УДК 004.896:519.876.5:004.94 **DOI** 10.15622/ia.24.2.5

В.В. Курейчик, В.И. Данильченко, Е.В. Данильченко

МАРШРУТИЗАЦИЯ АВТОНОМНЫХ УСТРОЙСТВ В ТРЁХМЕРНОМ ПРОСТРАНСТВЕ

Курейчик В.В., Данильченко В.И., Данильченко Е.В. **Маршрутизация автономных устройств в трёхмерном пространстве.**

Аннотация. Статья посвящена решению проблемы маршрутизации автономных устройств в трёхмерном пространстве, что является актуальной задачей в области интеллектуального управления. Трёхмерное пространство отличается высокой степенью свободы, сложной топологией и динамическими изменениями среды, что значительно усложняет задачу эффективного планирования траекторий. Разработка методов маршрутизации, обеспечивающих безопасность, энерго и вычислительную эффективность, имеет ключевое значение для повышения производительности автономных систем. В работе рассматривается комплексная система маршрутизации, основанная на гибридном подходе, объединяющем высокоуровневое моделирование рабочего пространства с метаэвристическими методами оптимизации. Для представления трёхмерной среды используются иерархические структуры данных, такие как октодеревья, что обеспечивает компактность и гибкость пространственных моделей. Эти модели преобразуются в графовые структуры, что позволяет описать маршрутизацию в виде оптимизационной задачи на графах. Предложен модифицированный метаэвристический муравьиный алгоритм, относящийся к классу роевых методов оптимизации. Алгоритм ориентирован на построение безопасных и энергоэффективных маршрутов, а также на решение задач поиска кратчайших гамельтоновых циклов и динамической перенастройки маршрута в условиях изменяющейся внешней среды. В работе представлены результаты вычислительного эксперимента, включающие тестирование алгоритма в условиях трёхмерного пространства, и сравнительный анализ с другими алгоритмами маршрутизации. Вычислительный эксперимент подтвердил эффективность разработанного алгоритма маршрутизации, включая сокращение времени вычислений и повышение энергоэффективности автономных устройств. Перспективы дальнейших исследований включают интеграцию предложенной системы в широкий спектр приложений для автономных устройств, направленных на оптимизацию процессов управления и повышение эффективности в динамически изменяющейся внешней среде. Отметим, что разработанный алгоритм может быть адаптирован для решения комплексных задач, в которых маршрутизация и размещение ветрогенераторов на плоскости взаимосвязаны. Задача размещения напрямую связана с построением маршрутов для обслуживания этих объектов, что требует комплексного подхода для эффективного решения этих задач. Это станет частью системы поддержки принятия решений, предназначенной для планирования и обслуживания ветрогенераторных комплексов, обеспечивая их эффективное функционирование и управление ресурсами.

Ключевые слова: метаэвристический алгоритм, муравьиная оптимизация, графовые математические модели, маршрутизация, моделирование трёхмерного пространства, энергетические системы.

1. Введение.

Задача маршрутизации автономных устройств в трёхмерном пространстве является одной из ключевых проблем в области робототехники, автономных транспортных систем

и беспилотных летательных аппаратов. С развитием технологий автономных систем, увеличением вычислительных мощностей и повышением требований к безопасности и энергоэффективности, проблема эффективного планирования траекторий приобрела особую актуальность. В последние десятилетия для решения этой задачи применяются как классические алгоритмы, так и современные методы, такие как гибридные метаэвристические подходы, основанные на комбинировании различных стратегий поиска оптимальных решений [1 – 3]. Таким образом, задача маршрутизации автономных устройств в трехмерном пространстве является важным и актуальным направлением исследований, которое требует модифицированных подходов и методов для повышения качества и эффективности навигации в сложных и динамичных условиях.

2. Анализ существующих решений. Среди известных и широко применяемых методов маршрутизации выделяется алгоритм Дейкстры, предложенный Эдсгером Дейкстрой в 1956 году [1]. Он используется для поиска кратчайших путей в статичных графах и гарантирует нахождение оптимального маршрута. Отметим, что его вычислительная сложность возрастает с увеличением размера пространства, что ограничивает его эффективность для больших или динамично изменяющихся сред [4 – 6]. Алгоритм «A star» или «A*», предложенный в 1968 году Патриком Хартом, является улучшенной версией алгоритма Дейкстры, так как использует эвристическую функцию для оценки проектного решения, что позволяет ускорить процесс поиска [1]. Но вместе с тем данный алгоритм имеет ограничения в работе с трехмерными пространствами, так как для его реализации требуется большие вычислительные затраты, то есть необходима квадратичная вычислительная сложность. В литературе представлено множество его модификаций, таких как Theta*, Lifelong Planning A (LPA*), D* и его расширенная версия D* Lite [1 – 8].

Среди множества модификаций алгоритма A* особый интерес представляют методы, направленные на его адаптацию к динамическим средам. Одним из примеров таких алгоритмов является Theta* [7], представляющий собой модификацию традиционного алгоритма A*. Он позволяет строить квазиоптимальные маршруты, учитывая прямую видимость между узлами. Однако, несмотря на это преимущество, Theta* требует значительных вычислительных ресурсов, особенно в сложных трехмерных средах [7, 8]. В отличие от него, алгоритм Lifelong Planning A* (LPA*) [8] предназначен для эффективного перерасчета маршрутов при изменении внешних условий. Тем не менее, LPA* сохраняет структуру классического A*,

что ограничивает его применение в динамических пространствах с большим количеством препятствий.

Для решения задачи маршрутизации разработаны алгоритмы семейства D*. Оригинальный алгоритм D* [7] обеспечивает адаптацию маршрута в реальном времени без необходимости полного пересчета маршрута, что повышает его эффективность для автономных систем. Улучшенная версия, D* Lite [8], упрощает вычисления и ускоряет процесс обновления маршрута, сохраняя высокую точность. Однако оба метода ориентированы преимущественно на двухмерные пространства и имеют значительные вычислительные затраты при расширении на трехмерные среды, поскольку количество возможных направлений движения возрастает экспоненциально, что значительно увеличивает сложность поиска.

Альтернативные подходы к маршрутизации включают графовые методы, такие как дорожные карты и диаграммы Вороного, которые широко применяются для навигации в сложных пространствах. Одним из известных и эффективных методов является Probabilistic Roadmap (PRM) [9], строящий граф возможных маршрутов, при этом используя случайную дискретизацию пространства свободного движения. Такой метод эффективен в работе с предварительно известной средой (предварительно отсортированными данными), так как позволяет заранее сформировать карту маршрутов, что позволяет ускорить процесс поиска. Однако его эффективность значительно снижается в динамически изменяющихся условиях, где требуется частое перестроение графа, что увеличивает вычислительные затраты.

Другим популярным и эффективным методом является Rapidly-exploring Random Trees (RRT) [9], который выполняет исследование пространства с помощью случайных деревьев. Данный алгоритм применяется для построения маршрута в трехмерных пространствах с высокой сложностью конфигурации, поскольку он эффективно охватывает большие области и находит маршруты в условиях высокой неопределенности. Важное преимущество RRT заключается в его способности быстро адаптироваться к изменениям в среде. Однако базовая версия алгоритма не гарантирует нахождение оптимального маршрута, а в условиях динамически изменяющейся среды приводит к нестабильным решениям, требующим дополнительной оптимизации.

В таблице 1 приведены вычислительные сложности рассматриваемых алгоритмов маршрутизации, включая как классические методы, так и метаэвристические подходы, которые используются для навигации в динамических пространствах.

Таблица 1. Вычислительная сложность алгоритмов

Алгоритм	**Вычислительная сложность**	**Примечание**
A*	*O(n*logh(n))*	h(n) – эвристическая функция
Theta*	*O(n*logn)*	Преимущество – учет прямой видимости между узлами
LPA*	*O(n*logn)*	Эффективен для динамических изменений в графе
D*	*O(n*logn)*	Подходит для динамических сред, пересчет маршрута в реальном времени
D* Lite	*O(n*logn)*	Упрощенная версия D*, подходит для автономных систем
RRT	*O(n)*	Зависит от количества итераций, эффективен в высокоразмерных пространствах
PRM	*O(n*k)*	Может потребоваться повторный запуск алгоритма при неудаче в нахождении пути
ACO (Муравьиный алгоритм)	*O(n*m)*	Адаптивность и самоорганизация, подходит для динамических сред, эффективно находит квазиоптимальные маршруты
PSO (Метод роя частиц)	*O(n*m)*	Эффективен для нахождения квазиоптимальных решений за полиномиальное время, имитирует коллективное поведение агентов
GA (Генетические алгоритмы)	*O(n*m)*	Требует значительных вычислительных ресурсов для многокритериальных задач, основан на принципах естественного отбора

Несмотря на рассмотренные преимущества, перечисленные методы имеют ограничения, не позволяющие эффективно применять их для маршрутизации автономных устройств в трехмерном пространстве. Высокие вычислительные затраты, сложность обработки объемных данных и ограниченная адаптивность к быстро меняющимся условиям снижают их применимость в сложных сценариях. В связи с этим возникает необходимость разработки модифицированных эвристических подходов и методов, способных эффективно решать задачу маршрутизации в трехмерной динамической среде.

Одним из таких подходов для решения задачи маршрутизации является использование метаэвристических методов, которые реализуют различные стратегии поиска, тем самым повышая эффективность маршрутизации. Среди них метод на основе роя частиц (PSO) имитирующих коллективное поведение агентов, генетические алгоритмы (GA), основанные на принципах естественного отбора. Как известно данные методы позволяют эффективно находить квазиоптимальные решения за полиномиальное время в оптимизационных задачах. Однако, при решении многокритериальных задач, эти методы требуют значительных вычислительных ресурсов, что ограничивает их применимость в реальных условиях с динамически изменяющейся средой [1 – 4].

Одним из наиболее перспективных и широко применяемых методов в данной области является метаэвристический метод муравьиной оптимизации (ACO), который, как известно из литературных источников, позволяет находить кратчайшие маршруты используя коллективное поведение биологических агентов, обладающими такими свойствами, как автономность, коммуникабельность, адаптация и самоорганизация, присущими живым системам [1, 3]. Дополнительным преимуществом таких подходов является их способность динамически реагировать на изменение условий внешней среды. В частности, использование таких метаэвристических методов в задачах маршрутизации автономных устройств в трёхмерном пространстве позволяет значительно сократить время, затраченное на вычисления, и оптимизировать процесс планирования траекторий с учетом множества критериев, таких как скорость, потребление энергии, обход препятствий, безопасность и предотвращение столкновений.

В исследовании рассматривается модифицированный муравьиный алгоритм оптимизации, который, на основе проведенного анализа, продемонстрировал высокий потенциал для решения задачи маршрутизации в трехмерном пространстве. Этот метод обеспечивает эффективный баланс между вычислительной сложностью, адаптивностью к изменениям окружающей среды и качеством маршрутов, являясь особенно перспективным для навигации автономных систем в условиях динамичных и сложных сред.

2. Постановка задачи. Задача маршрутизации автономных устройств в трёхмерном пространстве заключается в поиске оптимального маршрута для обхода заданных контрольных точек. Это представляет собой адаптированную вариацию классической задачи коммивояжёра, но с учётом специфики трёхмерного пространства.

Главное отличие заключается в необходимости предварительной формализации задачи, что требует построения полной взвешенной графовой модели на основе трёхмерных данных. Формализация достигается путём вычисления кратчайших путей между контрольными точками и созданием матрицы весов, которая отражает стоимость перемещения между вершинами графа [1, 5 – 7].

Задачу маршрутизации предлагается разделить на два ключевых этапа. Первый этап включает построение математической модели в виде графа, основанной на трёхмерном пространстве с возможными препятствиями. Для этого предлагается упрощённая пространственная модель, которая сохраняет размерность исходного пространства, но с адаптированной сложностью геометрии для снижения вычислительной нагрузки. Разработка упрощённой модели трёхмерного пространства, должна учитывать геометрические особенности и возможные преграды. Модель пространства представим в виде графа $G(X, U)$ (рисунок 1), где X – множество вершин (контрольных точек), а U – множество рёбер, соединяющих вершины.

Контрольные точки – это заранее определённые позиции в пространстве, через которые должен пройти маршрут автономного устройства. Они могут быть связаны с физическими объектами или быть абстрактными позициями, такими как места доставки или важные ориентиры на пути. Формирование контрольных точек происходит на основе анализа пространства с учётом геометрии, наличия препятствий и требований задачи. Контрольные точки выбираются так, чтобы обеспечивать оптимальный маршрут, минимизируя расстояние, время или затраты энергии, а также избегать столкновений с препятствиями.

В работе под динамической средой понимается система, в которой объекты и условия изменяются во времени. Эти изменения включают как перемещения объектов, так и изменения их характеристик или состояния, а также структуры среды, например, появления новых препятствий или исключение существующих. В динамической среде устройства или системы способны адаптироваться к этим изменениям в реальном времени, что требует высокой гибкости в планировании маршрутов и принятии решений. Ограничения на динамические объекты включают неопределённость в их поведении, трудности в прогнозировании будущих изменений и своевременной реакции на эти события [7 – 9].

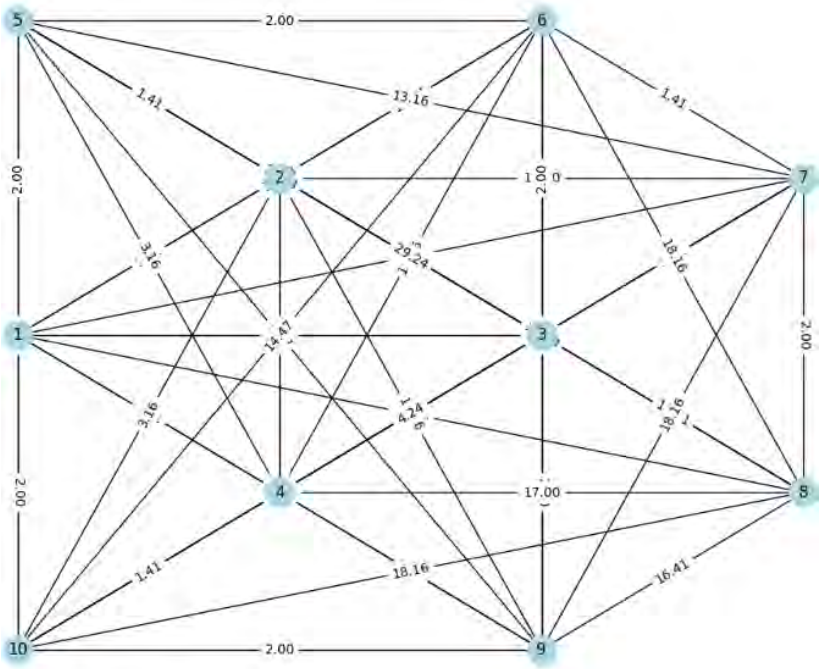

Рис. 1. Математическая модель размещения группы ветрогенераторов

Далее производятся вычисления кратчайших путей между каждой парой контрольных точек с учётом препятствий, и формируется матрица весов коэффициентов W, где элементы матрицы $W(i,j)$ представляют собой численное представление перемещения от вершины i к вершине j. Эта величина рассчитываться на основе следующего выражения [1]:

$$W(i,j) = \sqrt{\left(x_i - x_j\right)^2 + \left(y_i - y_j\right)^2 + \left(z_i - z_j\right)^2}, \quad (1)$$

где (x_i, y_i, z_i) и $\left(x_j, y_j, z_j\right)$ – координаты вершин i и j.

Если в пространстве присутствуют препятствия, то дополнительно вводится функция штрафа $P(i,j)$, которая корректирует вес ребра, добавляя дополнительный вес для ребра между вершинами i и j в случае наличия препятствий. Тогда матрица весов вычисляется в соответствии со следующим выражением:

$$W(i,j) = \sqrt{\left(x_i - x_j\right)^2 + \left(y_i - y_j\right)^2 + \left(z_i - z_j\right)^2} + P(i,j). \quad (2)$$

Функция штрафа $P(i,j)$ для рёбер между вершинами i и j выражается как сумма двух критериев:

$$P(i,j) = P_{dist}(i,j) + P_o(i,j), \quad (3)$$

где $P_{dist}(i,j)$ – штраф, зависящий от минимального расстояния до препятствия, который корректирует вес рёбер в графе в зависимости от коэффициента сложности $K_{\text{тип}}$:

$$P_{dist}(i,j) = \frac{K_{\text{тип}}}{dist_{min} + \epsilon}, \quad (4)$$

где $dist_{min}$ – минимальное расстояние до препятствия, ϵ – коэффициент для предотвращения деления на ноль. Если на траектории движения встречается препятствие, то требуется его обход, представляющий собой дополнительное увеличение веса рёбер, пропорциональное длине пути, необходимого для обхода препятствия, что определяется следующим образом:

$$P_o(i,j) = K_{\text{тип}} \cdot L_{\text{обход}}, \quad (5)$$

где $L_{\text{обход}}$ – длина дополнительного пути.

Введение функции штрафов позволяет учесть наличие препятствий в маршруте и корректировать веса рёбер в графе в зависимости от их воздействия на движение. При наличии крупных объектов, таких как горы, здания или зоны запрета, алгоритм не только учитывает расстояние до препятствия, но и оценивает сложность его обхода. Отметим, что для динамической адаптации движения требуется проведение дополнительного анализа и применение методов, позволяющих своевременно корректировать маршрут в ответ на изменения внешней среды. Такой подход позволит обеспечить универсальность модели и её соответствие реальным условиям, в которых динамические препятствия могут существенно влиять на эффективность маршрута.

Далее производится построение гамильтонова цикла, то есть нахождения последовательности вершин, которая проходит через все

контрольные точки с возвратом в исходную точку, что выражается следующим образом:

$$\mathrm{C} = \min_{\sigma \in S_n} \left(\sum_{i=1}^{n-1} W\left(v_{\sigma(i)}, v_{\sigma(i+1)}\right) + W\left(v_{\sigma(n)}, v_{\sigma(1)}\right) \right), \quad (6)$$

где σ – перестановка вершин графа, S_n – множество всех перестановок длины n, а $v_{\sigma(1)}$ – вершина маршрута.

Построение общей длины маршрута C вычисляется в соответствии со следующим выражением [8]:

$$C = \sum_{i=1}^{n-1} W(v_i, v_{i+1}) + W(v_n, v_1), \quad (7)$$

где v_i – вершины маршрута входящие в маршрут, а n – общее количество вершин.

Для обеспечения адаптивности к динамическим изменениям условий внешней среды важным аспектом является уменьшение времени перерасчёта построенного маршрута. Это достигается с использованием муравьиной эвристики, которая позволяет эффективно обновлять маршрут в ответ на динамические изменения в среде. Для повышения вычислительной эффективности алгоритма и минимизации затрат времени перерасчёта маршрутов предлагается ввести ограничение на объём перерасчётов. Это ограничение заключается в обработке только тех участков графа, которые были изменены, что значительно снижает вычислительную нагрузку. Отметим, что это ограничение используется, когда в пространстве появляются новые препятствия или изменения в характеристиках существующих, что влияет только на определённые участки маршрута. Механизм перерасчёта маршрута с учётом изменений в графе определяется следующим образом:

$$C_{new} = \sum_{i=1}^{k} W(v_i, v_{i+1}) + \sum_{i=k+1}^{n} W(v_n, v_1), \quad (8)$$

где C_{new} – новый маршрут после перерасчёта, k – количество изменённых вершин графа.

Для повышения качества решений в модель также введен критерий энергоэффективности маршрута. Этот критерий позволяет учитывать не только кратчайшую длину маршрута, но и количество энергии, затраченное на его выполнение. Как известно из литературы [4 – 7] критерий энергоэффективности имеет одно из ключевых значений при планировании маршрутов для автономных устройств, особенно в условиях ограниченных энергетических ресурсов. Энергоэффективности маршрута вычисляется на основе следующего выражения:

$$E_{eff} = \frac{d_{opt}}{E_{spent}}, \tag{9}$$

где d_{opt} – длина кратчайшего маршрута, E_{spent} – энергия, затраченная на его выполнение.

Применение такого подхода позволяет сравнивать различные маршруты с учётом не только их длины, но и энергетических затрат, что помогает выбрать наиболее приемлемый маршрут с точки зрения баланса между длиной маршрута и затрачиваемой энергией на его прохождение. Отметим, что энергия, затраченная на выполнение маршрута, зависит от нескольких факторов, таких как сопротивление воздуха, скорости направления ветра, углы подъёма/спуска, скорость перемещения устройства и др. Для автономных устройств, уменьшение энергии, требуемой для выполнения маршрута, позволяет значительно увеличить время их работы. Тогда расчет энергии, требуемой на преодоление каждого сегмента маршрута, вычисляется на основе следующего выражения:

$$E_{segment} = \alpha \cdot d_{segment}^2 + \beta \cdot d_{segment} + \gamma, \tag{10}$$

где α, β, и γ – коэффициенты, учитывающие физические особенности движения. При этом энергия, затраченная на весь маршрут, вычисляется как сумма энергий, затраченных на прохождения каждого сегмента:

$$E_{spent} = \sum_{i=1}^{n-1} E_{segment}(v_i, v_{i+1}), \tag{11}$$

где $E_{segment}$ – энергия, необходимая для преодоления пути между вершинами v_i и v_{i+1}. Таким образом, учет энергоэффективности в задаче маршрутизации позволяет повысить эксплуатационные характеристики автономных устройств, обеспечивая баланс между длиной маршрута и продолжительностью работы устройства. Отметим, что при применении данного подхода к кусочно-непрерывной траектории возникают разрывы в скорости движения автономного устройства и нарушается физическая реализуемость маршрута. Для улучшения маршрутизации автономных устройств в сложных средах предлагается модификация алгоритма маршрутизации, включающая процедуру сглаживания траектории, которая минимизирует острые углы и избыточные точки, повышая точность и энергоэффективность при сохранении баланса между оптимизацией пути и его точностью. Что позволяет устранить разрывы в скорости и обеспечить более реалистичное моделирование движения устройства.

Данная модель позволяет учесть не только длину маршрута, но и его сложность, обусловленную особенностями среды и динамическими характеристиками устройства. Важно отметить, что при моделировании маршрутов необходимо учитывать дополнительные энергозатраты, связанные с маневрами устройства, такими как торможение и разгон, что особенно важно при резких поворотах и изменениях направления движения. Для учёта этих факторов можно дополнительно ввести компоненту, отражающую энергозатраты, возникающие при маневрах:

$$E_{segment} = \alpha \cdot d_{segment}^2 + \beta \cdot d_{segment} + \gamma + \delta f(\theta), \qquad (12)$$

где $f(\theta)$ – функция, описывающая дополнительные энергозатраты, вызванные маневрами, а δ – коэффициент, учитывающий влияние этих затрат [8 – 10].

Полиномиальная модель энергозатрат учитывает различные аспекты движения устройства. Квадратичная компонента $\alpha \cdot d_{segment}^2$ отражает увеличение энергозатрат с ростом расстояния, включая аэродинамическое сопротивление и затраты на разгон или торможение. Линейная зависимость $\beta \cdot d_{segment}$ моделирует постоянные потери энергии, такие как трение и механические потери. Константная составляющая γ учитывает фиксированные энергозатраты на работу сенсоров или систем управления устройства.

При этом автономное устройство в данной задаче рассматривается как объект с тремя степенями свободы, обладающий определёнными формой, размерами и весом, а также точками приложения силы. Эти характеристики учитываются при построении маршрута, что позволяет обеспечить точное соответствие модели реальным условиям эксплуатации и учитывать факторы, влияющие на маневренность, энергоэффективность и безопасность движения устройства в трёхмерном пространстве.

3. Математическая модель трёхмерного пространства. Проблема эффективной обработки больших трёхмерных пространств с множеством объектов является одной из ключевых в области робототехники, компьютерных наук и геоинформационных систем. В рамках этой задачи большое внимание уделяется разработке методов для упрощения представления пространства и ускорения процессов обработки. Одним из наиболее распространённых решений является использование древовидных структур данных, таких как октодеревья и k-мерные деревья, которые широко применяются в таких областях, как компьютерная графика, геодезия и других [7 – 9].

Октодеревья представляют собой одну из наиболее эффективных структур для представления трёхмерного пространства [3, 9]. В отличие от k-мерных деревьев, которые являются бинарными и делят пространство на два подмножества, октодерево делит каждую ячейку на восемь равных частей, что позволяет более эффективно обрабатывать данные в трёхмерной среде. Это свойство делает октодеревья особенно подходящими для задач, связанных с анализом соседства, пространственной индексацией, а также для поиска коллизий между объектами. В своей основе октодерево является структурой, способной динамически адаптировать уровень детализации пространства, что позволяет эффективно управлять объёмами данных и минимизировать вычислительные затраты. Отметим, что октодеревья демонстрируют высокую эффективность при выполнении задач поиска ближайших соседей в трёхмерной среде. Иерархическая структура, которая последовательно делит пространство, позволяет быстро исключать области, не содержащие искомых объектов, существенно сокращая объём вычислений. Такое преимущество особенно важно при решении задач в масштабах реального времени, таких как маршрутизация автономных устройств. Также, универсальность октодеревьев заключается в их способности работать как со статичными, так и с динамически изменяемыми сценариями маршрутизации.

При этом модель октодерева в трёхмерном пространстве имеет следующие ограничения.

1. Зависимость производительности от уровня детализации (глубины дерева). С увеличением глубины дерева растёт количество узлов, что повышает вычислительные затраты. Количество ячеек на каждом уровне дерева пропорционально 8^d, где d – глубина дерева. Следовательно, производительность системы будет зависеть от точности, требуемой для представления объектов. Для поиска ближайших соседей или коллизий, например, число вычислений будет возрастать экспоненциально с увеличением глубины поиска. Тогда количество ячеек в дереве при глубине d определяется следующим образом:

$$N_{cells} = 8^d, \tag{13}$$

где N_{cells} – количество ячеек дерева. Заметим, если глубина октодерева превышает заданный порог, это приводит к значительным затратам памяти и времени, так как необходимо хранить и обрабатывать большое количество данных. При этом глубина октодерева определяет степень детализации модели: с увеличением числа уровней повышается точность представления структуры пространства, однако это ведёт к увеличению вычислительных нагрузок [3, 10]. При выборе оптимальной глубины октодерева необходимо учитывать несколько факторов. Для объектов, занимающих большую часть рабочего пространства, достаточно использовать октодерево минимальной глубины, что позволяет эффективно представлять их положение и взаимодействие без избыточного деления ячеек, тем самым сокращая объём вычислений. Для учета мелких объектов необходимо повышать детализацию, так как размеры ячеек на верхних уровнях октодерева могут быть слишком большими для их точного представления. При решении задач в динамически изменяющихся средах, оптимальная глубина дерева позволяет минимизировать объём перерасчётов при сохранении точности модели. Приемлемая глубина октодерева рассчитывается на основе следующего выражения:

$$d = \left\lceil \log_2 \frac{L}{L_{min}} \right\rceil, \tag{14}$$

где L – размер пространства, а L_{min} – минимальный размер объекта.

2. Управление памятью при работе с большими пространствами. При работе с большими пространствами и динамически изменяющимися сценариями, управление памятью становится актуальной задачей. В случае изменения положения объектов в рабочем пространстве, необходимо обновлять только те ветви дерева, которые затронуты, что существенно сокращает вычислительные затраты и увеличивает производительность системы.

Опишем процесс динамического обновления октодерева, где каждый узел может быть «свободным» или «занятым». Пусть T – это октодерево, $T(i)$ – это i узел этого дерева, O_i – объект, который может занимать или не занимать пространство в ячейке $T(i)$. Тогда состояние узла $T(i)$ в определённый момент времени будет определяться следующим образом [8 – 10]:

$$T(i) = \begin{cases} \text{занят}, & \text{если } O_i \text{ пересекает или находится в } T(i) \\ \text{свободен}, & \text{если } O_i \text{ не пересекает } T(i). \end{cases} \quad (15)$$

Алгоритм определения множества узлов, которые изменили своё состояние, включает следующие этапы:

1. для каждого объекта O_i, который изменил своё положение, вычисляется его новое пространственное расположение;
2. выполняется пересечение нового объёма объекта с существующей структурой октодерева;
3. формируется множество a_n, состоящее из всех узлов $T(i)$, которые пересекаются с новым положением объекта O_i , но не пересекались с его предыдущим положением;
4. аналогично, в a_n добавляются узлы, которые перестали пересекаться с объектом после его перемещения;
5. только узлы из множества a_n подвергаются обновлению состояния, что значительно снижает вычислительные затраты.

Такой подход позволяет минимизировать обновления структуры, оптимизируя вычислительные ресурсы при динамическом изменении сцены [10, 11].

При перемещении объект O_i необходимо производить пересчет только тех узлов, которые изменили свое состояние, в противном случае обновление состояния будет происходить для всех узлов. Данный процесс обновления состояния узлов дерева определяется следующим образом:

$$\forall i \in= a_n \qquad T(i) = update(O_i), \quad (16)$$

где a_n – это множество узлов, которые пересекаются с перемещаемым объектом. Это значительно снижает вычислительные затраты, особенно в случае динамических сцен с частыми изменениями положения объектов в пространстве за счет обновления только части используемого дерева.

Такой механизм обновления узлов деревьев в динамических сценариях помогает поддерживать высокую производительность системы маршрутизации и минимизировать ресурсоёмкость вычислений, что критически важно для автономных устройств, работающих в сложных и изменяющихся внешних условиях.

4. Построение модели маршрутизации. Для решения задачи маршрутизации автономных устройств в трёхмерном пространстве, моделируемом в виде графа, применяется эвристический алгоритм A* (A-star), который является модификацией алгоритма Дейкстры [1, 8 – 11]. Этот подход позволяет быстро находить оптимальные пути в графах, с учетом только прямоленейных маршрутов, однако его эффективность снижается, когда необходимо учитывать сложные траектории, в том числе те, которые имеют форму зигзагообразных путей, характерных для навигации в реальных или ограниченных пространствах. В условиях, когда движущийся объект должен преодолевать пространство с препятствиями или работать в ограниченных и сильно структурированных средах, прямолинейные пути могут быть невозможными, и возникает необходимость в более гибком планировании маршрута [11]. Алгоритм A* строит зигзагообразную траекторию, поскольку он стремится найти наиболее короткие и прямые отрезки, что определяется следующим выражением:

$$f(n) = g(n) + h(n), \tag{17}$$

где $g(n)$ – длина пути до текущей точки n, а $h(n)$ – оценка расстояния до цели. Тогда траектория состоит из множества коротких отрезков $d(P_i, P_{i+1})$, сумма которых задаёт общую длину и вычисляется следующим образом:

$$L_t = \sum_{i=1}^{N-1} d(P_i, P_{i+1}), \tag{18}$$

Дискретизация пространства приводит к формированию зигзагообразных маршрутов, так как фиксированные узлы

ограничивают выбор точек, соединяя их по принципу ближайшего соседства, образую острые углы между последовательными отрезками. Они вычисляются на основе следующего выражения:

$$\cos(\theta_i) = \frac{(P_i - P_{i-1}) \cdot (P_{i+1} - P_i)}{\|P_i - P_{i-1}\| \cdot \|P_{i+1} - P_i\|}, \tag{19}$$

Такой маршрут, построенный с помощью алгоритма A* приведен на рисунке 2.

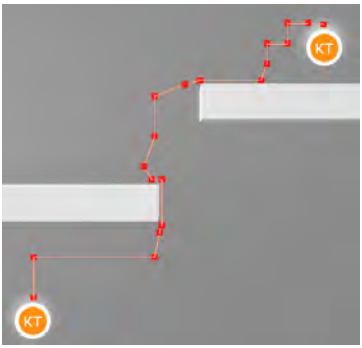

Рис. 2. Построение маршрута на основе алгоритма A*

Эти «изгибы» увеличивают суммарную длину маршрута, снижают точность, а также повышают вычислительные затраты и время движения автономных устройств, за счет выполнения дополнительных маневров, что отрицательно влияет на общую эффективность, особенно в контексте ограниченных ресурсов.

Для уменьшения числа острых углов применяется процедура сглаживание траектории, что достигается путём оптимизации отрезков пути, исключая избыточные точки. Итоговое решение (длина маршрута) включает сглаживание поворотов, что определяется следующим выражением:

$$L_s = \sum_{i=1}^{N-1} \|P_i - P_{i-1}\| + \beta \sum_{i=2}^{N-1} (1 - \cos(\theta_i)), \tag{20}$$

где первый компонент минимизирует длину, а второй сглаживает углы между отрезками.

С учетом этого в работе предлагается модифицированный алгоритм маршрутизации, реализующий процедуру сглаживания маршрута, за счет уменьшения количество избыточных точек на пути. При этом устраняются ненужные колебания и ошибки, что способствует оптимизации маршрута, повышению его точности, сокращению вычислительных затрат, а также снижается общая длина пути, уменьшается время обхода контрольных точек, что эффективно сказывается при обработке в сложных и динамически изменяющихся средах.

Процесс сглаживания траектории в алгоритмах маршрутизации, несмотря на повышение энергоэффективности автономного устройства, обладает рядом недостатков, которые необходимо учитывать при его реализации. Одним из ключевых недостатков является снижение точности маршрута: сглаживание упрощает траекторию, устраняя острые углы и зигзаги, что приводит к отклонению маршрута от заданных контрольных точек. Это критично в ситуациях, когда необходимо точное соблюдение траектории, например, для обхода препятствий или при движении в ограниченных пространствах. Для устранения данных недостатков в работе предлагается использовать адаптивные методы сглаживания.

Предложенный алгоритм отличается от существующих решений тем, что использует адаптивное сглаживание маршрута, динамически изменяя степень удаления промежуточных точек в зависимости от структуры окружающей среды и требований к точности движения. В отличие от стандартных методов, где точки удаляются по фиксированному критерию (например, углу между отрезками или расстоянию до соседних узлов), в предложенном алгоритме вводится весовая функция, учитывающая плотность контрольных точек, наличие препятствий и необходимую точность навигации. Что позволяет сохранять критически важные точки, обеспечивая надежность маршрута, и одновременно исключать избыточные узлы, сокращая вычислительные затраты.

Ключевой модификацией является введение механизма приоритетного сохранения контрольных точек, основанный на оценке значимости узлов. Для этого используются параметры: минимальная дистанция между сохранёнными точками, степень изменения угла маршрута и наличие внешних ограничений (стены, узкие коридоры, динамические препятствия). В сравнении с традиционными методами сглаживания, которые с одной стороны упрощают маршрут, приводя к его отклонению от заданных контрольных точек, с другой не обеспечивают достаточную эффективность в устранении избыточных

узлов. Данный метод актуален для автономных устройств, функционирующих в условиях сложных динамически изменяющихся сред, где требуется учитывать вариативность топологии и пространственные ограничения.

Реализация модифицированного алгоритма построения матрицы на основе контрольных точек и поиска оптимальных путей между ними с процедурой сглаживания реализован на Псевдокоде 1.

```
Алгоритм построения матрицы маршрутов
Входные данные: P - множество контрольных точек {P_1, P_2, ..., P_N}, где каждая точка имеет координаты в 3D-пространстве.
Выходные данные: M - матрица маршрутов, где M[i][j] содержит путь и его длину между P_i и P_j.
Шаг 1. Инициализация:
  Создать пустую матрицу маршрутов M размером N × N.
  Для всех i, j: установить M[i][j] как пустой маршрут.
Шаг 2. Формирование маршрутов:
  Для каждого i от 1 до N:
    Для каждого j от i+1 до N:
      маршрут = Найти маршрут (P_i, P_j)
      длина = Рассчитать длину (маршрут)
      M[i][j] = (маршрут, длина)
      M[j][i] = Инвертировать (маршрут, длина)
Шаг 3. Поиск маршрута (P_start, P_end):
  Если существует прямой путь между P_start и P_end:
    вернуть {P_start, P_end}
  Иначе:
    Использовать алгоритм поиска пути (например, A*) для нахождения оптимального маршрута.
    вернуть список точек маршрута.
Шаг 4. Вычисление длины маршрута (маршрут):
  длина = 0
  Для каждого i от 1 до длины (маршрут) - 1:
    длина += Расстояние(маршрут[i], маршрут[i+1])
  вернуть длина
Шаг 5. Инверсия маршрута (маршрут, длина):
  инвертированный_маршрут = Обратить последовательность (маршрут)
  вернуть (инвертированный_маршрут, длина)
Шаг 6. Оптимизация маршрутов:
  Для каждого маршрута в M:
    удалить избыточные точки, не влияющие на траекторию;
    минимизировать количество резких поворотов;
    обновить длину маршрута.
Шаг 7. Выход:
  Вернуть матрицу маршрутов M.
```

Псевдокод 1. Реализация модуля построение матрицы маршрутов

Рассмотрим работу этого алгоритма по шагам.

- Инициализация. Создается пустая матрица для хранения информации о маршрутах между всеми возможными парами точек. Каждая ячейка матрицы содержит данные о маршруте, включая последовательность точек и его длину. Что описывается на основе формулы 1.
- Получение входных данных. Система получает набор контрольных точек, которые необходимо связать. Эти точки представляют собой координаты в трёхмерном пространстве.
- Поиск маршрутов. Для каждой пары точек выполняется соединение. Это может быть прямой путь или более сложная траектория, в зависимости от препятствий или ограничений.
- Сохранение маршрутов. После нахождения соединения между двумя точками, данные о маршруте (последовательность точек на пути и длина маршрута) сохраняются в соответствующей ячейке матрицы.
- Обратные маршруты. Сохраняются инвертированные маршруты (обратные пути), что позволяет использовать симметричные данные, и при необходимости вернуться из конечной точки в начальную.
- Формирование матрицы решений. Заполняется итоговая матрица маршрутов, которая содержит всю необходимую информацию о соединениях между точками. Эта матрица используется для дальнейшего анализа, оптимизации или планирования траекторий в контексте навигации автономных устройств.

На рисунке 3 представлен маршрут после реализации механизма сглаживания, который уменьшает количество острых углов и зигзагов оптимизируя траекторию. Это позволяет сформировать эффективный маршрут, сокращая время движения и повышая точность при обходе препятствий.

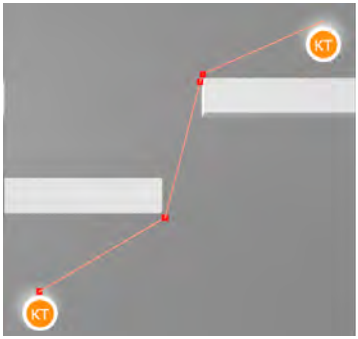

Рис. 3. Маршрут после применения механизма сглаживания

Таким образом, применение механизма сглаживания позволяет улучшить характеристики маршрута, обеспечивая более плавное и эффективное движение автономных устройств, что особенно важно в условиях сложной топологии и динамически изменяющейся сред.

5. Модифицированный муравьиный алгоритм. Для реализации всей маршрутизации в работе перелагается модифицированный метаэвристический муравьиный алгоритм. Алгоритм маршрутизации основывается на принципах поведения муравьиной колонии, которая при поиске пищи оставляет феромоны на рёбрах графа, тем самым формируя различные возможные маршруты. По мере того, как феромоны испаряются, более длинные и менее эффективные пути становятся менее выраженными, в то время как более короткие маршруты, наоборот, усиливаются, так как на них остаётся больше феромона [12 – 15]. Этот процесс позволяет моделировать поиск оптимального маршрута, аналогично поведению муравьёв в естественной среде [14].

Как известно одной из проблем при реализации муравьиных алгоритмов является большое количество параметров с широкими диапазонами значений, что усложняет настройку и адаптацию алгоритма для конкретных задач [15 – 17].

Для решения этой проблемы предлагается следующая модификация, заключающаяся в реализации двух механизмов. Это динамическая регулировка количества феромона, оставляемого муравьями, зависящая от качества пройденного маршрута, что позволяет избежать чрезмерного накопления феромона на менее удачных путях и способствует более равномерному исследованию решений. При этом процесс обновление количества феромона на ребре e_{ij} вычисляется на основе следующего выражения [1, 17]:

$$\tau_{ij}(t+1) = (1-\rho) \cdot \tau_{ij}(t) + \Delta\tau_{ij}(t), \quad (21)$$

где $\tau_{ij}(t)$ – количество феромона на ребре e_{ij} в момент времени t, ρ – коэффициент испарения феромона, а $\Delta\tau_{ij}(t)$ – изменение феромона, которое зависит от пути, выбранного муравья в текущем цикле.

В отличие от классического подхода, в данном алгоритме учитывается адаптивное изменение феромона в зависимости от качества маршрута, что позволяет избежать ранней фиксации на локальных решениях.

Дополнительно используется механизм введения «элитных муравьев», которые усиливают феромонные следы на наиболее перспективных маршрутах, ускоряя процесс сходимости и направляя

агентов к оптимальным траекториям. Эти изменения устраняют недостатки классического алгоритма, такие как медленная сходимость и склонность к застреванию в локальных минимумах [1].

Элитные муравьи оставляют больше феромона на наиболее перспективных маршрутах. Обновление феромона для элитных муравьёв описывается следующим образом:

$$\tau_{ij}{}^{ilite}(t+1) = (1-\rho) \cdot \tau_{ij}(t) + \Delta\tau_{ij}{}^{ilite}(t), \quad (22)$$

где $\Delta\tau_{ij}{}^{ilite}(t)$ – это дополнительное количество феромона, оставляемое элитными муравьями. Этот параметр задается как более высокое значение по отношению к $\Delta\tau_{ij}(t)$ для обычных муравьёв:

$$\tau_{ij}{}^{ilite}(t+1) = \varepsilon \cdot \Delta\tau_{ij}(t), \quad (23)$$

где $\varepsilon > 1$ – коэффициент усиления, который задает уровень феромона элитных муравьев.

Предложенный механизм элитных муравьев отличается от стандартного тем, что использует динамическую настройку коэффициента усиления ε в зависимости от текущего состояния поиска. Что позволяет достичь баланса между усилением лучших маршрутов и сохранением разнообразия решений. В классическом алгоритме муравьиного поиска обновление феромона осуществляется равномерно для всех муравьев, что способствует преждевременной фиксации на неоптимальных маршрутах. В предложенной модификации данный процесс регулируется динамически с учетом эффективности пройденного пути, что повышает устойчивость поиска и ускоряет его сходимость. Кроме того, в отличие от стандартного подхода, предложенный алгоритм адаптивно регулирует влияние элитных муравьев посредством коэффициента ε, предотвращая избыточную концентрацию феромона на отдельных маршрутах и обеспечивая сбалансированное исследование пространства решений.

Данный механизм способствует ускорению сходимости к оптимальным траекториям, снижая вероятность попадания в локальные «ямы». Он фокусирует внимание на наиболее эффективных путях, увеличивая интенсивность феромонного следа, что повышает вероятность выбора этих маршрутов на последующих итерациях. Таким образом, введение элитных муравьёв позволяет уменьшить влияние случайности и ускоряет процесс нахождения эффективного маршрута, устраняя недостатки классического муравьиного алгоритма.

Также элитные муравьи не только увеличивают интенсивность выделения феромонов, но и активно влияют на баланс глобального и локального поиска, адаптируя свою стратегию в зависимости от предыдущих итераций, что делает процесс нахождения оптимального маршрута более устойчивым и эффективным.

Модифицированный муравьиный алгоритм, в котором реализованы механизмы динамического обновления феромонов и использования элитных муравьёв представлен в виде Псевдокода 2.

Шаг 1. Инициализация параметров: задаём основные параметры алгоритма, включая количество муравьёв, число итераций, коэффициент испарения феромонов и влияния эвристики.

```
num_ants = 50
num_iterations = 100
evaporation_rate = 0.5
alpha = 1.0
beta = 2.0
elite_factor = 2.0
pheromones = [[0.1 for _ in range(len(M))] for _ in range(len(M))]
```

Шаг 2. Построение маршрутов: каждый муравей строит маршрут, выбирая следующий узел с учётом феромонов и эвристики.

```
def build_route():
    route, visited = [], set()
    current = random.randint(0, len(M) - 1)
    route.append(current)
    visited.add(current)
while len(visited) < len(M):
        next_node = select_next_node(current, visited)
        route.append(next_node)
        visited.add(next_node)
        current = next_node
    return route
def select_next_node(current, visited):
probabilities = [pheromones[current][neighbor]**alpha * (1/M[current][neighbor])**beta
                 for neighbor in range(len(M)) if neighbor not in visited]
    nodes = [neighbor for neighbor in range(len(M)) if neighbor not in visited]
    return random.choices(nodes, weights=probabilities)[0]
```

Шаг 3. Обновление феромонов: феромоны обновляются с учётом всех маршрутов и более сильным воздействием элитных муравьёв.

```
def update_pheromones(routes, best_route):
    for i in range(len(M)):
        for j in range(len(M)):
            pheromones[i][j] *= (1 - evaporation_rate)
    for route in routes:
        length = calculate_route_length(route)
        pheromone_value = 1.0 / length
        for i in range(len(route) - 1):
```

```
            a, b = route[i], route[i + 1]
            pheromones[a][b] += pheromone_value
            pheromones[b][a] += pheromone_value
    if best_route:
        elite_pheromone = elite_factor / calculate_route_length(best_route)
        for i in range(len(best_route) - 1):
            a, b = best_route[i], best_route[i + 1]
            pheromones[a][b] += elite_pheromone
            pheromones[b][a] += elite_pheromone
Шаг 4. Обновление феромонов: феромоны обновляются с учётом всех маршрутов и более сильным воздействием элитных муравьёв.
best_route, best_length = None, float("inf")
for iteration in range(num_iterations):
    routes = [build_route() for _ in range(num_ants)]
for route in routes:
        length = calculate_route_length(route)
        if length < best_length:
            best_length = length
            best_route = route
update_pheromones(routes, best_route)
Шаг 5. Результат: выводим лучший маршрут и его длину.
print("Лучший маршрут:", best_route)
print("Длина маршрута:", best_length)
```

Псевдокод 2. Реализация модифицированного муравьиного алгоритма

Рассмотрим работу модифицированного муравьиного алгоритма по шагам.

- Инициализация параметров. На первом этапе задаются начальные параметры алгоритма, включая количество муравьёв, коэффициенты испарения феромона и максимальное число итераций. Также инициализируются данные об уровне феромонах на рёбрах графа.
- Построение и обновление уровня феромонов. Количество феромона на рёбрах зависит от качества маршрута. Больше феромона остаётся на коротких, эффективных путях, а менее эффективных путях происходит испарение феромонов с течением времени.
- Использование «элитных муравьёв». Для ускорения сходимости алгоритма и предотвращения попадания в локальные «ямы» вводится механизм элитных муравьёв. Эти муравьи оставляют большее количество феромона на наиболее перспективных маршрутах, что позволяет быстрее направить поиск к искомому решению. Это улучшает скорость сходимости и направляет муравьев к более эффективным траекториям.
- Основной цикл и итерации. Выполняется основной цикл, в котором муравьи начинают своё движение от стартовой точки,

выбирая пути на основе вероятностей, пропорциональных количеству феромона на рёбрах, что вычисляется на основе следящей формулы [1, 11]:

$$P_{ij} = \frac{\tau_{ij}^{a} \cdot \mu_{ij}^{\beta}}{\sum_{K \in N_i} \tau_{ij}^{a} \cdot \mu_{ij}^{\beta}}, \quad (24)$$

где τ_{ij}^{a} – количество феромона на ребре e_{ij}, μ_{ij}^{β} – эвристическая информация (например, обратная длина пути), N_i – множество соседей текущего узла, а a и β – параметры, определяющие количество феромона и эвристической информации.

- Постобработка и обновление количества феромонов. После завершения всех итераций, выполняется обновление феромонов не рёбрах, по которым прошли муравьи, с учётом элитных муравьев. Обновление количества феромона позволяет зафиксировать наилучшие пути [1, 11].
- Оценка и завершение работы. Продолжается процесс до достижения заданного числа итераций или критерия останова. В конце работы выбирается оптимальный маршрут.

Модифицированный муравьиный алгоритм реализует механизмы динамического обновления феромонов и использования элитных муравьёв для поиска оптимальных маршрутов в динамических и сложных средах. Данный алгоритм повышает эффективность поиска квазиоптимальных решений за счет быстрой и устойчивой сходимости.

Таким образом, модифицированный муравьиный алгоритм не только использует элитных муравьев и динамическое обновление феромонов, но и обеспечивает баланс между локальным и глобальным поиском, предотвращая преждевременную сходимость алгоритма.

6. Компьютерное моделирование. Данный подход к маршрутизации автономных устройств в трёхмерном пространстве был реализован в среде компьютерного моделирования Unreal Engine 5 (UE 5) [17 – 19]. На рисунке 4 приведено октодерево рабочего пространства, представленного в виде подразделяющейся кубической структуры.

Здесь в качестве модели используется ветрогенераторный парк, состояние которого анализируется автономным устройством, управляемым предложенным методом. Этот подход иллюстрирует одну из потенциальных областей применения разработанной системы.

Рис. 4. Октодерево рабочего пространства

На рисунке 5 изображен полный граф всех возможных маршрутов.

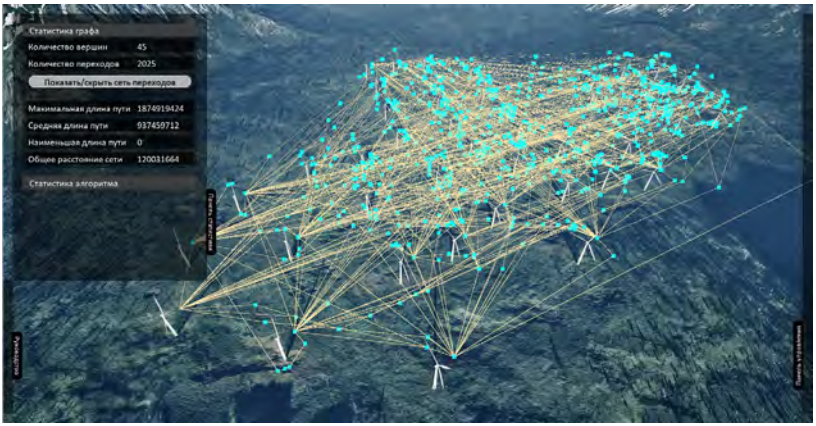

Рис. 5. Полный граф всех возможных маршрутов

Отметим, что вблизи объектов, структура дерева делится на максимальное количество уровней, что позволяет более точно моделировать сложные геометрические особенности и препятствия моделируемого ландшафта.

На рисунке 6 показан результат реализации модифицированного муравьиного алгоритма при построении кратчайшего маршрута, интерпретированного в виде точек в пространстве.

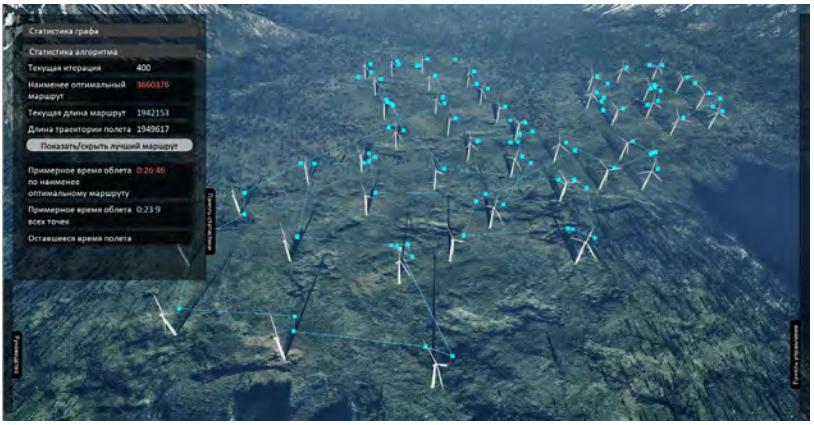

Рис. 6. Кратчайший маршрут (квазиоптимальное решение)

В результате проведенного компьютерного моделирования и предложенного подхода маршрутизации автономных устройств в 3D-пространстве с использованием Unreal Engine 5 продемонстрировано высокое качество реализации и производительности. Моделирование в виртуальной среде, включая использование октодерева и реализации модифицированных алгоритмов, показало точность в обработке сложных геометрических объектов при построении квазиоптимальных траекторий, что подтверждает эффективность предложенного подхода при решении задачи маршрутизации.

7. Вычислительный эксперимент. Для подтверждения эффективности и качества предложенного подхода, проведена серия экспериментальных исследований. В качестве аналога был рассмотрен классический муравьиный алгоритм.

По результатам предварительного анализа исходного графа маршрутизации было найдено эталонное значение целевой функции, которое составило 1,500,000 (в единицах Unreal Engine, где одна UE единица соответствует одному сантиметру) [17 – 20].

Для экспериментальной проверки предложенного подхода использовались сгенерированные случайные графы. Проведена серия тестов с целью оценки изменения параметров сходимости и качества маршрутов.

Оценка качества маршрутов Q играет ключевую роль в оценке эффективности алгоритма маршрутизации, поскольку она позволяет определить, насколько найденный маршрут приближен к оптимальному решению. Что рассчитывается по следующей формуле [1, 19]:

$$Q = \frac{D_{opt} - D_{current}}{D_{opt}} \cdot 100\%, \quad (25)$$

где D_{opt} – эталонное значение целевой функции, а $D_{current}$ – текущий результат, полученный с использованием алгоритма.

В качестве базового использовался классический муравьиный алгоритм с настройками, включающими следующие параметры: количество муравьёв (50), число итераций (100), коэффициент испарения феромонов (0.5), влияние феромонов (α = 1.0) и влияние эвристики (β = 2.0) [1]. Эти параметры были выбраны на основе предварительных экспериментов, направленных на балансировку качества решения и вычислительных затрат.

Результаты экспериментов, представленные на графике (рисунок 7), демонстрируют зависимость качества решения от времени выполнения различных алгоритмов. Что позволяет проанализировать эффективность предложенного подхода и сравнить его с классическими методами маршрутизации.

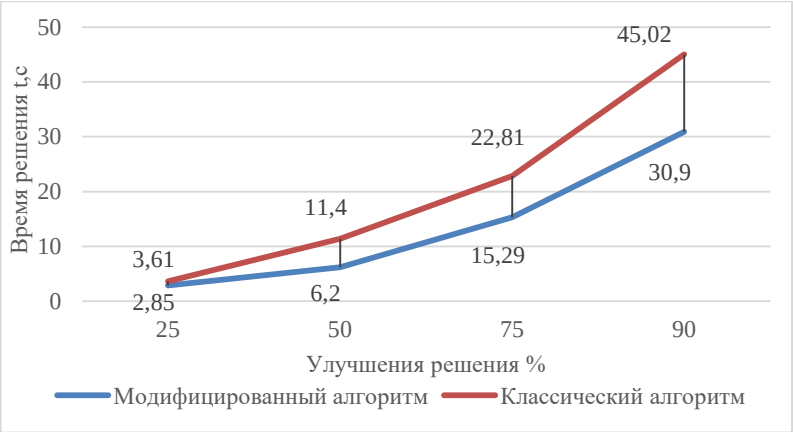

Рис. 7. График зависимости качества решения от времени выполнения алгоритмов

На рисунке 8 представлена зависимость качества решения от количества вершин графа (ветрогенераторов). График демонстрирует, как увеличивается эффективность алгоритмов с ростом числа объектов, что подчеркивает улучшение качества решения при расширении задачи.

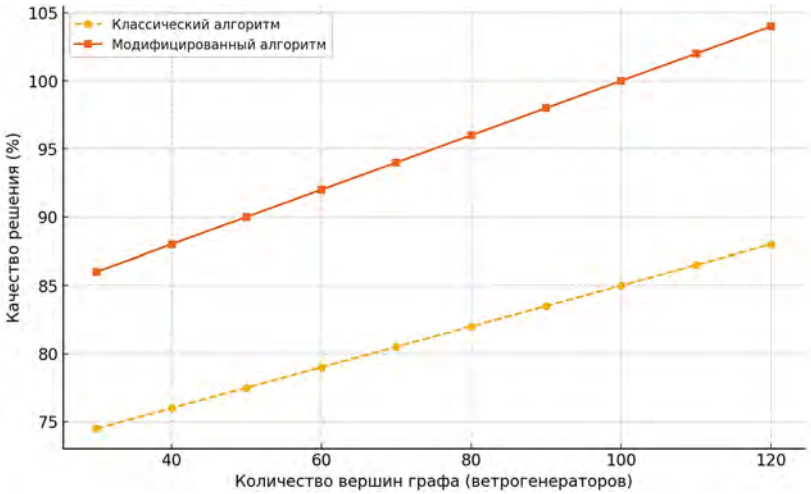

Рис. 8. График зависимости качества решения от количества вершин графа (ветрогенераторов)

При увеличении числа вершин в графе растет и вычислительная сложность задачи. Однако применение модификаций муравьиного алгоритма, таких как динамическая корректировка уровня феромонов и использование элитных муравьев, позволяет сохранять высокую эффективность и адаптироваться к динамическим изменениям в пространстве поиска, что позволяет улучшить результаты маршрутизации в условиях динамических и сложных средах.

Модифицированный муравьиный алгоритм продемонстрировал лучшие результаты, достигая 95% качества решений при 120 вершинах, в то время как классический алгоритм ограничивается на 90%. Предложенная модификация алгоритма, позволяет улучшить эффективность поиска, сокращая время для достижения 90% качества с 45,02 до 30,9 секунды. Несмотря на увеличение времени одной итерации из-за механизма «элитных муравьёв» и дополнительных операций с матрицей феромонного следа, ускорение сходимости за счет усиления феромонного влияния на перспективные маршруты компенсирует этот эффект. Таким образом, предложенный подход позволил улучшить качество и эффективность маршрутизации в среднем на 10-15%.

8. Заключение. В исследовании рассмотрена проблема маршрутизации автономных устройств в трёхмерном пространстве, которая представляет собой одну из ключевых задач в области интеллектуального управления. Актуальность данной проблемы

обусловлена необходимостью эффективного решения задач оптимизации траекторий в условиях высокой степени свободы, сложной топологии и динамично изменяющейся среды. Особенности трёхмерной среды, включающие высокую степень свободы, сложную топологическую организацию и динамически изменяющиеся параметры, существенно усложняют процесс построения эффективных и безопасных маршрутов. В связи с этим, разработанный подход к решению задачи маршрутизации интегрирует высокоуровневое моделирование рабочего пространства с использованием иерархических структур данных, таких как октодеревья. Он основан на применении метаэвристических алгоритмов оптимизации, которые обеспечивают высокую адаптивность к динамическим изменениям и вычислительную эффективность, необходимую для решения задачи маршрутизации в условиях сложной топологии. Данный подход к маршрутизации автономных устройств в трёхмерном пространстве был реализован в среде компьютерного моделирования Unreal Engine 5.

Для подтверждения эффективности предложенного подхода была проведена серия вычислительных экспериментов, которая подтвердила эффективность разработанного алгоритма маршрутизации, включая сокращение времени вычислений и повышение энергоэффективности автономных устройств в среднем на 10-15%.

Для задачи маршрутизации в трёхмерном пространстве оптимальным решением является использование модифицированного муравьиного алгоритма, который продемонстрировал высокую производительность и энергоэффективность.

Перспективы дальнейших исследований включают расширение возможностей системы, включая интеграцию пользовательских алгоритмов маршрутизации без необходимости перекомпиляции исходного кода, внедрение асинхронной обработки данных и оптимизацию времени выполнения итераций. Предполагается, что такие улучшения позволят существенно повысить адаптивность и вычислительную эффективность системы, сохранив её работоспособность в заданных режимах.

Литература

1. Курейчик В.В., Гладков Л.А., Кравченко Ю.А., Родзин С.И. Интеллектуальные системы: модели и методы метаэвристической оптимизации: Монография // Чебоксары: ООО «Издательский дом «Среда», 2024. 228 с. DOI: 10.31483/a-10639.
2. Даринцев О.В., Мигранов А.Б. Аналитический обзор подходов к распределению задач в группах мобильных роботов на основе технологий мягких вычислений //

Информатика и автоматизация. 2022. Т. 21. № 4. С. 729–757. DOI: 10.15622/ia.21.4.4.

3. Курейчик В.В., Родзин С.И. Вычислительные модели эволюционных и роевых биоэвристик (обзор) // Информационные технологии. 2021. Т. 27. № 10. С. 507–520. DOI: 10.17587/it.27.507-520.
4. Перепелкин Д.А., Иванчикова М.А., Нгуен В.Т. Нейросетевая многопутевая маршрутизация в программно-конфигурируемых сетях на основе генетического алгоритма // Информационные технологии. 2023. Т. 29. № 12. С. 622–629. DOI: 10.17587/it.29.622-629.
5. Казакова Е.М. Применение метода роя частиц в задачах оптимизации // Известия Кабардино-Балкарского научного центра РАН. 2022. № 5(109). С. 48–57. DOI: 10.35330/1991-6639-2022-5-109-48-57.
6. Казаков К.А., Семенов В.А. Обзор современных методов планирования пути // Тр. ИСП РАН. 2016. Т. 28(4). С. 241–294. DOI: 10.15514/ISPRAS-2016-28(4)-14.
7. Гладков Л.А., Гладкова Н.В. Эволюционирующие многоагентные системы и эволюционное проектирование // Известия ЮФУ. Технические науки. 2020. № 4(214). С. 48–59. DOI: 10.18522/2311-3103-2020-4-48-59.
8. Рачков Т.И., Кузьмина И.А. Мета-эвристический алгоритм децентрализованного управления группой // Математические методы в технологиях и технике. 2021. № 3. С. 96–99. DOI: 10.52348/2712-8873_MMTT_2021_3_96.
9. Костин А.С., Майоров Н.Н. Исследование моделей и методов маршрутизации и практического выполнения автономного движения беспилотными транспортными системами для доставки грузов // Вестник государственного университета морского и речного флота им. адмирала С.О. Макарова. 2023. Т. 15. № 3. С. 524–536. DOI: 10.21821/2309-5180-2023-15-3-524-536.
10. Акопов А.С., Бекларян Л.А., Бекларян А.Л. Оптимизация характеристик интеллектуальной транспортной системы с использованием генетического алгоритма вещественного кодирования на основе адаптивной мутации // Информационные технологии. 2023. Т. 29. № 3. С. 115–125. DOI: 10.17587/it.29.115-125.
11. Карпенко А.П. Эволюционные операторы популяционных алгоритмов глобальной оптимизации. Опыт систематизации // Математика и математическое моделирование. 2018. № 1. С. 59–89. DOI: 10.24108/mathm.0118.0000103.
12. Медведев М.Ю., Костюков В.А., Пшихопов В.Х. Метод оптимизации траектории мобильного робота в поле источников-репеллеров // Информатика и автоматизация. 2021. Т. 20. № 3. С. 690–726. DOI: 10.15622/ia.2021.3.7.
13. Литвиненко А.М., Кудрявцев Г.В., Ибрагимов М.У.У. Исследование адаптивной системы управления электроэнергетическим комплексом с ветроэлектрогенератором // Вести высших учебных заведений Черноземья. 2021. № 3(65). С. 10–17. DOI: 10.53015/18159958_2021_3_10.
14. Курочкин А.Г., Титенко Е.А. Модифицированный алгоритм сглаживания точек маршрута // Известия Юго-Западного государственного университета. 2016. № 5(68). С. 43–51.
15. Шмалько Е.Ю., Румянцев Ю.А., Байназаров Р.Р., Ямшанов К.Л. Идентификация нейросетевой модели робота для решения задачи оптимального управления // Информатика и автоматизация. 2021. Т. 20. № 6. С. 1254–1278. DOI: 10.15622/ia.20.6.3.
16. Ургапов В.А. Компьютерное моделирование основных типов движения мультикоптера в трёхмерном пространстве // Вестник Совета молодых ученых Рязанского государственного агротехнологического университета имени П.А. Костычева. 2016. № 1(2). С. 216–220.

17. Жарков С.Н. Стохастическое формирование проактивного множества при кластеризации в мобильных беспроводных сенсорных сетях // T-Comm: Телекоммуникации и транспорт. 2013. Т. 7. № 5. С. 29–34.
18. Шилов Н.Г., Пономарев А.В., Смирнов А.В. Анализ методов онтолого-ориентированного нейро-символического интеллекта при коллаборативной поддержке принятия решений // Информатика и автоматизация. 2023. Т. 22. № 3. С. 576–615. DOI: 10.15622/ia.22.3.4.
19. Danilchenko V.I., Danilchenko E.V., Kureichik V.M. Application of Genetic Algorithms in Solving the Problem of Placing Elements on a Crystal Taking into Account the Criterion of the Maximum Number of Linear Segments // Proceedings of the Fifth International Scientific Conference “Intelligent Information Technologies for Industry” (IITI’21). Lecture Notes in Networks and Systems. 2022. vol. 330. pp. 276–284. DOI: 10.1007/978-3-030-87178-9_28.
20. Валькман Ю.Р., Тарасов В.Б. От онтологий проектирования к когнитивной семиотике // Онтология проектирования. 2018. Т. 8. № 1(27). С. 8–34. DOI: 10.18287/2223-9537-2018-8-1-8-34.

Курейчик Владимир Викторович — д-р техн. наук, профессор, заведующий кафедрой, САПР им. В.М. Курейчика, ИКТИБ, Южный Федеральный университет. Область научных интересов: компьютерное моделирование, автоматизация проектирования. Число научных публикаций — 411. vkur@sfedu.ru; улица Энгельса, 1, к. Г-435, 347928, Таганрог, Россия; р.т.: +7(863)438-3451.

Данильченко Владислав Иванович — канд. техн. наук, доцент кафедры, САПР им. В.М. Курейчика, ИКТИБ, Южный Федеральный университет. Область научных интересов: компьютерное моделирование, автоматизация проектирования. Число научных публикаций — 58. vdanilchenko@sfedu.ru; улица Энгельса, 1, к. Г-435, 347928, Таганрог, Россия; р.т.: +7(863)437-1651.

Данильченко Евгения Владимировна — ассистент кафедры, САПР им. В.М. Курейчика, ИКТИБ, Южный Федеральный университет. Область научных интересов: автоматика, вычислительная техника. Число научных публикаций — 20. lipkina@sfedu.ru; улица Энгельса, 1, к. Г-435, 347928, Таганрог, Россия; р.т.: +7(863)437-1651.

Поддержка исследований. Исследование выполнено за счет гранта Российского научного фонда № 24-71-00035, https://rscf.ru/project/24-71-00035/ в Южном федеральном университете.

DOI 10.15622/ia.24.2.5

V. KUREYCHIK, V. DANILCHENKO, E. DANILCHENKO

ROUTING OF AUTONOMOUS DEVICES IN THREE-DIMENSIONAL SPACE

Kureychik V., Danilchenko V., Danilchenko E. **Routing of Autonomous Devices in Three-Dimensional Space.**

Abstract. The article addresses the problem of routing autonomous devices in three-dimensional space, which is a relevant task for intelligent control. The three-dimensional space is characterized by a high degree of freedom, complex topology, and dynamic environmental changes, which significantly complicate the task of effective trajectory planning. The development of routing methods that ensure safety, energy efficiency, and computational efficiency is crucial for improving the performance of autonomous systems. The paper presents a comprehensive routing system based on a hybrid approach that combines high-level modeling of the working space with metaheuristic optimization methods. Hierarchical data structures, such as octrees, are used to represent the three-dimensional environment, providing compactness and flexibility for spatial models. These models are transformed into graph structures, allowing the routing problem to be described as an optimization problem on graphs. A modified metaheuristic ant colony optimization algorithm, belonging to the class of swarm optimization methods, is proposed. The algorithm is designed to build safe and energy-efficient routes, as well as to solve problems related to finding the shortest Hamiltonian cycles and dynamically reconfiguring routes in a changing external environment. The paper presents the results of computational experiments, including algorithm testing in three-dimensional space and a comparative analysis with other routing algorithms. The computational experiment confirmed the effectiveness of the developed routing algorithm, including reduced computation time and improved energy efficiency of autonomous devices. The prospects for further research include integrating the proposed system into a wide range of applications for autonomous devices aimed at optimizing control processes and enhancing performance in a dynamically changing external environment. It is worth noting that the developed algorithm can be adapted to solve complex tasks where routing and wind generator placement on a plane are interrelated. The placement problem is directly connected to route construction for servicing these objects, which requires a comprehensive approach for an efficient solution. This will be part of a decision support system designed for the planning and servicing of wind power complexes, ensuring their effective operation and resource management.

Keywords: metaheuristic algorithm, ant colony optimization, graph-based mathematical models, routing, three-dimensional space modeling, energy systems.

References

1. Kureichik V.V., Gladkov L.A., Kravchenko Yu.A., Rodzin S.I. Intellektual'nye sistemy: modeli i metody metajevristicheskoj optimizacii: Monografija [Intelligent Systems: Models and Methods of Metaheuristic Optimization. Monograph]. Cheboksary: Publishing House «Sreda», 2024. 228 p. DOI: 10.31483/a-10639. (In Russ.).
2. Darintsev O.V., Migranov A.B. [Analytical Review of Approaches to Task Allocation in Groups of Mobile Robots Based on Soft Computing Technologies]. Informatika i avtomatizacija – Informatics and Automation. 2022. vol. 21. no. 4. pp. 729–757. DOI: 10.15622/ia.21.4.4. (In Russ.).

3. Kureichik V.V., Rodzin S.I. [Computational Models of Evolutionary and Swarm Bioheuristics: A Review]. Informacionnye tehnologii – Information Technologies. 2021. vol. 27. no. 10. pp. 507–520. DOI: 10.17587/it.27.507-520. (In Russ.).
4. Perepelkin D.A., Ivanchikova M.A., Nguyen V.T. [Neural Network Multi-Path Routing in Software-Defined Networks Based on Genetic Algorithm]. Informacionnye tehnologii – Information Technologies. 2023. vol. 29. no. 12. pp. 622–629. DOI: 10.17587/it.29.622-629. (In Russ.).
5. Kazakova E.M. [Application of Particle Swarm Optimization in Optimization Problems]. Izvestija Kabardino-Balkarskogo nauchnogo centra RAN – Proceedings of the Kabardino-Balkar Scientific Center of the Russian Academy of Sciences. 2022. no. 5(109). pp. 48–57. DOI: 10.35330/1991-6639-2022-5-109-48-57. (In Russ.).
6. Kazakov K.A., Semenov V.A. [An overview of modern methods for motion planning]. Trudy ISP RAN – Труды ИСП РАН. 2016. vol. 28(4). pp. 241–294. DOI: 10.15514/ISPRAS-2016-28(4)-14. (In Russ.).
7. Gladkov L.A., Gladkova N.V. [Evolving Multi-Agent Systems and Evolutionary Design]. Izvestija JuFU. Tehnicheskie nauki – Proceedings of Southern Federal University. Technical Sciences. 2020. no. 4(214). pp. 48–59. DOI: 10.18522/2311-3103-2020-4-48-59. (In Russ.).
8. Rachkov T.I., Kuzmina I.A. [Metaheuristic Algorithm for Decentralized Group Control]. Matematicheskie metody v tehnologijah i tehnike – Mathematical Methods in Technologies and Engineering. 2021. no. 3. pp. 96–99. DOI: 10.52348/2712-8873_MMTT_2021_3_96. (In Russ.).
9. Kostin A.S., Maiorov N.N. [Study of Models and Routing Methods for Autonomous Movement of Unmanned Transport Systems for Freight Delivery]. Vestnik gosudarstvennogo universiteta morskogo i rechnogo flota im. admirala S.O. Makarova – Bulletin of the Admiral S.O. Makarov State University of Maritime and Inland Shipping. 2023. vol. 15. no. 3. pp. 524–536. DOI: 10.21821/2309-5180-2023-15-3-524-536. (In Russ.).
10. Akopov A.S., Beklaryan L.A., Beklaryan A.L. [Optimization of Intelligent Transportation System Characteristics Using Genetic Algorithm with Real-Valued Encoding Based on Adaptive Mutation]. Informacionnye tehnologii – Information Technologies. 2023. vol. 29. no. 3. pp. 115–125. DOI: 10.17587/it.29.115-125. (In Russ.).
11. Karpenko A.P. [Evolutionary Operators of Population Algorithms for Global Optimization: A Systematic Review]. Matematika i matematicheskoe modelirovanie – Mathematics and Mathematical Modeling. 2018. no. 1. pp. 59–89. DOI: 10.24108/mathm.0118.0000103. (In Russ.).
12. Kostyukov V.A., Medvedev M.Yu., Pshikhopov V.Kh. [Trajectory Optimization Method for Mobile Robots in the Field of Repeller Sources]. Informatika i avtomatizacija – Informatics and Automation. 2021. vol. 20. no. 3. pp. 690–726. DOI: 10.15622/ia.2021.3.7. (In Russ.).
13. Litvinenko A.M., Kudryavtsev G.V., Ibrahimov M.U. [Study of an Adaptive Control System for an Electric Power Complex with Wind Turbine Generators]. Vesti vysshih uchebnyh zavedenij Chernozem'ja – Herald of Higher Educational Institutions of the Black Earth Region. 2021. no. 3(65). pp. 10–17. DOI: 10.53015/18159958_2021_3_10. (In Russ.).
14. Kurochkin A.G., Titenko E.A. [Modified Algorithm for Smoothing Route Points]. Izvestija Jugo-Zapadnogo gosudarstvennogo universiteta – Proceedings of the Southwest State University. 2016. no. 5(68). pp. 43–51. (In Russ.).
15. Shmalko E.Yu., Rumyantsev Yu.A., Baynazarov R.R., Yamshanov K.L. [Identification of a Neural Network Model for Optimal Control of a Robot].

Informatika i avtomatizacija – Informatics and Automation. 2021. vol. 20. no. 6. pp. 1254–1278. DOI: 10.15622/ia.20.6.3. (In Russ.).
16. Urgapov V.A. [Computer Modeling of Basic Multicopter Motion Types in Three-Dimensional Space]. Vestnik Soveta molodyh uchenyh Rjazanskogo gosudarstvennogo agrotehnologicheskogo universiteta imeni P.A. Kostycheva – Herald of the Young Scientists Council of the Ryazan State Agrotechnological University named after P.A. Kostychev. 2016. no. 1(2). pp. 216–220. (In Russ.).
17. Zharkov S.N. [Stochastic Formation of a Proactive Set in Clustering of Mobile Wireless Sensor Networks]. T-Comm: Telekommunikacii i transport – T-Comm: Telecommunications and Transport. 2013. vol. 7. no. 5. pp. 29–34. (In Russ.).
18. Shilov N.G., Ponomaryov A.V., Smirnov A.V. [Analysis of Ontology-Oriented Neuro-Symbolic Intelligence Methods in Collaborative Decision-Making Support]. Informatika i avtomatizacija – Informatics and Automation. 2023. vol. 22. no. 3. pp. 576–615. DOI: 10.15622/ia.22.3.4. (In Russ.).
19. Danilchenko V.I., Danilchenko E.V., Kureichik V.M. Application of Genetic Algorithms in Solving the Problem of Placing Elements on a Crystal Taking into Account the Criterion of the Maximum Number of Linear Segments. Proceedings of the Fifth International Scientific Conference «Intelligent Information Technologies for Industry» (IITI'21). Lecture Notes in Networks and Systems. 2022. vol. 330. pp. 276–284. DOI: 10.1007/978-3-030-87178-9_28.
20. Valkman Yu.R., Tarasov V.B. [From Design Ontologies to Cognitive Semiotics]. Ontologija proektirovanija – Ontology of Design. 2018. vol. 8. no. 1(27). pp. 8–34. DOI: 10.18287/2223-9537-2018-8-1-8-34. (In Russ.).

Kureychik Vladimir — Ph.D., Dr.Sci., Professor, Head of the department, V.M. Kureychik CAD department, ICTIS, Southern Federal University. Research interests: computer modeling, design automation. The number of publications — 411. vkur@sfedu.ru; 1, cor. G-435, Engels St., 347928, Taganrog, Russia; office phone: +7(863)438-3451.

Danilchenko Vladislav — Ph.D., Associate professor of the department, V.M. Kureychik CAD department, ICTIS, Southern Federal University. Research interests: computer modeling, design automation. The number of publications — 58. vdanilchenko@sfedu.ru; 1, cor. G-435, Engels St., 347928, Taganrog, Russia; office phone: +7(863)437-1651.

Danilchenko Evgeniya — Assistant of the department, V.M. Kureychik CAD department, ICTIS, Southern Federal University. Research interests: automation, computer engineering. The number of publications — 20. lipkina@sfedu.ru; 1, cor. G-435, Engels St., 347928, Taganrog, Russia; office phone: +7(863)437-1651.

Acknowledgements. The research was funded by the Russian Science Foundation project No. 24-71-00035, https://rscf.ru/project/24-71-00035/ implemented by the Southern Federal University.