным решением будет использование векторных моделей.

Для получения данных сейчас используется один из основных подходов:

1) готовое API для получения данных от стороннего сервиса;

2) непосредственное взаимодействие с набором данных и получение необходимой информации на основе их обработки и анализа.

Первый способ намного быстрее и проще, если имеется доступ к подобному сервису, предоставляющему весь необходимый функционал. Однако для построения системы электронных экскурсий лучше делать свою обработку.

Наиболее простым способом получения картографических данных является использование сервиса OpenStreetMap (OSM) [5] – картографический проект, поддерживаемый и развиваемый сообществом энтузиастов, на основе личного вклада, каждого зарегистрированного пользователя и направленный на создание свободно распространяемой карты мира.

Подобный способ распространения несет ряд сложностей, в виде отсутствия строгого регламента организации объектов или возможных неточностей из-за политики открытости. Однако это также позволяет поддерживать эффективное обновление данных и их актуальность.

Данные в нем представлены в XML-формате, где самый минимальный элемент – «Node» или «точка». Точка имеет географические координаты и ряд тегов, поясняющих, чем она является. Каждая точка может быть объединена связью и являться частью полигона или линии, которые также будут являться объектами, с соответствующим перечнем тегов с описанием.

Существует несколько способов получения данных о структуре города, но все они сводятся к получению и парсингу (обработке) OSM файла. Следовательно, после получения файла с информацией по всем объектам необходимо распарсить (разобрать) данные и получить массивы дорог и зданий. Парсинг данных можно произвести либо до, либо после того, как выгружать данные с интересующим регионом из интернет-ресурса. Последний способ намного удобнее, так как избавляет от необходимости написания фильтра объектов и предоставляет на выходе документ, содержащий только искомые объекты.

В OSM данные хранятся структурированно, но самое минимальное приближение, которое удастся получить – это штат, область или район. Вне зависимости от региона мы получаем довольно большой набор данных, что не избавит нас от необходимости фильтрации ненужных данных.

Существуют сторонние сервисы, работающие с OSM: osm2pgsql, OSMConvert или YourMaps. Для данной работы самым удобным среди них является – yourmaps.io [6]. Он предоставляет достаточно полный перечень инструментов для взаимодействия и не требует дополнительных затрат на скачивание или обучение.

YourMaps предоставляет простой и понятный интерфейс для взаимодействия с пользователем, продемонстрированный на рис. 1. С его помощью можно создавать запросы посредством визуального программирования и на выходе получить geojson. По сути – это тот же json, но с поддержкой отображения данных на сайте после загрузки.

Рис. 1. Выбор обрабатываемой зоны

Данный ресурс позволит уменьшить объем обрабатываемых данных до минимума и сразу приступить к строительству графа дорог.

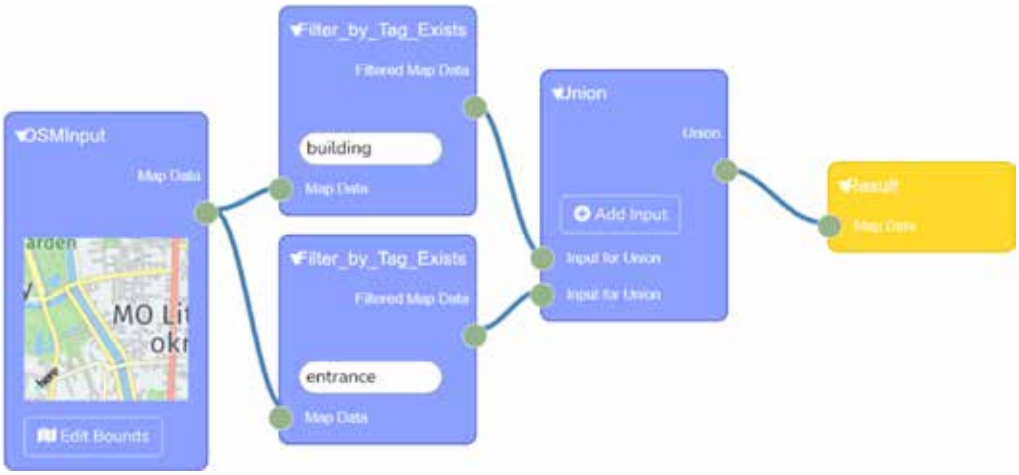

Рис. 2. Запрос зданий и их входов

В качестве координат в geojson используются широта и долгота, которые далее можно преобразовать в удобный формат с помощью математических преобразований. Узлами графа [6] может быть место стыка дорог или точка входа в интерактивную зону (музей или храм), как показано на рис. 2. Так как данные по сути своей хранятся в векторном виде, то для оценки веса можно использовать квадратичное расстояние от точки до точки.

Формат представления структуры геоинформационных данных для формирования датасета

OSM и geojson представляют одну и ту же информацию, но есть несколько существенных отличий:

− в OSM хранится техническая информация о создании записи;

− в OSM вместо массива координат используется список (или массив) ссылок на ноды, которые в свою очередь содержат координаты.

При парсинге файла и составлении графа потребуется произвести дополнительные действия, чтобы определить расположение ключевых точек объекта. Сначала необходимо найти их по индексу, а после этого уже считать необходимую информацию. Так как geojson не использует подобной косвенной адресации, то может показаться более перспективным решением. Однако, если требуется достичь точного соотношения связей объектов, то подобная организация играет на руку. Это позволяет соединить два объекта, к примеру дороги и здания, посредством ссылки на общую точку соприкосновения. Обычно такой точкой является точка выхода с тегом «*entrance*». Таким образом, в geojson время для определения позиций объекта будет константным, в то время как в OSM придется просмотреть ряд точек для поиска необходимых данных, что бесполезно в случае пренебрежения точностью.

Данный метод разрабатывается для дальнейшего использования в приложениях виртуальной реальности. Поэтому более удобным решением будет выбор geojson в качестве представления геоинформационных данных из GIS-систем. По сути, он является объектом JSON, парсеры которого присутствуют в любом языке программирования, чего не скажешь о OSM.

Составление графа позиционирования объектов на основе геоинформационных данных

Ориентирование на платформы виртуальной реальности позволяет решить несколько важных проблем и пренебречь определёнными факторами в пользу упрощения задачи. Предлагается не учитывать реальное положение входов и выходов зданий, а использовать для обозначения одну из ближайших к центру дороги точек. В силу чего становится достаточным использование JSONa.

При построении графа ключевыми факторами являются:

− позиция точки на карте;

− индекс вершины в графе;

− вес вершины.

Полученные координаты хранятся в виде широты и долготы, то есть являются сферическими и используют 6 знаков после запятой для достижения минимального порога точности. Для удобства возможно выполнение преобразования их координат в декартову систему с центром в левом нижнем углу захваченной области, что по-

зволит иметь дело только с положительными координатами.

Во-первых, необходимо перейти от GEOJSON к JSON формату, поменяв расширения. Далее можно воспользоваться любой доступной библиотекой для парсинга JSON файла. Так как граф создается единожды, допустимо пренебречь скоростью построения, в угоду удобству.

Итоговое количество вершин неизвестно, поэтому стоит прибегнуть к использованию матрицы смежности. Допустимым вариантом будет список дуг или список смежности (принцип хранения особенной важности не имеет).

Ключевым элементом является способ соотношения координат с вершиной графа, хранящейся в структуре. Можно хранить пары (ключ-значение) из координат и индекса вершины соответственно. Это даст список индексированных вершин. Алгоритм построения графа будет следующим:

1. Если не конец, то взять новый объект-дорогу.
2. Взять массив его координат.
3. Если не конец, то взять следующую пару координат.
4. Преобразовать координаты в удобную систему.
5. Проверить, есть ли в списке точка с такими координатами.
6. Если нет, то внести ее, иначе взять индекс имеющейся.
7. Если в памяти есть предыдущая вершина, то пометить вместе с текущей, они образуют дугу и ее нужно внести в структуру графа (список дуг) и пометить как начало следующей, иначе просто помечаем как начало следующей.
8. Переход в пункт 4.

Когда все объекты дорог просмотрены, а их вершины внесены в граф, то можно считать, что граф переходов построен. Далее необходимо присвоить веса всем вершинам, имеющим значимость, и отсеять вершины, которые не играют особой роли – имеющие не более двух связей и нулевой вес.

Составление маршрутов для конкретизации приоритетной области

При составлении маршрутов может возникнуть одна из двух возможных ситуаций:

− Пользователь знает конкретный интересующий объект;

− Пользователь знает несколько интересующих объектов.

Самыми эффективными из них являются метод имитации отжига, метод ветвей и границ. Однако если количество точек невелико и конечно, то можно воспользоваться первым, так как он гарантированно найдет самый оптимальный (по заданному критерию) маршрут, пускай и в ущерб времени. Для измерения веса соединяющих дуг можно взять евклидово расстояние между точками или измерять вес в соответствии с культурной значимостью района. После определения с весовой оценкой можно определить порядок обхода точек и перейти к вычислению маршрутов для каждой точки поочередно и последующему их объединению в один.

Сама по себе задача поиска маршрута является тривиальной. Существует множество различных способов построить маршрут из точки А в точку В. Сложностью является размер обрабатываемой информации. В наше время даже средние города могут состоять из сотен тысяч узлов, необходимых для достижения точности маршрута, что может оказывать серьезное воздействие на производительность системы. Для обхода этой проблемы используются различные алгоритмы оптимизации поиска или самого графа.

Наибольшую популярность в поисковых системах получили дополненная реальность, а также 3D визуализация среды, в которой строится маршрут (трехмерная модель города с маршрутом и воссозданными зданиями). Для последней ранее использовались объекты-болванки, более или менее повторяющие структуру здания, но с ростом технологий увеличивается и требование к ним. Сегодня уже есть технологии, позволяющие воссоздавать макеты зданий не в виде серых коробок, а с накладыванием на них примерной текстуры, взятой по средствам спутниковой фотографии, а также путём полного или частичного объектного воспроизведения объекта.

В случае системы виртуальных экскурсий необходимо пойти еще дальше и повысить качество визуализации ключевых городских объектов. Таким образом количество загружаемой и обрабатываемой информации увеличивается. Встает вопрос об оптимизации процессов обработки информации.

Получается, в работе помимо проблемы оптимизации маршрута поиска присутствует проблема оптимизации отображения 3D симуляции.

Конкретизация приоритетной области 3D сцены для управления уровнями детализации

Современные решения по реализации уровней детализации можно подразделить на 3 типа: статичные (дискретные)

уровни детализации (Discrete Levels of Detail – DLOD), непрерывные (динамические) уровни детализации (Continuous Levels of Detail – CLOD) и иерархические уровни детализации (Hierarchical Levels of Detail – HLOD). Каждый из данных методов имеет свои плюсы и минусы и используется в соответствии с необходимыми решениями [7]. Также известно комбинированное использование данных методов [8].

Для решения задач с отображением большого числа статических объектов лучше всего подходит применение HLOD [9, 10]. Данный метод зарекомендовал себя в последнее время, так как он применяет упрощённые представления не поэлементно, а для организованных в многоуровневые иерархии групп (англ. LOD batching). При обходе вглубь иерархии уровней детализации программа обрабатывает пакеты объектов, не осуществляя выбора уровня детализации для каждого из них по отдельности [10]. Такой подход позволяет достичь более высокого уровня упрощения, сократить общее время обхода дерева 3D сцены, а также уменьшить общее количество вызовов отрисовки (drawcalls), что положительно сказывается на общей производительности в условиях высокодетализированных объектов, так как для отрисовки высокодетализированного игрового объекта на сцене графическим API выполняется значительная работа для каждого вызова отрисовки, что при их большом количестве приводит к лишним накладным расходам на вычислительный процессор [10].

Разрабатываемый метод базируется на том факторе, что после того, как нам становится известен маршрут, мы получаем достаточный набор данных для того, чтобы определить уровни детализации для объектов в каждом секторе. Сектор включает в себя группы объектов, объединённые в кластеры.

Такой подход позволит уже на этапе построения маршрута определить уровни детализации для большинства объектов на карте. Также это позволит отказаться от перерисовки объектов из низкоприоритетных секторов во время прохождения маршрута, что позволит снизить затрачиваемые ресурсы [10].

После того, как был проложен маршрут через заданные пользователем точки, программа разбивает обрабатываемый район на 2^n секторов. Значение n вычисляется по следующей формуле:

$$n = \zeta / \kappa , \qquad (*)$$

где ζ – длина стороны рассматриваемого района; κ – длина стороны сектора.

Длины выбираются таким образом, чтобы полученная секторная сеть не была очень мелкой и каждый сектор имел в себе несколько объектов для дальнейшего объединения в кластер. При этом число n обязательно должно быть целым (в ином случае проводится округление в большую сторону). По результатам вычислений мы получаем секторную сеть района.

Относительно проложенного маршрута сектора могут делиться на следующие типы, как показано на рис. 3:

− c-way-сектор – сектор, внутри которого находится хотя бы один узел проложенного маршрута (розовый сектор);

− a-way-сектор – сектор, внутри которого нет ни одного узла проложенного маршрута, но при изменении пользователем маршрута следования данный сектор на 3D сцене должен быть отрисован максимально быстро (синий сектор);

− s-way-сектор – сектор, внутри которого нет ни одного узла проложенного маршрута (бирюзовый сектор).

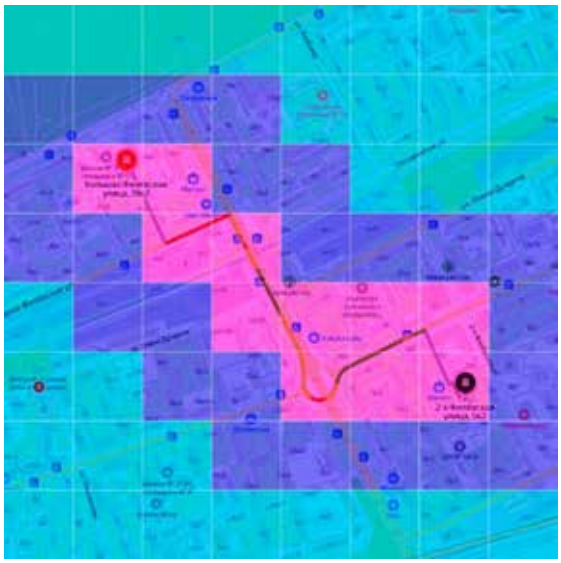

Рис. 3. Выделение секторов относительно проложенного маршрута с сеткой 8 x 8

Относительно текущего положения пользователя [11] сектора делятся на следующие (рис. 4):

− c-player-сектор – сектор, в узле которого находится пользователь (зелёный сектор);

− a-player-сектор – сектор, смежный с текущим сектором пользователя (жёлтый сектор);

− s-player-сектор – сектор, не являющийся смежным с текущим сектором пользователя (красный сектор).

Данных условий может быть недостаточно. Стоит учитывать, что в каждом

из секторов находится 1 и более узлов графа позиционирования геоинформационных объектов. Узел может иметь или не иметь связей с соседними узлами, в зависимости от сложности графа. Поэтому возможна ситуация, при которой сектор А, являющийся смежным с сектором В, не имеет в себе ни одного такого узла, который был бы связан хотя бы с одним узлом сектора В (например – сектора разделены рекой, оврагом или железнодорожными путями). Тогда присвоение высокого уровня детализации для данного сектора не является целесообразным.

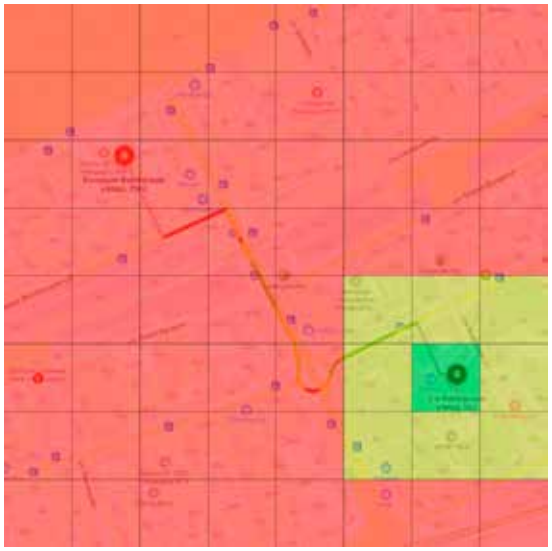

Рис. 4. Выделение секторов относительно положения пользователя

Если существует возможность объединения секторов на низком уровне, то мы рассматриваем секторные сети на 2^{n-k} секторов, где k – целое число больше 0. На рис. 3 можно отметить, что в правом верхнем углу, а также в левом нижнем присутствует скопление секторов с одинаковыми уровнями детализации. Перейдя с секторной сетки 2^3 на сетку 2^2, как показано на рис. 5, можно заметить, что сектора в правом верхнем углу объединились в один сектор, в соответствии с этим объекты, расположенные в нём, могут быть собраны в новый, более крупный, кластер и обрабатываться уже в таком виде. Также на этой сетке повысились уровни детализации секторов, которые раньше имели более низкую детализацию.

Для нахождения баланса между двумя сетками предлагается провести их наложение друг на друга и определить более точные уровни детализации.

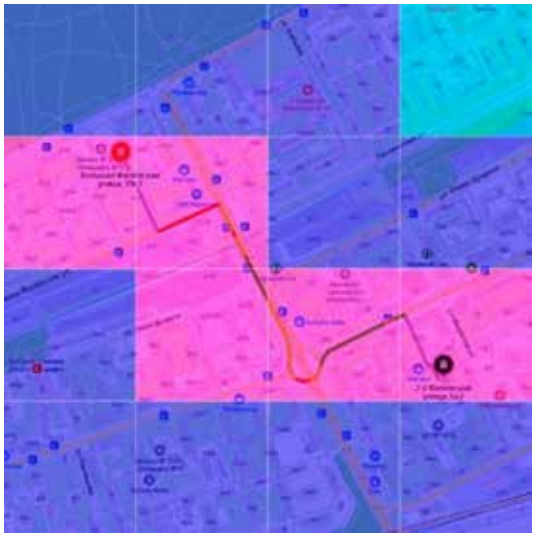

Рис. 5. Выделение секторов относительно проложенного маршрута на сетке 8 х 8

Представим сетку «4 х 4» в виде матрицы. Для данной сетки веса сектора определяются следующим образом:

– если он относится к c-way-сектору, то он получает оценку «2»;

– если он относится к a-way-сектору, то он получает оценку «1»;

– если он относится к s-way-сектору, то он получает оценку «0».

По итогу получается квадратная матрица:

$$\mathrm{Sec} = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 2 & 2 & 1 & 1 \\ 1 & 2 & 2 & 2 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

Аналогичным образом представим сетку «8 х 8», представленную на рис. 3. Получаем квадратную матрицу:

$$\mathrm{Sec} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 2 & 2 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 2 & 2 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 2 & 2 & 2 & 2 & 1 \\ 0 & 0 & 1 & 1 & 2 & 2 & 2 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Проведём сложение матриц, чтобы более детально определить уровни детализации для секторов.

Чтобы провести сложение матриц приведём матрицу «4 x 4» к виду «8 x 8». Для этого мы сначала дублируем каждый столбец на 1 столбец вправо, а далее полученные строки дублируем на 1 строку вниз. Проводим расчёты:

$$\text{Sec} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 2 & 2 & 2 & 2 & 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 & 1 & 1 & 1 & 1 \\ 1 & 1 & 2 & 2 & 2 & 2 & 2 & 2 \\ 1 & 1 & 2 & 2 & 2 & 2 & 2 & 2 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 2 & 2 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 2 & 2 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 2 & 2 & 2 & 2 & 1 \\ 0 & 0 & 1 & 1 & 2 & 2 & 2 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} =$$

$$= \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 2 & 2 & 2 & 1 & 1 & 1 & 0 & 0 \\ 3 & 4 & 4 & 3 & 2 & 1 & 1 & 1 \\ 3 & 3 & 4 & 4 & 2 & 2 & 2 & 2 \\ 1 & 2 & 3 & 4 & 4 & 4 & 4 & 3 \\ 1 & 1 & 3 & 3 & 4 & 4 & 4 & 3 \\ 1 & 1 & 1 & 2 & 2 & 2 & 2 & 2 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

Результатом проведённых расчётов является новая матрица, относительно которой мы можем более точно определить, какие уровни детализации использовать для секторов, как показано на рис. 6.

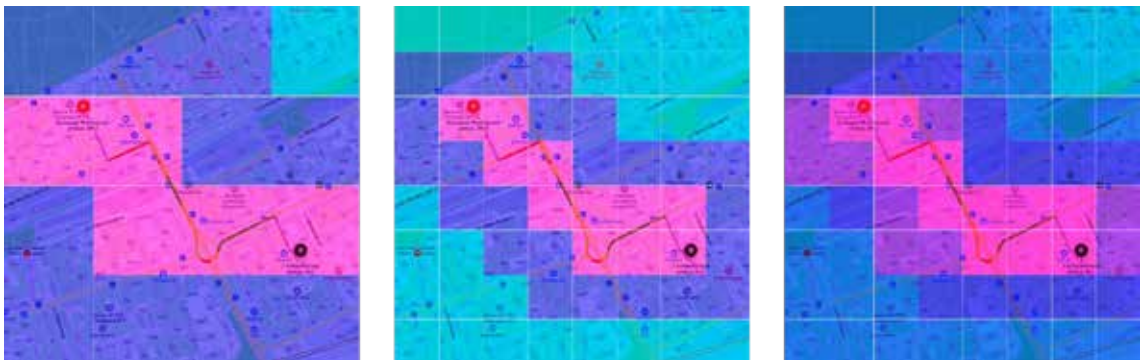

Рис. 6. Результирующее распределение уровней детализации

Заключение

По результатам проведённой работы была изучена структура формата хранения геоинформационных данных в GIS-системах, на основе которой был предложен алгоритм построения графа позиционирования объектов внутри трёхмерной сцены. Также предложен метод контроля уровней детализации на основе конкретизации приоритетной области с точки зрения проложенного маршрута и текущего положения пользователя при помощи дискретизации рассматриваемой области. Такой подход, по мнению авторов, позволит снизить лишние накладные расходы при рендеринге сцен, в рамках реализации компьютерных игр с фотографическими высокодетализи-

рованными сценами, используя для этого пространственное положение игрового объекта и камеры от третьего лица. В результате данного этапа исследований был предложен метод взаимодействия GIS-системы и игрового движка для построения наиболее достоверных виртуальных городских пространств. Данный подход может использоваться в системах виртуальной реальности: симуляторах, образовательных программах и продвинутых GIS-системах.

Список литературы

1. Жигалов К.Ю. Использование игровых визуализаторов графики в современных геоинформационных системах // Cloud of Science. 2016. № 3. С. 71–80.

2. Milojevic-Dupont N., Hans N., Kaack L.H., Zumwald M., Andrieux F., de Barros Soares D., Lohrey S., Pichler P., Creutzig F. Learning from urban form to predict building heights. Plos one. 2020. Vol. 15. No.12. DOI: 10.1371/journal.pone.0242010.

3. Голубев К.В., Загарских А.С. Обзор проблем разработки мультимасштабного виртуального глобуса и существующих методов их решения // Альманах научных работ молодых ученых Университета ИТМО. 2017. С. 47–50.

4. Ананьев Ю.С. Геоинформационные системы. Томск: ТПУ, 2003. 70 с.

5. Mooney P., Minghini M. A review of OpenStreetMap data. Mapping and the Citizen Sensor. London: Ubiquity Press, 2017. 400 p.

6. Zhu Z., Tan J. A multi-source heterogeneous vector space data integration scheme based on geojson. 26th International Conference on Geoinformatics. IEEE. 2018. Vol. 1. P.1–4.

7. Mooney P., Corcoran P. The annotation process in OpenStreetMap. Transactions in GIS. 2012. Vol. 16. No. 4. P. 561–579.

8. Luebke D., Reddy M., Cohen J.D., Varshney A., Watson B., Huebner R. Level of detail for 3D graphics. San Francisco: Morgan Kaufmann, 2003. 390 p.

9. Zach C., Mantler S., Karner K. Time-critical rendering of discrete and continuous levels of detail. Proceedings of the ACM symposium on Virtual reality software and technology. 2002. Vol. 1. P.1–8.

10. Семёнов В.А., Шуткин В.Н., Золотов В.А., Морозов С.В. Расширение метода иерархических уровней детализации для динамических сцен с детерминированным характером событий // Труды Международной конференции по компьютерной графике и зрению «Графикон». 2019. № 29. С. 37–41.

11. Кудряшов А.П., Меженин А.В. Методы оптимизации сцен в игровых компьютерных приложениях // Альманах научных работ молодых ученых университета ИТМО. 2019. № 1. С. 139–143.