УДК 004.622; 004.021
DOI: 10.26102/2310-6018/2025.51.4.047

Адаптивный алгоритм геопоиска для движущихся объектов на основе R-деревьев

В.А. Дорохин✉

Государственный университет «Дубна», Дубна, Российская Федерация

Резюме. В статье предложен адаптивный алгоритм геопоиска на основе R*-деревьев, ориентированный на сценарии реального времени с участием динамически перемещающихся пользователей. Традиционные методы, основанные на фиксированной точке и круговой или квадратной зоне поиска, не учитывают кинематические параметры пользователя – скорость и направление движения, – что приводит к низкой релевантности результатов, избыточной передаче данных и повышенной нагрузке на клиентские устройства и сеть. Для решения этой проблемы разработан модифицированный подход, формирующий асимметричную поисковую область, расширенную вдоль вектора движения пользователя. Размер и форма этой области динамически регулируются с учетом текущей скорости и настраиваемого коэффициента ее влияния. Дополнительно вводится параметр «зона видимости» целевых объектов, позволяющий индексировать не только их координаты, но и радиусы потенциального отображения, что повышает точность фильтрации. Алгоритм реализован для решения задачи маршрутизации объектов метапространства и использует R*-дерево с оптимизированной массовой загрузкой данных для минимизации перекрытий минимальных ограничивающих прямоугольников (MBR). Результаты имитационного моделирования показали, что предложенный метод повышает релевантность выдачи до 7 раз по сравнению с классическим радиальным поиском, при этом не увеличивая вычислительную сложность операции поиска и снижая объем передаваемых данных. Подход применим в задачах навигации, дополненной и смешанной реальности, беспилотных систем и других областях, где критична актуальность пространственной информации в реальном времени.

Ключевые слова: геопоиск, R-дерево, пространственное индексирование, алгоритмы реального времени, релевантность поиска, метапространство.

Для цитирования: Дорохин В.А. Адаптивный алгоритм геопоиска для движущихся объектов на основе R-деревьев. *Моделирование, оптимизация и информационные технологии.* 2025;13(4). URL: 2025;13(4). URL: https://moitvivt.ru/ru/journal/pdf?id=2090 DOI: 10.26102/2310-6018/2025.51.4.047

An adaptive R-Tree-based geosearch algorithm for moving users

V.A. Dorokhin✉

Dubna State University, Dubna, the Russian Federation

Abstract. This paper presents an adaptive R*-tree-based geospatial search algorithm tailored for real-time applications involving dynamically moving users. Conventional methods that rely on a fixed point and a circular or square search area neglect user kinematics – specifically speed and heading – resulting in low result relevance, excessive data transmission, and increased load on client devices and network infrastructure. To address this, we propose a modified approach that dynamically shapes an asymmetric search region extended along the user's motion vector. The size and geometry of this region are adjusted in real time based on current speed and a configurable speed-influence coefficient. Additionally, a "visibility zone" parameter is introduced for target objects, enabling indexing not only of their coordinates but also of their potential display radii, thereby improving filtering accuracy. The algorithm is implemented within a metaspace object routing task and employs an R*-tree with optimized bulk-loading to minimize overlaps of Minimum Bounding Rectangles (MBR). Simulation results demonstrate improvement in search relevance up to 7 times compared to classical radial search, without increasing computational complexity and while reducing

network payload. The method is applicable to navigation, augmented/mixed reality, autonomous systems, and other domains requiring real-time spatial relevance.

Keywords: geospatial search, R-tree, spatial indexing, real-time algorithms, search relevance, metaspace.

For citation: Dorokhin.V.A. An adaptive R-Tree-based geosearch algorithm for moving users. *Modeling, Optimization and Information Technology.* 2025;13(4). (In Russ.). URL: 2025;13(4). URL: https://moitvivt.ru/ru/journal/pdf?id=2090 DOI: 10.26102/2310-6018/2025.51.4.047

Введение

Классические алгоритмы геопоиска, направленные на получение набора объектов, получают на вход точку и радиус поиска либо поисковый квадрат.

Тем не менее, существует большое количество задач, в которых пользователь динамичен и требуется в режиме реального времени обновлять результат поиска. Это задачи поиска POI [1], получения объектов карты в навигаторе движущегося автомобиля, маршрутизации запросов глобального метапространства дополненной и смешанной реальности [2], навигации для беспилотных систем и другие.

Как правило, задача решается двумя основными методами – постоянное обновление списка объектов в заданном радиусе от пользователя [3] либо пакетное обновление всей базы данных объектов заданного региона и загрузка следующего пакета при приближении к границе [4].

В случае, если система обладает маршрутом движения пользователя, может быть загружена только релевантная маршруту информация [5]. Этот подход эффективен, но не может использоваться в условиях неопределенности. Существует ряд методов повышения релевантности получаемых POI за счет отслеживания трафика, ретроспективы поведения пользователя и т. д. [6].

Данные методы увеличивают сложность поиска и требуют большего количества входных параметров для эффективной работы. Предлагается модифицировать классический алгоритм поиска на основе пространственного индексирования путем добавления параметров скорости и направления движения пользователя, что позволит повысить релевантность возвращаемых объектов.

Материалы и методы

Метод построения пространственного запроса на основе координат, скорости и направления клиента. Предлагается строить алгоритм поиска на основе пространственного индексирования. Поскольку искомые объекты являются потенциально видимыми объектами, предлагается ввести параметр «дальность обнаружения». Это позволит улучшить результаты путем исключения из поиска малозаметных объектов на периферии и предварительного получения объектов, видимых издалека.

Среди структур данных, используемых в области пространственного индексирования, популярны иерархические структуры данных, такие как KD-дерево, Quad-дерево и т. д. В случае работы в условиях малых размерностей данные структуры показывают высокую производительность и низкую сложность поиска. Среди них стоит особо отметить семейство R-деревьев[1].

R-дерево [7] является обобщением B+-дерева[2] в многомерном случае посредством использования N-мерных прямоугольников для ограничения объектов в

[1] Papadopoulos A.N., Corral A., Nanopoulos A., Theodoridis Ya. R-Tree (And Family). In: *Encyclopedia of Database Systems.* New York: Springer; 2009. P. 2453–2459. https://doi.org/10.1007/978-0-387-39940-9_300

[2] Liu L., Özsu M.T. *Encyclopedia of Database Systems.* New York: Springer; 2009. 4355 p. https://doi.org/10.1007/978-0-387-39940-9

поддереве. Алгоритм используется во многих СУБД индустриального уровня, таких как Oracle, PostgreSQL, SQLite, MySQL и т. д.

Одним из ключевых моментов, влияющих на производительность R-дерева, является алгоритм, решающий задачу расщепления переполненной вершины при вставке новой записи. Это решение определяет распределение записей по его вершинам. Существует ряд алгоритмов расщепления вершин. Помимо этого, существуют различные вариации структуры данных: R*-дерево [8], Revised R*-дерево (RR*-дерево) [9], Гильбертово R-дерево [10].

Для реализации релевантного поиска предлагается использовать за основу алгоритм R*-дерева и алгоритм массовой загрузки данных в R-дерево сверху вниз с минимизацией перекрытий [11]. Данный подход позволит оптимизировать процесс поиска за счет предварительной подготовки данных на этапе создания дерева. Поиск осуществляется по паре координат R^2, что позволяет изменять зону поиска, а также учитывать зону видимости целевых объектов.

Таким образом, применив алгоритмы построения и поиска R*-дерева можно получить базовый метод поиска. Добавление указываемых при создании объектов зон видимости позволяет оптимизировать релевантность возвращаемых объектов за счет учета радиуса их отображения. Для добавления данного критерия в существующее дерево – вместо добавления узла с координатами точки создается пара координат, описывающих квадрат видимости объекта метапространства, а именно:

$$\left(\left(\varphi - \frac{R}{111}, \lambda - \frac{R}{111 \cdot \cos\left(\frac{\varphi \cdot \pi}{180}\right)}\right), \left(\varphi + \frac{R}{111}, \lambda + \frac{R}{111 \cdot \cos\left(\frac{\varphi \cdot \pi}{180}\right)}\right)\right),$$

где φ – широта, λ – долгота, R – радиус видимости объекта поиска.

Для повышения релевантности получаемых объектов предлагается оптимизация зоны поиска. Классические поисковые алгоритмы используют фиксированный радиус для получения искомых объектов. Ввиду наличия на клиентских устройствах возможности получения в реальном времени скорости и направления движения – становится возможно их использование в процессе формирования поискового квадрата.

Предлагаемый алгоритм поиска (Рисунок 1) сводится к 4 основным этапам: нормализация, формирование поискового квадрата, поиск, фильтрация.

На вход алгоритм получает следующий набор параметров:

– ϕ_0, λ_0: исходные координаты (широта, долгота) в градусах;
– v: скорость движения (км/ч);
– θ: направление движения (градусы от севера);
– r: базовый радиус зоны (км);
– k: коэффициент влияния скорости (безразмерный).

Этап нормализации включает в себя получение абсолютного влияния скорости и перевода направления в радианы.

Абсолютное влияние скорости рассчитывается по формуле $s = v \cdot k$, где коэффициент влияния скорости представляется как параметр поиска, либо используется стандартное значение для выбранной задачи.

Направление поступает в систему как азимут относительно севера в градусах, для дальнейшего использования данное значение переводится в радианы $\theta_{rad} = (\theta \cdot \pi)/180$.

Для формирования области поиска составляются проекции смещения для каждой из координат:

$$\begin{cases} s_{north} = s \cdot \cos(\theta_{rad}) - \text{северная составляющая}, \\ s_{east} = s \cdot \sin(\theta_{rad}) - \text{восточная составляющая}. \end{cases}$$

Рисунок 1 – Алгоритм адаптивного геопоиска с учетом скорости и направления движения
Figure 1 – Adaptive geospatial search algorithm with movement speed and direction parameters

Расчет границ зон поиска производится отдельно для широты и долготы путем вычисления максимальных смещений.

Для широты:

1. Вычисление максимального смещения к северу и югу:

$$\Delta\phi_+ = \frac{r + max(s_{north},0)}{111}, \Delta\phi_- = \frac{r + max(-s_{north},0)}{111}.$$

2. Определение границ широты:

$$\begin{cases}\phi_{max} = \phi_0 + \Delta\phi_+, \\ \phi_{max} = \phi_0 - \Delta\phi_-.\end{cases}$$

Для долготы производится поправка на широту, вычисляются максимальные смещения к востоку и западу:

$$\Delta\lambda_+ = \frac{r + max(s_{east},0)}{111 \cdot \cos(\phi_{rad})}, \Delta\lambda_- = \frac{r + max(-s_{east},0)}{111 \cdot \cos(\phi_{rad})}.$$

На основе смешений вычисляются границы долготы:

$$\begin{cases}\lambda_{max} = \lambda_0 + \Delta\lambda_+, \\ \lambda_{max} = \lambda_0 - \Delta\lambda_-.\end{cases}$$

Таким образом, выходные координаты, описывающие точки зоны поиска, составляются из следующих точек:

– юго-западная точка $(\phi_{min}, \lambda_{min})$;
– северо-восточная точка $(\phi_{max}, \lambda_{max})$.

Данный подход позволяет строить зону поиска с использованием скорости и направления движения клиента, что повышает потенциальную релевантность объектов, передаваемых для работы подсистеме визуализации. В случае $v = 0$ зоной поиска является симметричный прямоугольник, а в случае $v > 0$ – зона ассиметрично расширяется в направлении θ.

После получения координат поисковой зоны производится классический поиск по R-дереву. На выход алгоритм поиска возвращает набор целевых объектов. Схематичное отображение работы алгоритма будет иметь вид, как на Рисунке 2, где 5 – отображение направление и скорости клиента, а цветовое разделение объектов – отношение объекта определенному классу.

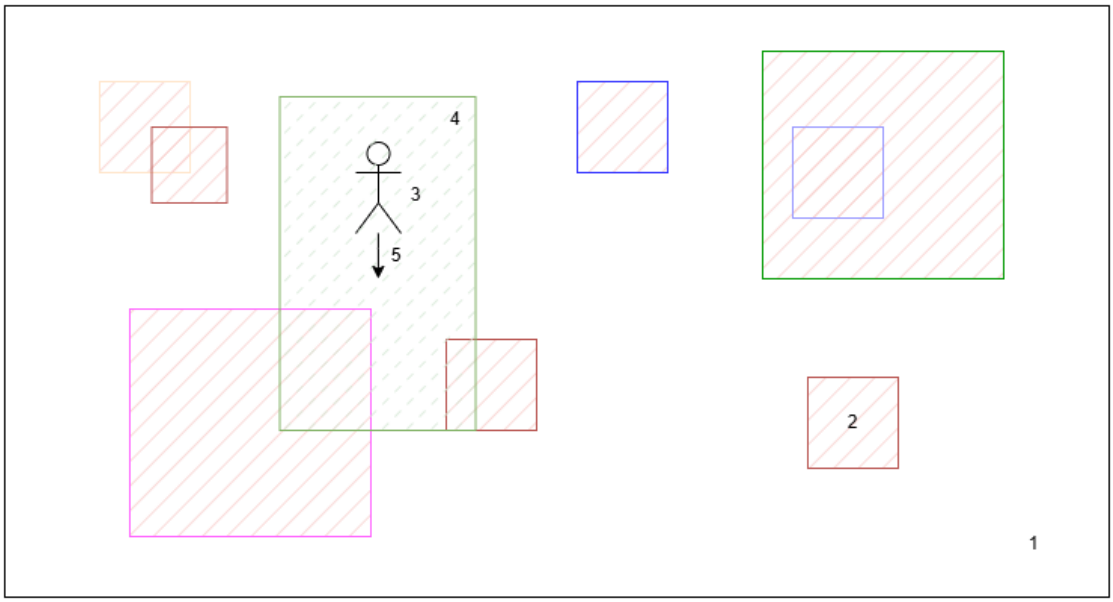

Рисунок 2 – Схематичное отображение работы предложенного алгоритма
Figure 2 – Proposed algorithm principial scheme

Применение предложенного алгоритма в задаче поиска объектов метапространства. Метапространство – это виртуальная трехмерная среда, координаты которой привязаны к координатам реального мира. Метапространство может содержать двумерные и трехмерные объекты, которые интегрируются в изображение реального мира и, в некоторых случаях, вступают с ним во взаимодействие [2].

Сервер маршрутизации метапространства (XRL)[3] отвечает за доставку релевантных данных об объектах метапространства конечному устройству визуализации в зависимости от входящего запроса. Для поиска используется предложенный алгоритм на основе R-дерева. Диаграмма компонентов программного обеспечения представлена на Рисунке 3. Основная задача программного обеспечения – минимизация времени обработки запроса и максимизация релевантности результата.

Программное обеспечение представляет собой серверное приложение с интерфейсом доступа по протоколу REST. Для работы сервера необходимо предоставить набор объектов метапространства, являющихся в рамках задачи целевыми объектами поиска. Набор исходных объектов может обновляться по расписанию либо в ручном режиме.

[3] Дорохин А.А., Дорохин В.А., Теряев Л.Н. Имитационная модель двухколесного транспорта для создания иммерсивных тренажеров расширенной реальности: опубл. 13.11.2024. Свидетельство о государственной регистрации программы для ЭВМ № 2024686966 Российская Федерация.

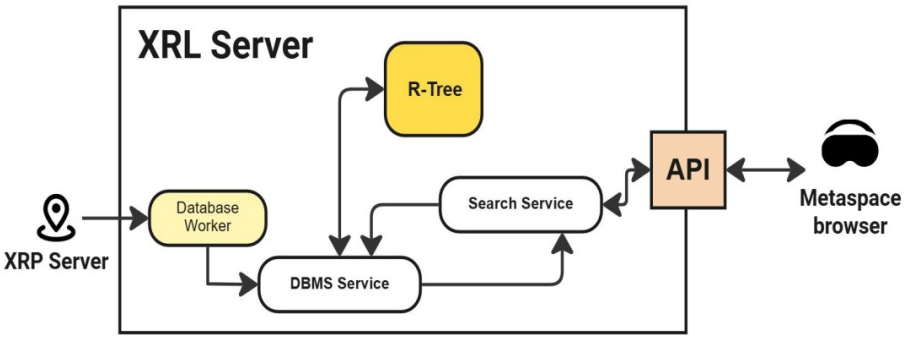

Рисунок 3 – Диаграмма компонентов поискового программного обеспечения
Figure 3 – Search software component diagram

Процесс создания графа происходит при старте системы, либо при необходимости обновления объектов. Обновление может происходить как для одной записи, так и массово. В случае массовой загрузки используются алгоритмы улучшенной балансировки дерева.

Запрос к серверу включает в себя следующий набор параметров:

– *Latitude* – широта, в градусах;
– *Longitude* – долгота, в градусах;
– *Altitude* – высота над уровнем земли, в метрах;
– *Rotation* – азимут направления относительно севера, в градусах;
– *Speed* – скорость, в км/ч.

Полученные значения преобразуются в поисковый квадрат. Для повышения релевантности объектов относительно запроса происходит расчет поискового квадрата в зависимости от направления и скорости. Масштабы и влияние параметров задаются глобальными серверными переменными. Изменение параметров напрямую влияет на вычислительную нагрузку и сетевой трафик.

Полученный поисковой квадрат используется для поиска всех попадающих внутрь объектов с помощью обхода графа в ширину. После получения списка искомых объектов формируется ответ и возвращается клиенту.

Результаты и обсуждение

Для моделирования системы под нагрузкой был разработан экспериментальный клиент, генерирующий данные позиционирования по заданному маршруту. Помимо клиента будет использоваться сгенерированный по средним характеристикам набор виртуальных объектов, на основе которых будет функционировать XRL сервер.

Данный эксперимент позволит выявить:

– соотношение времени инициализации графа объектов XRL сервера в зависимости от количества объектов;
– скорость поиска объектов в зависимости от минимального поискового радиуса;
– сравнить объем сетевого трафика при использовании XRL сервера по сравнению с классической пакетной загрузкой объектов;
– вычислить соотношение релевантных объектов к нерелевантным с использованием классического поиска по квадрату и предложенного алгоритма.

Для проведения эксперимента виртуальные объекты создаются случайным образом, в координатах между 57–58 градусами северной широты и 37–38 градусами восточной долготы. Эксперимент проводится в условиях ограничения одним вычислительным потоком среднестатистического персонального компьютера.

Рост времени инициализации графа объектов XRL сервера в зависимости от количества объектов представлен на Рисунке 4. Как видно из графика, зависимость линейная и составляет примерно 2 секунды для 1 миллиона объектов.

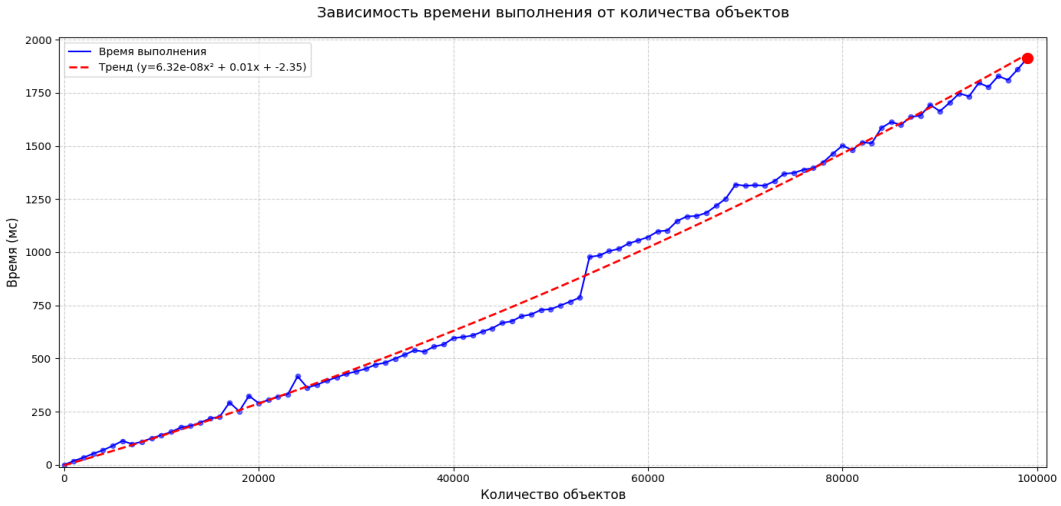

Рисунок 4 – Зависимость времени инициализации поисковой системы от количества объектов
Figure 4 – Search engine initialization time depending on the number of objects

На идентичном наборе данных был проведен замер скорости поиска, результаты представлены на Рисунке 5. Поиск производился в одном и том же квадрате с длиной ребра в 1 км. С увеличением количества объектов росла их плотность на единицу пространства.

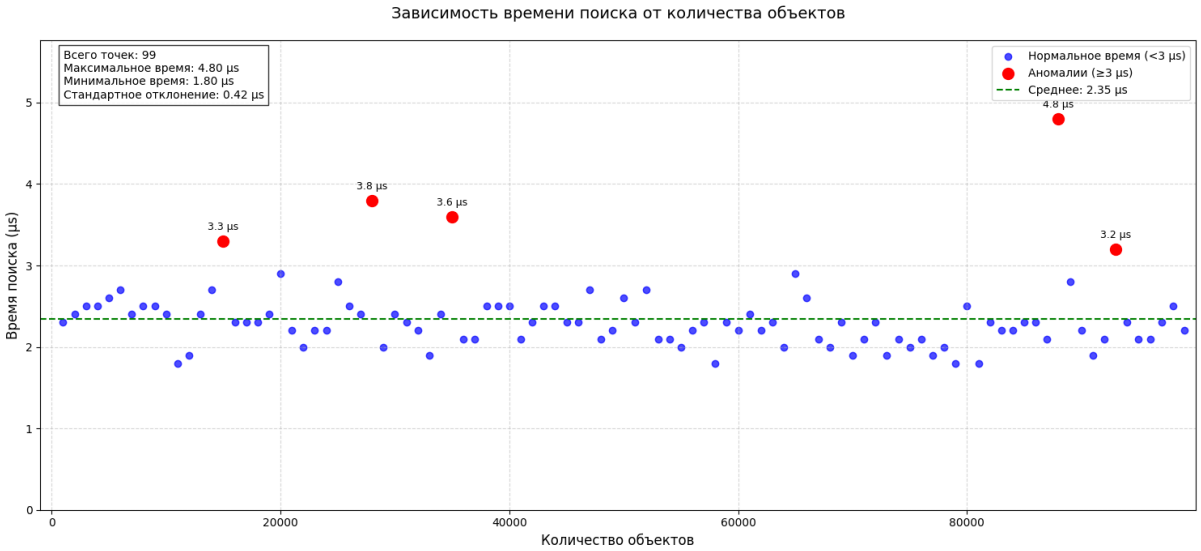

Рисунок 5 – Зависимость времени поиска от количества объектов
Figure 5 – Search time dependence on the number of objects

Как видно из графика, изменение количества объектов не влияет на скорость поиска, за исключением редких аномалий, которые не имеют критического влияния. Влияние на скорость поиска может оказывать степень точности используемых для поиска координат, но эти параметры ограничены настройками конкретного сервера и являются статичным параметром.

Сложность поиска, в том числе, не страдает в процессе увеличения радиуса поиска. На Рисунке 6 отображен график зависимости времени поиска от радиуса поиска (с шагом в 1 км).

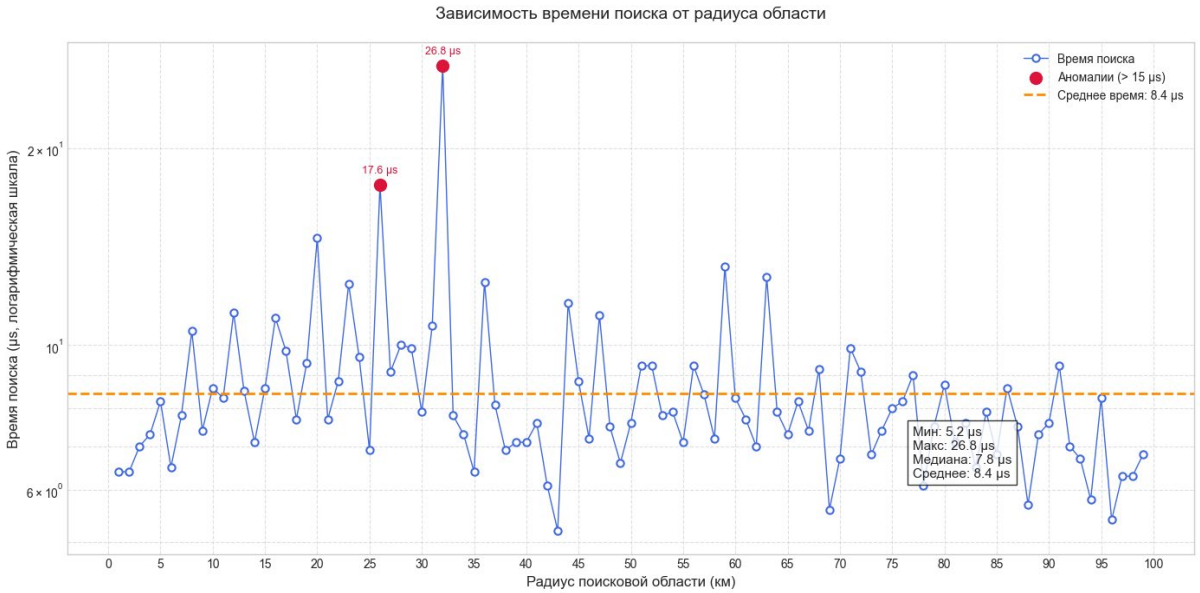

Рисунок 6 – Зависимость времени поиска от радиуса поиска
Figure 6 – Search time dependence on search radius

Для тестирования релевантности поиска был разработан симулятор, позволяющий визуализировать процесс работы клиента с объектами метапространства, а также проводить симуляцию передвижения пользователя. На Рисунке 7 отображены примеры работы симулятора. Красная линия является абстрактной интерпретацией вектора направления пользователя. Синие метки на карте – все объекты, существующие в поисковом множестве. Красные – полученные от сервера как релевантные на основе данных положения, направления и скорости.

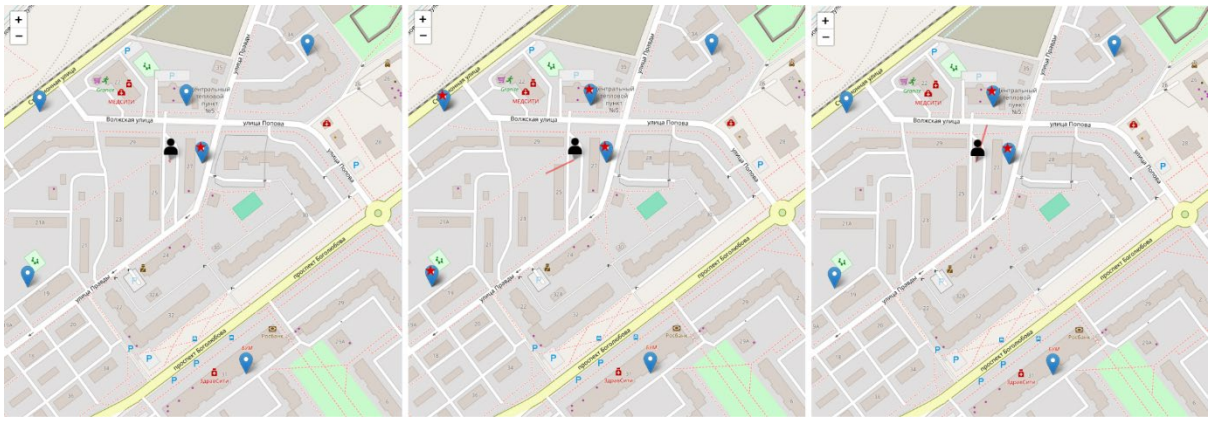

Рисунок 7 – Симулятор клиента метапространства
Figure 7 – Metaspace client simulator

Для определения эффективности разработанного алгоритма релевантного поиска предлагается сравнение по следующему сценарию:

– Генерация определенного количества виртуальных объектов.
– Построение маршрута пользователя.

– Ручное выделение целевых точек, точно попадающих во множество видимых.
– Прохождение маршрута с использованием релевантного поиска.
– Прохождение маршрута с использованием стандартного радиального поиска.
– Сравнение соотношений количества полученных объектов к целевым.

На Рисунке 8 изображен процесс симуляции. На верхней панели отображается общее количество полученных в результате поиска объектов. Обновление списка объектов происходило одновременно для сравниваемых алгоритмов при удалении эмулируемого пользователя от последней точки обновления на 50 метров.

Ввиду использования областей видимости – релевантный алгоритм имеет возможность использовать более низкий базовый радиус поиска при отсутствии движения, а также регулировать его в зависимости от скорости движения.

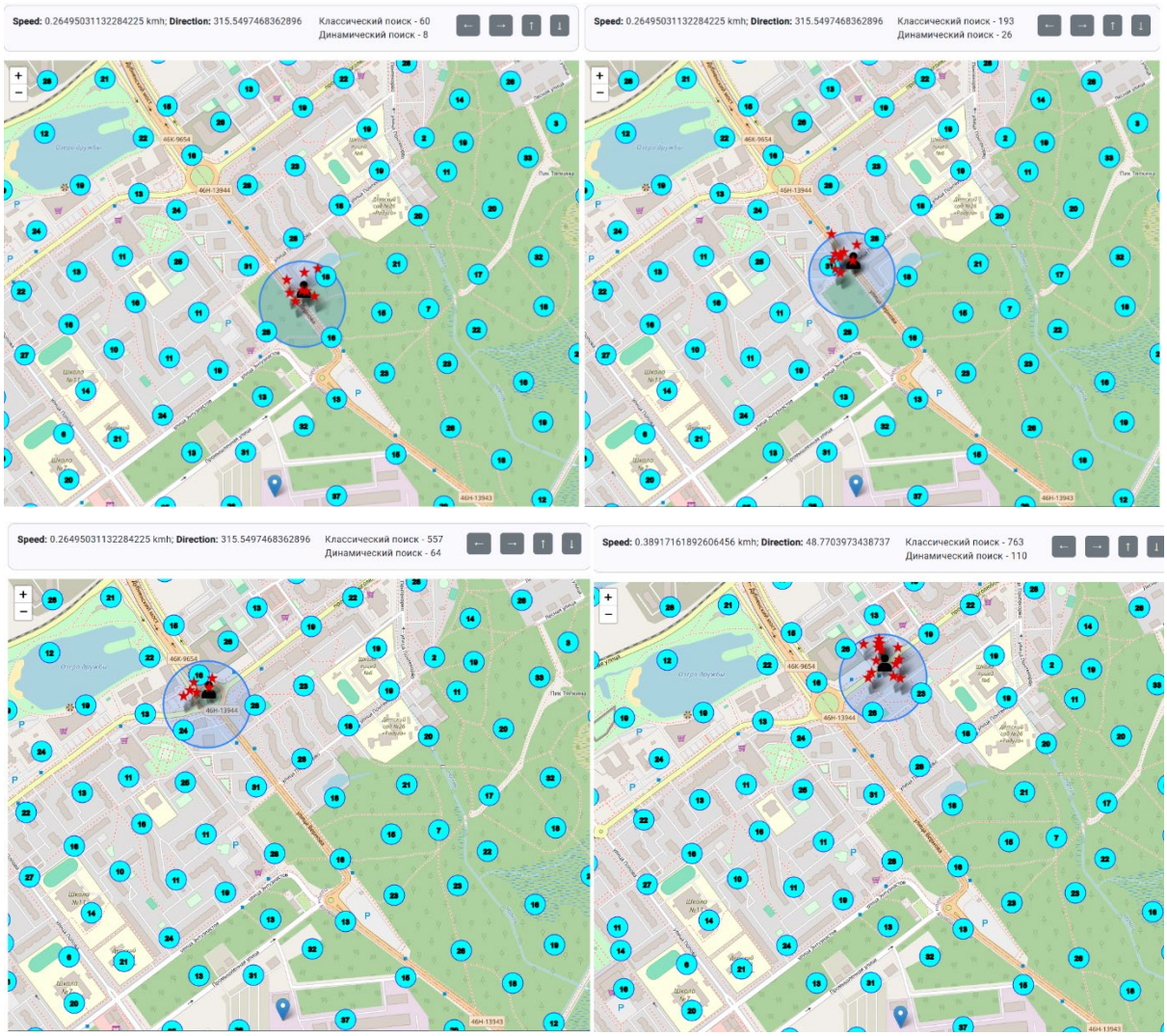

Рисунок 8 – Процесс прохождения маршрута в симуляторе
Figure 8 – The process of a route testing in the simulator

Минимальные радиусы обоих алгоритмов рассчитывались исходя из минимально необходимого для точного нахождения любого из целевых объектов. Представленный симулируемый маршрут включал проход по улице и перпендикулярный поворот. Как видно из рисунка, к концу симуляции присутствовало следующее соотношение полученных объектов:

– классический поиск – 763 объекта;
– релевантный поиск – 110 объектов.

При этом для заданного маршрута было определено 55 целевых объектов, которые точно попадали в зону видимости и отрисовки симулируемого клиента. Таким образом, релевантность классического и предлагаемого геопоиска составляют соответственно 0,072 и 0,5. Значение показателя, равное единице, означает, что пользователь получает исключительно целевые объекты. Проведенные тесты для других маршрутов дают результаты того же порядка. В среднем, прирост показателя релевантности получаемых объектов по результатам замеров различных маршрутов составляет от 4 до 7 раз.

Заключение

Предложенный алгоритм позволяет повысить релевантность геопоиска целевых объектов за счет интеграции скорости и направления движения пользователя в запрос классического пространственного индекса. Данный подход не увеличивает сложность поиска, при этом дает ощутимый прирост релевантности в задачах реального времени за счет уменьшения базового радиуса поиска и оптимизации формы поисковой зоны.

Введение параметра зоны видимости объекта также позволяет повысить релевантность получаемых данных. Внедрение полученного алгоритма для решения задачи поиска объектов метапространства позволило значительно оптимизировать нагрузку на клиентское устройство и сетевой трафик. Алгоритм может применяться в ряде других задач, требующих проведение геопоиска в реальном времени, например, в системах автомобильной навигации.

СПИСОК ИСТОЧНИКОВ / REFERENCES

1. Han Q., Yoshikawa A., Yamamura M. Hierarchical Graph Learning with Cross-Layer Information Propagation for Next Point of Interest Recommendation. *Applied Sciences.* 2025;15(9). https://doi.org/10.3390/app15094979
2. Дорохин В.А., Подгорный С.А., Токарева Н.А. Классификация и модель объектов метапространства расширенной реальности. *Моделирование, оптимизация и информационные технологии.* 2025;13(1). https://doi.org/10.26102/2310-6018/2025.48.1.032
   Dorokhin V.A., Podgornyi S.A., Tokareva N.A. Classification and Model of Objects in Extended Reality Metaspace. *Modeling, Optimization and Information Technology.* 2025;13(1). (In Russ.). https://doi.org/10.26102/2310-6018/2025.48.1.032
3. Shi Y., Hendawi A.H., Fattah H., Mohamed A. RxSpatial: Reactive Spatial Library for Real-Time Location Tracking and Processing. In: *SIGMOD '16: Proceedings of the 2016 International Conference on Management of Data, 26 June – 1 July 2016, San Francisco, CA, USA.* New York: Association for Computing Machinery; 2016. P. 2165–2168. https://doi.org/10.1145/2882903.2899411
4. Nimbalkar V.P. Enhancing Vehicle Navigation and Safety Through Integration of Pre-Recorded Maps with Vehicle Control Unit. *International Journal of Innovative Research in Computer Science and Technology.* 2024;12(4):117–123. https://doi.org/10.55524/ijircst.2024.12.4.18
5. Li P., Wang Z., Zhang X., Wang P., Liu K. Next Arrival and Destination Prediction via Spatiotemporal Embedding with Urban Geography and Human Mobility Data. *Mathematics.* 2025;13(5). https://doi.org/10.3390/math13050746

6. Nutheti S.S. POI Recommendation from a Data Perspective. ResearchGate. URL: https://www.researchgate.net/publication/392708648_POI_Recommendation_from_a_Data_Perspective [Accessed 12th September 2025].
7. Guttman A. R-Trees: A Dynamic Index Structure for Spatial Searching. In: *SIGMOD'84, Proceedings of Annual Meeting, 18–21 June 1984, Boston, MA, USA.* ACM Press; 1984. P. 47–57. http://doi.acm.org/10.1145/602259.602266
8. Beckmann N., Kriegel H.-P., Schneider R., Seeger B. The R*-Tree: An Efficient and Robust Access Method for Points and Rectangles. In: *SIGMOD '90: Proceedings of the 1990 ACM SIGMOD International Conference on Management of Data, 23–26 May 1990, Atlantic City, NJ, USA.* New York: Association for Computing Machinery; 1990. P. 322–331. http://doi.org/10.1145/93597.98741
9. Beckmann N., Seeger B. A Revised R*-Tree in Comparison with Related Index Structures. In: *SIGMOD '90: Proceedings of the 1990 ACM SIGMOD International Conference on Management of Data, 23–26 May 1990, Atlantic City, NJ, USA.* New York: Association for Computing Machinery; 1990. P. 799–812. http://doi.org/10.1145/1559845.1559929
10. Kamel I., Faloutsos Ch. Hilbert R-Tree: An Improved R-Tree Using Fractals. In: *VLDB '94: Proceedings of the 20th International Conference on Very Large Data Bases, 12–15 September 1994, Santiago de Chile, Chile.* San Francisco: Morgan Kaufmann; 1994. P. 500–509.
11. Lee T., Lee S. OMT: Overlap Minimizing Top-Down Bulk Loading Algorithm for R-Tree. In: *The 15th Conference on Advanced Information Systems Engineering (CAiSE '03), CAiSE Forum, Short Paper Proceedings, Information Systems for a Connected Society, 16–20 June 2003, Klagenfurt/Velden, Austria.* 2003. P. 69–72.

ИНФОРМАЦИЯ ОБ АВТОРЕ / INFORMATION ABOUT THE AUTHOR

Дорохин Виктор Александрович, старший преподаватель Государственного университета «Дубна», Дубна, Российская Федерация.
e-mail: victor.doroh@gmail.com

Victor A. Dorokhin, Senior Lecturer of Dubna State University, Dubna, the Russian Federation.

Статья поступила в редакцию 06.10.2025; одобрена после рецензирования 17.11.2025; принята к публикации 26.11.2025.

The article was submitted 06.10.2025; approved after reviewing 17.11.2025; accepted for publication 26.11.2025.