# МАТЕМАТИЧЕСКОЕ И ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ВЫЧИСЛИТЕЛЬНЫХ МАШИН, КОМПЛЕКСОВ И КОМПЬЮТЕРНЫХ СЕТЕЙ (ТЕХНИЧЕСКИЕ НАУКИ)

УДК 004.42, 681.51

## ПОСТРОЕНИЕ ПРОГРАММНОЙ ТРАЕКТОРИИ ДВИЖЕНИЯ БЕСПИЛОТНОГО НАЗЕМНОГО ТРАНСПОРТНОГО СРЕДСТВА[1]

*Статья получена редакцией 09.06.2019, в окончательной версии – 26.08.2019.*

***Егунов Виталий Алексеевич,*** Волгоградский государственный технический университет, 400005, Российская Федерация, г. Волгоград, пр. им. Ленина, 28,
кандидат технических наук, доцент, e-mail: vegunov@mail.ru
***Марков Алексей Евгеньевич***, Волгоградский государственный технический университет, 400005, Российская Федерация, г. Волгоград, пр. им. Ленина, 28,
магистрант, e-mail: markovalex95@gmail.com
***Скориков Андрей Викторович***, Волгоградский государственный технический университет, 400005, Российская Федерация, г. Волгоград, пр. им. Ленина, 28,
ассистент, e-mail: scorpion_energy@mail.ru
***Тарасов Павел Сергеевич***, Волгоградский государственный технический университет, 400005, Российская Федерация, г. Волгоград, пр. им. Ленина, 28,
ассистент, e-mail: tarasradio@mail.ru

Рассматриваются вопросы построения программной траектории движения, которые могут быть использованы в инициативных проектах по созданию беспилотных транспортных средств, реализуемых творческими коллективами Волгоградского государственного технического университета. Приводится авторский метод построения данной траектории, представляющий собой модификацию известного алгоритма прокладки маршрута для беспилотных наземных транспортных средств. Для описания траектории движения используются полиномы. С целью поиска коэффициентов полиномов, описывающих оптимальную траекторию, осуществляется решение системы линейных алгебраических уравнений. Применение данного метода позволяет оптимизировать построение траектории движения с учетом заданных ограничений. Представленные в статье решения были апробированы на реальной авторской модели беспилотного транспортного средства, оснащенной бортовым компьютером NVidia Jetson TX2. Она представляет собой неоднородную параллельную вычислительную систему, включающую в себя два 64-битных четырехядерных ARM-процессора и GPU Pascal.

**Ключевые слова:** беспилотный автомобиль, построение траектории, оптимальная траектория, метод планирования движения, система линейных алгебраических уравнений, объезд препятствий, алгоритм Дейкстры

## A SOFTWARE CONSTRUCTION OF THE UNMANNED GROUND VEHICLE TRAJECTORY

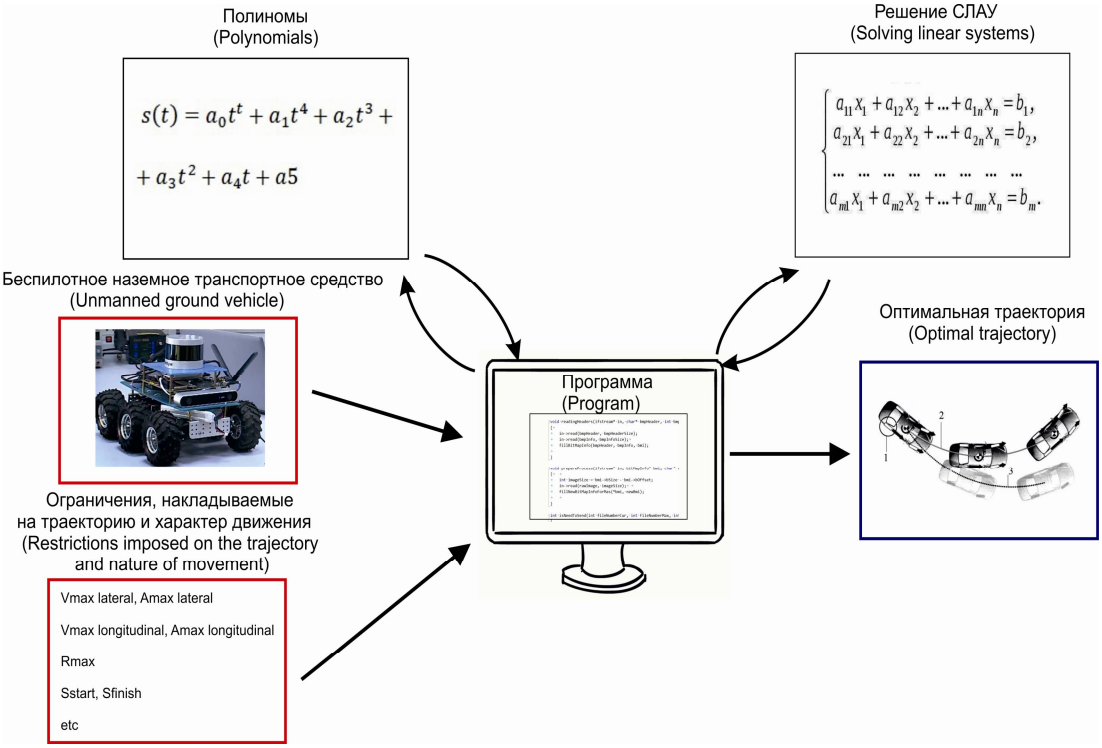
***Egunov Vitaly A.***, Volgograd State Technical University, 28 Lenin Ave., Volgograd, 400005, Russian Federation,
Cand. Sci. (Engineering), Associate Professor, e-mail: vegunov@mail.ru
***Markov Alexey E.***, Volgograd State Technical University, 28 Lenin Ave., Volgograd, 400005, Russian Federation,
student, e-mail: markovalex95@gmail.com
***Skorikov Andrey V.***, Volgograd State Technical University, 28 Lenin Ave., Volgograd, 400005, Russian Federation,
Assistant, e-mail: scorpion_energy@mail.ru
***Tarasov Pavel S.***, Volgograd State Technical University, 28 Lenin Ave., Volgograd, 400005, Russian Federation,
Assistant, e-mail: tarasradio@mail.ru

[1] Исследование выполнено при финансовой поддержке РФФИ и Администрации Волгоградской области в рамках научного проекта № 18-47-340010 р_а.

The questions of building a program trajectory that can be used in initiative projects to create unmanned vehicles implemented by the creative teams of the Volgograd State Technical University are considered. The author's method of constructing this trajectory, which is a modification of the well-known algorithm for routing unmanned ground vehicles. Polynomials are used to describe the trajectory of motion. To find the coefficients of polynomials describing the optimal trajectory, the system of linear algebraic equations is solved. The use of this method allows to optimize the construction of the trajectory taking into account the given restrictions.The presented solutions were tested on the real author's model of an unmanned vehicle equipped with an onboard computer NVidia Jetson TX2, which is a heterogeneous parallel computing system, which includes two 64-bit Quad-core ARM-processor and GPU Pascal.

**Keywords:** unmanned vehicle, construction of the trajectory, the optimal trajectory, the method of motion planning, a system of linear algebraic equations, obstacle avoidance, Dijkstra algorithm

**Graphical annotation (Графическая аннотация)**




**Введение.** Задачи управления беспилотными автомобилями, которые можно рассматривать как часть класса задач автоматизированного управления транспортом, в настоящее время являются актуальными, и их решению придается большое значение. Существенные результаты в данной области были достигнуты такими компаниями, как Tesla, Google, Uber и др. В России в качестве основных результатов можно отметить разработки компании «Яндекс». Однако и у других российских компаний также имеются проекты по созданию беспилотных автомобилей, которые находятся в стадиях запуска или начального развития. Одной из ключевых задач, решаемых исследователями и разработчиками при создании беспилотных автомобилей, является прокладка маршрута. Несмотря на большое количество публикаций [3, 5, 10–13, 18–19], эта тематика, с точки зрения авторов данной работы, исследована недостаточно полно. Поэтому настоящая статья посвящена разработке модификации известного алгоритма прокладки маршрута для беспилотных транспортных средств [17], его программной реализации и апробации.

**Постановка задачи.** Задача автоматизированного управления транспортным средством без участия человека может быть решена различными способами. Однако всегда можно выделить две ее ключевые составляющие – построение программной траектории движения и регулятора, обеспечивающего движение по этой траектории. Отметим, что в ряде случаев при построении траектории необходимо учитывать наличие навесного оборудования, способного динамически изменять габариты транспортного средства [1].

В данной работе описываются теоретические и практические подходы к разработке и реализации системы построения программной траектории движения. Они могут быть использованы, в частности, в инициативных проектах по созданию беспилотных транспортных средств, реализуемых творческими коллективами ВолгГТУ. Основной задачей в данном случае является

разработка и реализация системы построения достижимой программной траектории, которая позволит автономному наземному транспортному средству двигаться, избегая препятствий.

**Обзор существующих решений.** Имеется большое количество методов построения программной траектории движения, которые могут быть применены к наземным транспортным средствам. В работах [3, 5, 10–11, 18] дан обзор различных методов планирования движения применительно к беспилотным автомобилям. Подробные обзоры целостных систем управления беспилотным автомобилем приведены в статьях [8, 12, 13, 19] участников соревнований автомобилей-роботов DARPA Urban Challenge.

Среди большого количества методов планирования движения можно выделить следующие: методы планирования на графах, случайные (sampled-based) методы, методы интерполяции кривыми, методы численной оптимизации и ряд других. Многие из фактически применяемых методов являются гибридными и сочетают в себе несколько подходов.

Наиболее простыми и распространенными методами построения пути являются методы поиска на графах. Широко распространенным методом данной группы является метод клеточной декомпозиции, когда препятствия представляются в виде регулярной сетки (Occupancy Grid). При этом каждой клетке ставится в соответствие наличие (или вероятность наличия) препятствия. Поиск пути перемещения объекта может осуществляться рядом алгоритмов поиска на графах. Подобные методы широко применяются для мобильных роботов. Недостатком подобного подхода является тот факт, что указанные методы позволяют получить только геометрический путь, без учета профиля (динамики) изменения скорости во времени. При этом получаемые пути могут не удовлетворять кинематическим и динамическим ограничениям для конструкции робота, а возможно, и его грузов.

Существует ряд модификаций графовых методов, например, алгоритм Hybrid A* [9]. В этом алгоритме дискретным элементам сетки сопоставляются непрерывные координаты автомобиля в пространстве состояний, а переходы между вершинами графа осуществляются с помощью решения прямой задачи динамики с учетом управляющих воздействий, приложенных к модели автомобиля.

Другим широко распространенным методом является метод регулярной решетки состояний (state lattice) [17]. В этом методе используется граф, узлы которого расположены в форме регулярной решетки, а ребра соединяют начальное состояние и все возможные состояния, которые могут быть достижимы из него. Важной особенностью этого метода является то, что может быть выделено некое подмножество этих вершин и ребер, инвариантное к положению начального состояния. Таким образом, можно заранее рассчитать ограниченный набор состояний и достижимых траекторий для переходов между ними, а затем использовать их для поиска наиболее подходящего решения. Это позволяет существенно сэкономить вычислительные ресурсы управляющего устройства.

Еще одним недостатком графовых алгоритмов, использующих регулярные сетки, является быстрый рост сложности при увеличении размерности задачи. Поэтому в настоящее время для сложных задач планирования движения, например, задач перемещения твердого тела в трехмерном пространстве, применяются алгоритмы на основе Rapidly-Exploring Random Trees (RRT) [14]. Идея этого метода заключается в том, что из начального состояния строится дерево, которое может достичь конечного состояния, путем выбора случайных точек в конфигурационном пространстве или пространстве состояний. Алгоритм является итеративным, при этом на каждой итерации используются случайные выборки состояний (вершин дерева), за счет которых дерево пытается расшириться. Выбирается случайное состояние, после чего находится ближайшее к нему состояние в дереве. Если между ними нет препятствий, то строится ребро от ближайшего состояния к новому (либо по направлению к новому, но не превышающее заданной длины). Если препятствие есть, новая вершина отклоняется, т.е. ребро не строится. Существует ряд модификаций этого алгоритма, например, RRT* [18], представляющий его вариант, оптимизированный с точки зрения получения решения, так как базовый алгоритм возвращает неоптимальную траекторию.

Другое семейство алгоритмов, применяемое в системах планирования движения беспилотных автомобилей, основано на представлении траектории движения в форме кривых. В данных методах применяются полиномы различных порядков (третьего, пятого), кривые Безье, окружности, сплайны [6]. Чаще всего эти методы используются при движении по дорогам, что является более простой задачей, чем построение траектории для движения в произвольном окружении. Кривые применяются для построения траектории движения между последовательностью ключевых точек, описывающих маршрут движения.

**Предлагаемые решения.** В данной работе предложен метод построения программной траектории беспилотного наземного транспортного средства с использованием полиномов пятого порядка. Этот метод является модификацией решения, предложенного одной из команд [16], участвовавшей в DARPA Urban Challenge. Данное мероприятие представляет собой соревнования автомобилей-роботов, проводимых в США, целью которых является создание полностью автономных транспортных средств.

Планирование траектории осуществляется независимо для продольного движения вдоль опорной траектории и поперечного (по отношению к этой траектории) движения без учета неровностей рельефа дороги, т.е. принимается, что движение осуществляется по плоскости. Планирование осуществляется в подвижной системе координат, движущейся по опорной траектории вместе с автомобилем, как показано на рисунке 1.

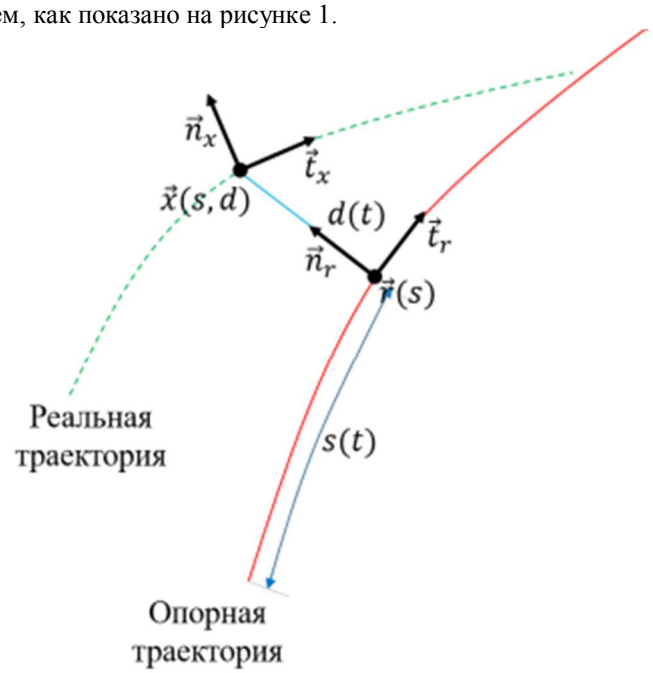



Рисунок 1 – Система координат для планирования движения

Опорная траектория (изображена на рисунке 1 красной линией) представлена натурально параметризованной кривой, т.е. кривой, параметризованной длиной ее дуги. Здесь $s(t)$ – покрытая длина дуги в момент времени $t$, т.е. длина дуги опорной траектории, которую преодолел автомобиль к моменту времени $t$.

Положение автомобиля в глобальной системе координат обозначено радиус-вектором $\vec{x}$. Этому положению автомобиля соответствует подвижная система координат $\vec{r}(s)$, представляющая собой ортонормированный репер Френе [7], где $\vec{t}_r$ и $\vec{n}_r$ – касательный и нормальный векторы к опорной траектории соответственно. Таким образом, положение автомобиля в глобальной декартовой системе координат $(x, y)$ может быть представлено в подвижной системе координат как $(s, d)$, где $d$ – расстояние между автомобилем и опорной траекторией.

За основу функции качества, применяемой при принятии решения в процессе прокладки маршрута, был взят интеграл от производной ускорения по времени $\dddot{s}$, называемый рывком (jerk) [16]:

$$J_s = \int_0^T \dddot{s}(t)^2\, dt, \qquad (1)$$
$$J_d = \int_0^T \dddot{d}(t)^2\, dt.$$

Данный выбор является распространенным во многих алгоритмах планирования движения автомобилей. Использование функции качества типа (1) позволяет строить траектории с более плавными маневрами, при выполнении которых величина ускорения не превышает значения, задаваемого в качестве ограничения. Использование данной функции качества позволяет также в целом уменьшать количество маневров и рывков. В итоге предпочтение будет отдаваться траекториям, позволяющим достичь целей перемещения с меньшим количеством

маневров. Минимизация числа рывков в процессе движения способствует комфорту пассажиров, а также лучшей сохранности некоторых видов перевозимых грузов.

Полную функцию качества для продольного и поперечного движения можно записать следующим образом:

$$C_d = K_{dj}\int_0^T \dddot{d}(t)^2 dt + K_d d(T)^2 + K_{dt}T,$$
$$C_s = K_{sj}\int_0^t \ddot{s}(t)^2 dt + K_s(s(T) - S_1)^2 + K_v\left(\dot{s}(T) - \dot{S}_1\right) + K_{st}T, \quad (2)$$
$$C = K_{lon}C_s + K_{lat}C_d,$$

где $s(t), d(t)$ – продольная и поперечная траектории соответственно; T – длительность маневра; $S_1$ – целевое продольное состояние; $K_{dj}$, $K_d$, $K_{dt}$, $K_{sj}$, $K_s$, $K_v$, $K_{st}$ – весовые коэффициенты.

В соответствии с [15], траектории, минимизирующие подобные функции качества, могут быть найдены в форме полиномов пятого порядка:

$$s(t) = a_0t^5 + a_1t^4 + a_2t^3 + a_3t^2 + a_4t + a_5. \quad (3)$$

Полином здесь используется не только для интерполяции формы траектории, по которой движется автомобиль, но и для определения профиля скорости и ускорения. В качестве них применяются первая и вторая производные функции $s(t)$ соответственно. Это позволяет строить траектории, которые будут непрерывными и гладкими для положения ($s(t)$) и скорости ($ds(t)$ / $dt$). Полная траектория движения формируется из набора таких коротких участков, описываемых полиномами. Для этого планирование движения автомобиля повторяется с определенной периодичностью, а в качестве начальных условий используются текущее положение, скорость и ускорение автомобиля. Как было сказано выше, полиномы непрерывны и гладки по положению и скорости. Таким образом, итоговая траектория, полученная из набора участков, также будет непрерывной и гладкой.

Коэффициенты полиномов для оптимальной траектории на отдельном участке могут быть найдены путем решения системы из шести линейных уравнений. При этом необходимо учитывать начальное состояние траектории (текущее состояние автомобиля, получаемое с использованием сенсоров) $S_0$, требуемое конечное состояние $S_1$ автомобиля и длительность маневра $T$. В данной работе для получения начального приближения для времени маневра применяется длительность равноускоренного движения при заданных $S_0, S_1$.

Для определения оптимальной траектории, которая удовлетворяет ограничениям (максимальное продольное ускорение, максимальная продольная скорость, максимальное поперечное ускорение, минимальный радиус кривизны) и не пересекается с препятствиями, формируется набор траекторий-кандидатов путем варьирования конечного состояния (или других параметров – поперечного отклонения, продольного положения и скорости, длительности маневра). После чего из полученных траекторий выбирается оптимальная. При этом она не пересекается с препятствиями и удовлетворяет заданным ограничениям. Пример набора траекторий представлен на рисунке 2. Красным цветом здесь обозначены траектории, удовлетворяющие ограничениям, серым – не удовлетворяющие ограничениям. Для построения графиков использовалась библиотека matplotlib для языка Python.

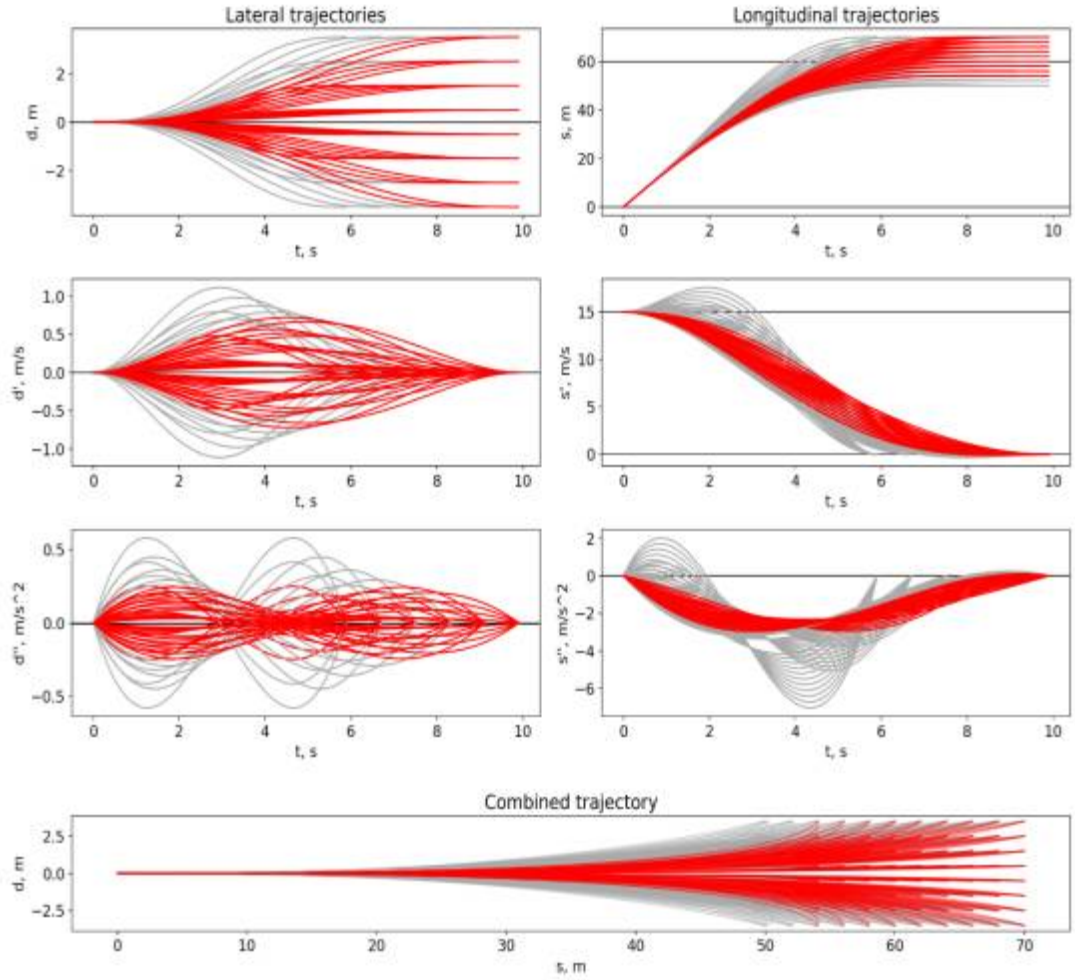



Рисунок 2 – Пример формирования набора траекторий

На рисунке 2 применены следующие обозначения:

- графики lateral trajectories использованы для описания поперечного перемещения, при этом d – собственно зависимость поперечного перемещения от времени, d` – график зависимости поперечной скорости, а d`` – график зависимости поперечного ускорения от времени;
- графики longitudinal trajectories использованы для описания продольного перемещения, при этом d – собственно зависимость продольного перемещения от времени, d` – график зависимости продольной скорости, а d`` – график зависимости продольного ускорения от времени;
- на графике combined trajectory представлена собственно траектория движения с учетом продольных и поперечных перемещений.

Пример выбора оптимальной траектории для объезда препятствия показан на рисунке 3. Здесь синим цветом обозначено препятствие, красным – целевое положение, серые линии используются для обозначения траекторий, не удовлетворяющих ограничениям (пересекающиеся с препятствием), зеленым – удовлетворяющие ограничениям, красным – выбранная оптимальная траектория. Если заданным ограничениям удовлетворяют несколько траекторий-кандидатов, то в простейшем случае любая из них может быть выбрана в качестве оптимальной. Также может быть выбрана наилучшая траектория из подходящих траекторий-кандидатов путем дальнейшего уточнения ограничений. Однако данный вопрос выходит за рамки данной работы и будет в дальнейшем рассмотрен в отдельной статье.

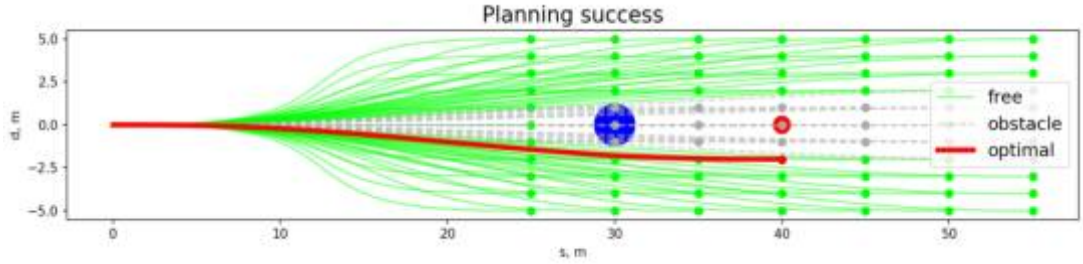



Рисунок 3 – Пример планирования траекторий с учетом препятствия

Для улучшения планирования траектории в более сложных случаях было предложено осуществлять планирование на несколько шагов по времени вперед. При этом каждое конечное состояние является начальным состоянием для следующего этапа планирования. Таким образом, была применена концепция, похожая на state lattice [17]. Пример планирования траекторий на несколько шагов представлен на рисунке 4. Для наглядности осуществлялось варьирование только поперечных параметров траекторий.

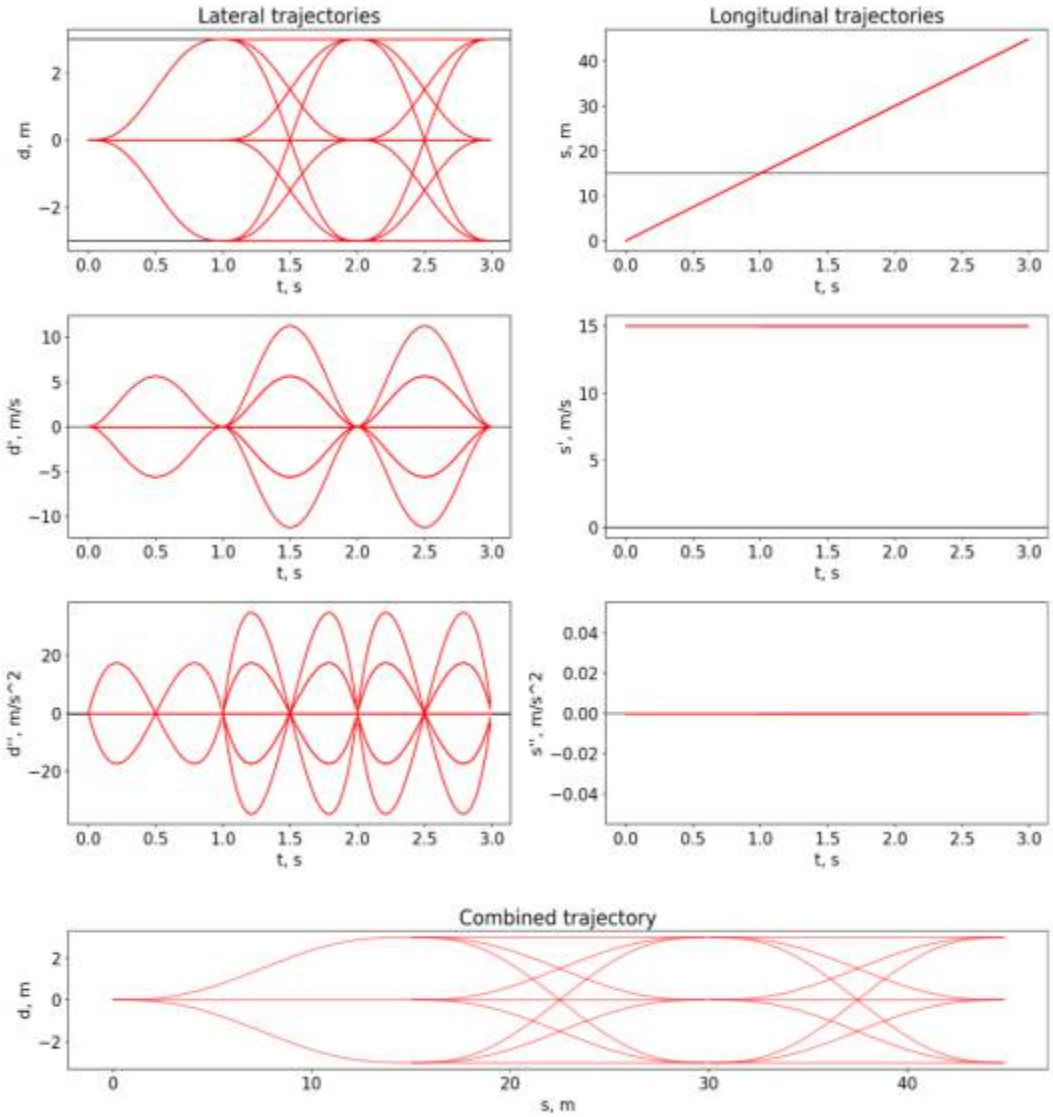



Рисунок 4 – Планирование траекторий на несколько шагов вперед

В этом случае требуется определить оптимальное конечное состояние и оптимальную траекторию для его достижения. В данном случае применяется алгоритм Дейкстры [4] для нахождения кратчайшего пути на графе. Вес траектории складывается из весов ребер, определяющихся резкостью маневров, и весов вершин, определяющихся удаленностью вершины от конечного положения траектории:

$$C_{move} = K_{lon}K_{sj}\int_0^T \dddot{s}(t)^2 dt + K_{lat}K_{dj}\int_0^T \dddot{d}(t)^2 dt,$$
$$C_{state} = K_{lon}\left[K_s(s(T) - S_1)^2 + K_v\left(\dot{s}(T) - \dot{S}_1\right)^2\right] + K_{lat}K_d(T)^2, \quad (4)$$
$$C = C_{move} + C_{state}.$$

Полученный алгоритм можно описать следующим образом. Вначале осуществляется построение графа возможных состояний и траекторий путем варьирования конечных состояний, как было описано выше. Затем с помощью алгоритма Дейкстры определяется кратчайший путь и его вес $C_{move}$ от начального состояния до всех состояний в графе. На последнем шаге выбирается конечное состояние с минимальным суммарным весом $C$.

Блок-схема алгоритма формирования набора траекторий приведена на рисунке 5.

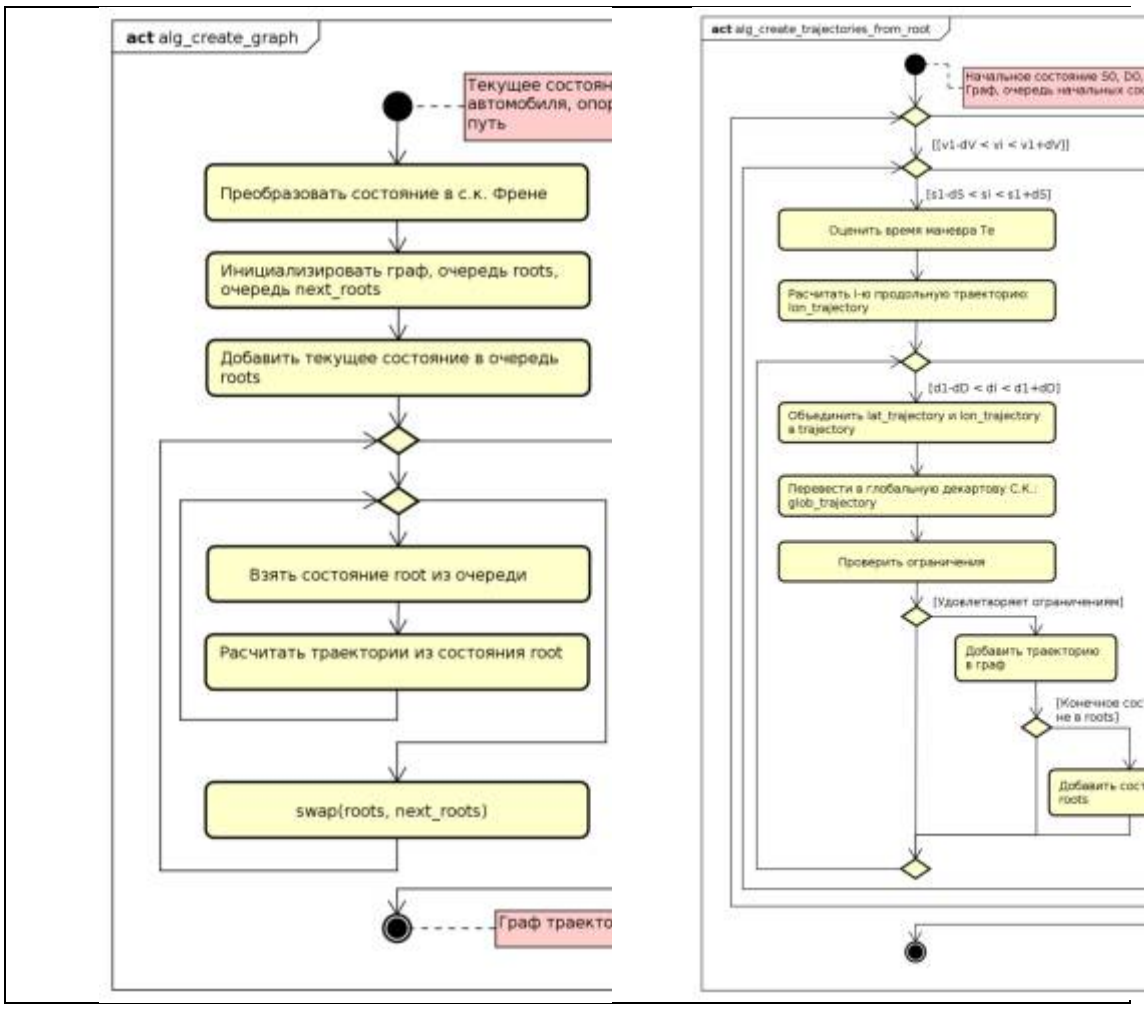



Рисунок 5 – Блок-схема алгоритма формирования набора траекторий: а – блок-схема алгоритма построения графа; б – блок-схема шага расчета набора траекторий из заданного начального состояния

На вход подаются состояние автомобиля, полученное с помощью системы SLAM (Simultaneous Localization And Mapping), карта препятствий и локальная цель. Вначале происходит преобразование начального и целевого состояния в подвижную систему координат Френе, заданную опорной траекторией. Инициализируются пустой граф и две пустых очереди для хранения новых состояний, из которых впоследствии осуществляется формирование траекторий. В качестве первого начального состояния в очередь помещается текущее состояние

автомобиля в системе координат Френе. Алгоритм работает до тех пор, пока не будет построено необходимое количество слоев графа. При заполнении каждого слоя из очереди извлекается очередное состояние, начиная с которого осуществляется формирование набора траекторий. Конечные состояния этих траекторий помещаются в очередь, для использования при формировании нового слоя. В алгоритме применяются две очереди состояний, которые переключаются после заполнения каждого слоя для оптимизации процесса вычислений.

Следующим этапом алгоритма планирования траектории является нахождение кратчайших путей до всех вершин с помощью алгоритма Дейкстры. Блок-схема этого алгоритма, представленная в форме UML-диаграммы, изображена на рисунке 6.

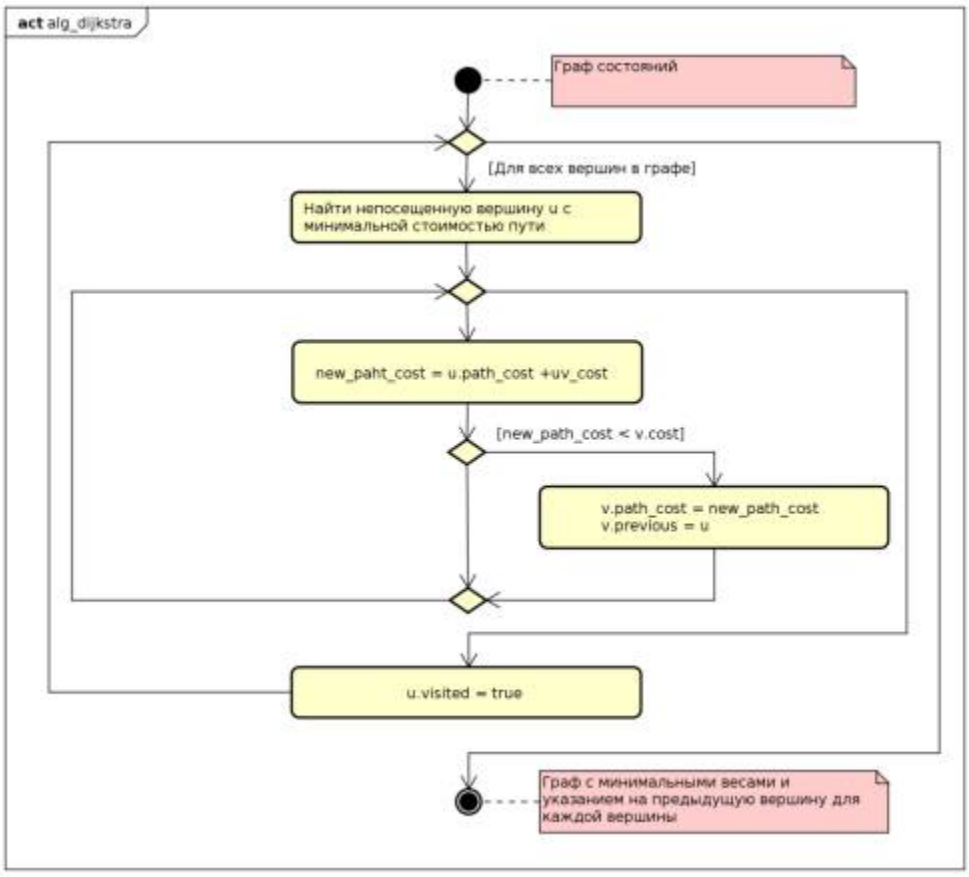



Рисунок 6 – Реализация алгоритма Дейкстры

В начале алгоритма стоимость путей до всех вершин (кроме начальной) инициализируется максимально возможными значениями. Все вершины отмечаются как непосещенные. На каждой итерации алгоритма выбирается вершина u с минимальной стоимостью пути от текущей до начальной. Это может быть сделано простым перебором за O(N), причем использование различных структур данных может сократить данную оценку. Затем перебираются все вершины v, связанные ребром с u. Рассчитывается стоимость пути до вершин v при движении через вершину u: new_path_cost = u:path_cost + uv_edge_cost, и если эта стоимость меньше, чем та, которая сохранена в вершине v, то значение стоимости пути до вершины v обновляется. Для того чтобы в дальнейшем можно было восстановить кратчайший маршрут, а не только получить его стоимость, для вершины v обновляется предыдущая вершина кратчайшего маршрута – вершина v.previous. После обработки всех связанных вершин вершина u отмечается как посещенная. Затем алгоритм переходит к следующей итерации.

Алгоритм завершается после того, как все вершины будут посещены. Финальным этапом работы алгоритма является выбор оптимального конечного состояния. Оно обладает минимальным суммарным весом C = C*move* + C*state*, где C*move* получен в результате использования алгоритма Дейкстры, а C*state* для каждой вершины рассчитан при формировании графа.

**Полученные результаты.** Предложенный алгоритм был протестирован на небольшой мобильной колесной платформе, использованной в качестве модели автомобиля, представленной на рисунке 7.

Рисунок 7 – Платформа, использованная для экспериментов

Данная статья написана во многом по следам исследований, проведенных в рамках подготовки магистерских диссертаций. Поэтому выбор компонентов был обусловлен, в первую очередь, наличием этих компонентов в вузе.

Платформа оснащена LiDAR VLP-16 для обнаружения препятствий, стереокамерой ZED, использующейся для определения положения в пространстве с помощью SLAM и бортовым компьютером NVidia Jetson TX2.

Стереокамера ZED оснащена двумя камерами с высоким разрешением, качественной оптикой и встроенной системой инерциальной навигации, повышающей точность определения ориентации камеры. Поступающее в комплекте с камерой программное обеспечение, реализующее алгоритм SLAM, позволяет определять положение камеры с высокой точностью (сантиметры и менее). Встраиваемый компьютер NVidia Jetson TX2 имеет компактные размеры, малое энергопотребление, но при этом обладает сравнительно высокой производительностью по сравнению с другими встраиваемыми системами, такими как, например, Raspbery Pi. Он оснащен GPU NVidia с архитектурой Pascal, что позволяет существенно повысить быстродействие для ряда задач.

В ходе экспериментов было протестировано движение модели по заданной опорной траектории, на которой имелись препятствия. Эксперименты показали, что предложенный алгоритм позволяет успешно объезжать препятствия. На рисунке 8 представлена визуализация работы алгоритма в реальных условиях.

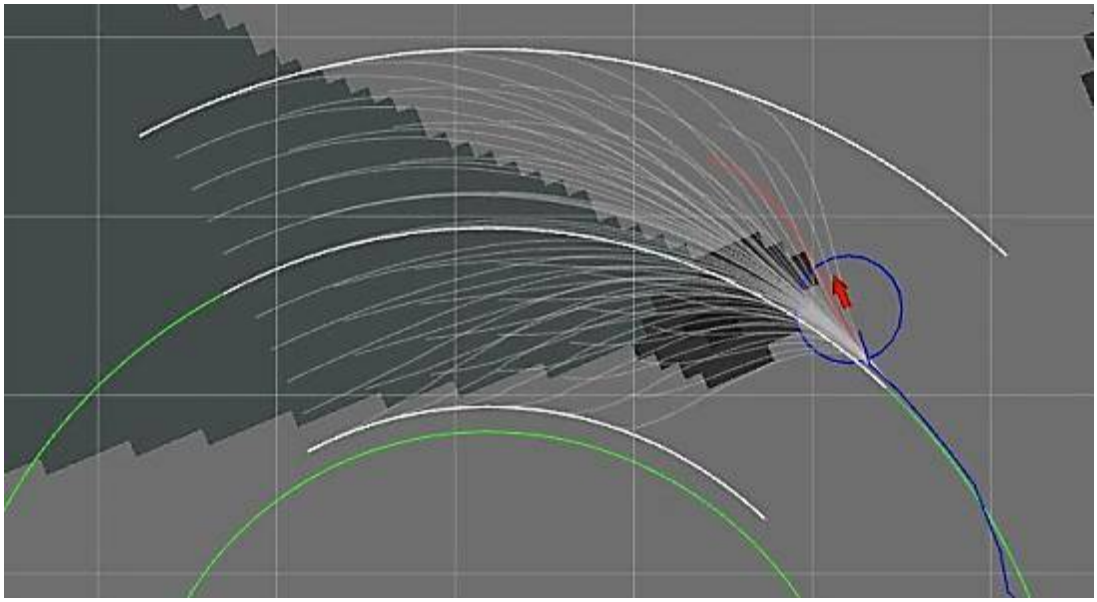


Рисунок 8 – Результат эксперимента с объездом препятствий

В ходе эксперимента осуществлялось движение модели автомобиля по заданной спиралевидной траектории (она обозначена на рисунке зеленой линией) при наличии препятствий. Для распознавания препятствий использовалось облако точек, полученное с помощью LiDAR. Препятствия представлялись в виде регулярной сетки, ячейки которой могут иметь одно из следующих значений: свободно, препятствие и неизвестно. Такая карта препятствий носит название Occupancy Grid. Алгоритм построения карты препятствий содержит несколько шагов:

- получение облака точек;
- разбиение облака точек на сетку с ячейкой заданного размера (при этом изначально все ячейки имеют значение «неизвестно»);
- ячейка помечается как препятствие, если число точек в ней, расположенных выше заданной высоты ($z > z\_max$), превышает некоторое заданное значение (cnt_max);
- в качестве свободных принимаются ячейки, расположенные в зоне прямой видимости от LiDAR до ячеек препятствий, не включая их;
- вокруг каждой ячейки препятствия создавалась буферная зона путем растрирования окружности некоторого радиуса с помощью алгоритма Брезенхема [6] – для того, чтобы учесть габариты транспортного средства.

На рисунке 8 препятствия обозначены черным цветом, свободные клетки – серым, все остальное – неизвестно (тень от препятствия). Красной стрелкой с окружностью вокруг нее обозначено текущее положение автомобиля, получаемое с помощью SLAM-алгоритма, синей линией – реальная траектория автомобиля. Толстые белые линии – границы поперечного планирования траекторий, светло-белые линии – сформированные траектории-кандидаты, выбранная оптимальная программная траектория для следующего участка движения обозначена красной линией. Алгоритм работает итеративно, поэтому до того, как автомобиль доедет до конца запланированной траектории (красная линия), будет сформирован новый участок программной траектории. Рисунок построен при помощи RViz визуализатора, входящего в состав ROS.

**Выводы.** Авторами был разработан и реализован метод построения программных траекторий для автономного наземного транспортного средства. Представленные решения были апробированы на реальной авторской модели беспилотного транспортного средства, испытания которой признаны удачными. Модель объезжала препятствия, ее перемещения соответствовали задаваемым ограничениям.

Дальнейшее улучшение возможностей системы построения программных траекторий может быть осуществлено за счет применения специализированных вычислительных средств для повышения производительности бортовой вычислительной системы [2]. Также для проверки и/или формирования участков траектории с целью достижения лучшего учета динамических ограничений транспортного средства может быть использована программа «ФРУНД» (Формирование решений уравнений нелинейной динамики, http://frund.vstu.ru) – программный комплекс для решения задач многотельной динамики твердых и упругих тел, разрабатываемый творческим коллективом ВолгГТУ.

**Библиографический список**

1. Егунов В. А. Об управлении манипуляционным механизмом мобильного робота / В. А. Егунов, А. П. Жуков, М. И. Потапов // Известия Волгоградского государственного технического университета. – 2011. – № 11 (84). – С. 49–51.
2. Егунов В. А., Лукьянов В. С. Аппаратные методы решения задач линейной алгебры : монография / В. А. Егунов, В. С. Лукьянов.– Волгоград : Волгоградский гос. технический ун-т, 2007.
3. Казаков К. А. Обзор современных методов планирования движения / К. А. Казаков, В. А. Семенов // Труды Института системного программирования РАН. – 2016.
4. Кормен, Т. Алгоритмы: построение и анализ = Introduction to Algorithms / Т. Кормен, Ч. Лейзерсон, Р. Ривест, К. Штайн ; под ред. И. В. Красикова. – 2-е изд. – Москва : Вильямс, 2005. – 1296 с.
5. Лю В. Методы планирования пути в среде с препятствиями (обзор) / В. Лю // Математика и математическое моделирование. – 2018. – № 1. – С. 15–58.
6. Палагута К. А. Кривые Безье как опорные кривые для построения траектории автомобиля / К. А. Палагута, А. А. Алексеев. – Москва : Мир, 1989.
7. Роджерс Д. Алгоритмические основы машинной графики / Д. Роджерс // Инновации на основе информационных и коммуникационных технологий. – 2010. – № 1. – С. 453–455.
8. Топоногов В. А. Дифференциальная геометрия кривых и поверхностей / В. А. Топоногов. – Москва : Физматкнига, 2012. – ISBN 978-5-89155-213-5.
9. A Perception-Driven Autonomous Urban Vehicle / J. J. Leonard [et al.] // Journal of Field Robotics. – 2008. – Oct. – Vol. 25. – P. 727–774. DOI: 10.1002/rob.20262.
10. Application of Hybrid A* to an Autonomous Mobile Robot for PathPlanning in Unstructured Outdoor Environments / J. Petereit [et al.] // ROBOTIK 2012 : 7th German Conference on Robotics. – 05/2012. – P. 1–6.
11. A Review of Motion Planning Techniques for Automated Vehicles / D. Gonzalez Bautista [et al.] // IEEE Transactions on Intelligent Transportation Systems. – 2015. – Nov. – P. 1–11. DOI: 10.1109/TITS.2015.2498841.
12. A Survey of Motion Planning and Control Techniques for Self-drivingUrban Vehicles / B. Paden [et al.] // CoRR. – 2016. – Vol. abs/1604.07446. – arXiv: 1604.07446. – Режим доступа: http://arxiv.org/abs/1604.07446, свободный. – Заглавие с экрана. – Яз. рус.
13. Autonomous driving in urban environments: Boss and the Urban Challenge // Journal of Field Robotics Special Issue on the 2007 DARPA UrbanChallenge. Part I. – 2008. – June. – Vol. 25, № 8. – P. 425–466.
14. Junior: The Stanford Entry in the Urban Challenge / M. Montemerlo[et al.] // Journal of Field Robotics. – 2008. – Sept. – Vol. 25. – P. 569–597. DOI: 10.1002/rob.20258.
15. LaValle S. M. Rapidly-Exploring Random Trees: A New Tool for Path Planning / S. M. LaValle. – 1998.
16. Local Path Planning And Motion Control For Agv In Positioning / A. Takahashi [et al.] // Proceedings. IEEE/RSJ International Workshop on Intelligent Robots and Systems. (IROS '89) The Autonomous Mobile Robots and Its Applications. – 09/1989. – P. 392–397. DOI: 10. 1109/IROS.1989.637936.
17. Optimal Trajectory Generation for Dynamic Street Scenarios in a FrenetFrame / M. Werling [et al.]. – 06/2010. – P. 987–993. DOI:10.1109/ROBOT.2010.5509799.
18. Pharpatara P. Shortest path for aerial ehicles in heterogeneous environment using rrt* / P. Pharpatara, B. Herisse and Y. Bestaoui // Proceedings of the IEEE International Conference on Robotics and Automation. – 2015. – P. 6388–6393.
19. Pivtoraiko M. Efficient constrained path planning via search instate lattices / M. Pivtoraiko, A. Kelly // The 8th International Symposium on Articial Intelligence, Robotics and Automation in Space. – 09/2005.
20. Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions / C. Katrakazas [et al.] // Transportation Research Part C: Emerging Technologies. – 2015. – Nov. – Vol. 60. – P. 416–442. DOI: 10.1016/j.trc.2015.09.011.
21. Team AnnieWAY's autonomous system for the 2007 DARPA Urban Challenge/ S. Kammel [et al.] // Journal of Field Robotics. – 2008. – Vol. 25, № 9. – P. 615–639. DOI: 10.1002/rob.20252. – Режим доступа: eprint:https://onlinelibrary.wiley.com/doi/pdf/10.1002/rob.20252, свободный. – Заглавие с экрана. – Яз. рус.

**References**

1. Egunov V. A., Zhukov A. P., Potapov M. I. Ob upravlenii manipulyatsionnym mekhanizmom mobilnogo robota [About controlling manipulative mechanism for mobile robot]. *Izvestiya VolgGTU. Seriya. Aktualnye problemy upravleniya, vychislitelnoy tekhniki i informatiki v tekhnicheskikh sistemakh* [News of Volgograd State Technical University. Series: Actual problems of control, computer engineering and informatics in technical systems], 2011, no. 11 (84), pp. 49–51.
2. Egunov V. A., Lukyanov V. S. *Apparatnye metody resheniya zadach lineynoy algebry* [Hardware methods for solving linear algebra problems]. Volgograd, Volgograd State Technical University, 2007.
3. Kazakov K. A., Semenov K. A. Obzor sovremennykh metodov planirovaniya dvizheniya [Review of modern methods of traffic planning]. *Trudy Instituta sistemnogo programmirovaniya RAN* [Proceedings of the Institute of System Programming RAS], 2016.
4. Kormen T., Leyzerson Ch., Rivest R., Shtayn K. *Algoritmy: postroenie i analiz* [Introduction to Algorithms], 2nd ed. Moscow, Wiliams, 2005. 1296 p.

5. Lyu V. Metody planirovaniya puti v srede s prepyatstviyami (obzor) [Path planning methods in an environment with obstacles (overview)]. *Matematika i matematicheskoe modelirovanie* [Matematika i matematicheskoe modelirovanie], 2018, no. 1, pp. 15–58.

6. Palaguta K. A., Alekseyev A. A. Krivye Bezye kak oprnye krivye dlya postroyeniya trayektorii avtomobilya [Bezier curves as reference curves for the construction of the trajectory of the car]. *Innovatsii na osnove informatsionnykh i kommunikatsionnykh tekhnology* [Innovation through information and communication technologies], 2010, no. 1, pp. 453–455.

7. Rodzhers D. *Algoritmicheskie osnovy mashinnoy grafiki* [Algorithmic bases of computer graphics], Moscow, Mir Publ., 1989.

8. Toponogov V. A. *Differentsialnaya geometriya krivykh i poverhnostey* [Differential geometry of curves and surfaces]. Moscow, Fizmatkniga Publ., 2012.

9. Leonard J. J. et al. A Perception-Driven Autonomous Urban Vehicle. *Journal of Field Robotics*, 2008, Oct., vol. 25, pp. 727–774. DOI: 10.1002/rob.20262.

10. Petereit J. et al. Application of Hybrid A* to an Autonomous Mobile Robot for PathPlanning in Unstructured Outdoor Environments. *ROBOTIK 2012; 7th German Conference on Robotics*, 05/2012, pp. 1–6.

11. Gonzalez Bautista D. et al. A Review of Motion Planning Techniques for Automated Vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 2015, Nov., pp. 1–11. DOI: 10.1109/TITS.2015. 2498841.

12. Paden B. et al. A Survey of Motion Planning and Control Techniques for Self-drivingUrban Vehicles. *CoRR*., 2016, vol.abs/1604.07446, arXiv: 1604.07446. Available at: http://arxiv.org/abs/1604.07446.

13. Autonomous driving in urban environments: Boss and the Urban Challenge. *Journal of Field Robotics Special Issue on the 2007 DARPA UrbanChallenge, Part I*, 2008, June, vol. 25, no. 8, pp. 425–466.

14. Montemerlo M. et al. Junior: The Stanford Entry in the Urban Challenge. *Journal of Field Robotics*, 2008, Sept., vol. 25, pp. 569–597. DOI: 10.1002/rob.20258.

15. LaValle S. M. *Rapidly-Exploring Random Trees: A New Tool for Path Planning*, 1998.

16. Takahashi A. et al. Local Path Planning And Motion Control For Agv In Positioning. *Proceedings. IEEE/RSJ International Workshop on Intelligent Robots and Systems*. *(IROS '89) 'The Autonomous Mobile Robots and Its Applications*, 09/1989, pp. 392–397. DOI: 10. 1109/IROS.1989.637936.

17. Werling M. et al. Optimal Trajectory Generation for Dynamic Street Scenarios in a FrenetFrame, 06/2010, pp. 987–993. DOI:10.1109/ROBOT.2010.5509799.

18. Pharpatara P., Herisse B., and Bestaoui Y. Shortest path for aerial ehicles in heterogeneous environment using rrt*. *Proceedings of the IEEE International Conference on Robotics and Automation*, 2015, pp. 6388–6393.

19. Pivtoraiko M., Kelly A. Efficient constrained path planning via search instate lattices. *The 8th International Symposium on Articial Intelligence, Robotics and Automation in Space*, 09/2005.

20. Katrakazas C. et al. Real-time motion planning methods for autonomous on-road driving:State-of-the-art and future research directions. *Transportation Research Part C: Emerging Technologies*, 2015, Nov., vol. 60, pp. 416–442. DOI: 10.1016/j.trc.2015.09.011.

21. Kammel S. et al. Team AnnieWAY's autonomous system for the 2007 DARPA Urban Challenge. *Journal of Field Robotics*, 2008, vol. 25, no. 9, pp. 615–639. DOI: 10.1002/rob.20252. Available at: eprint:https://onlinelibrary.wiley.com/doi/pdf/10.1002/rob.20252.

УДК 004.021, 004.043, 004.4

## СТРУКТУРА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ПРОЦЕССА МОДЕЛИРОВАНИЯ МЕЖМОЛЕКУЛЯРНОГО ВЗАИМОДЕЙСТВИЯ КЛЕТОЧНОЙ МЕМБРАНЫ С ТОКСИКАНТАМИ ПРИ ПОИСКЕ АНТИДОТОВ

*Статья поступила в редакцию 29.08.2019, в окончательной версии – 02.09.2019.*

***Жарких Леся Ивановна***, Астраханский государственный университет, 414056, Российская Федерация, г. Астрахань, ул. Татищева, 20а,

кандидат технических наук, доцент, e-mail: lesy_g@mail.ru

Предложена структура программного обеспечения для процесса моделирования межмолекулярного взаимодействия клеточной мембраны с токсикантами и для осуществления проверки эффективности предполагаемых антидотов. Описаны этапы проведения моделирования взаимодействий между молекулами и поиска антидотов к токсическому воздействию. Представлены схемы каждого этапа работы программного обеспечения. Описаны алгоритмы модулей и программ системы и проиллюстрированы результаты выполнения программы визуализации. Предложены базы данных для хранения результатов моделирования молекулярных взаимодействий, структуры и состава клеточных мембран, Z-матриц молекул, выявленных и заблокированных активных центров компонентов клеточных мембран и найденных антидотов к данному токсиканту. Представленная в статье компьютерная разработка позволит значительно уменьшить время проведения эксперимента, имитировать натурный эксперимент, проанализировать и проверить адекватность результатов математического моделирования на известных межмолекулярных взаимодействиях. Полученные результаты предоставляют возможность проводить поиск широкого круга антидотов для различных токсикантов.