

АННОТАЦИЯ

ABSTRACT

## СОДЕРЖАНИЕ

ВВЕДЕНИЕ . . . . .	3
1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ УПРАВЛЕНИЯ БЕСПИЛОТНЫМ АВТОМОБИЛЕМ . .	5
1.1 Обзор применяемого аппаратного обеспечения . . . . .	5
1.2 Обзор общей структуры системы управления беспилотными автомобилями . . . . .	7
1.3 Обзор методов планирования движения . . . . .	10
1.3.1 Формулирование задачи планирования движения . . . . .	10
1.3.2 Методы поиска на графах . . . . .	11
1.3.3 Hybrid A* . . . . .	12
1.3.4 State Lattice. . . . .	13
1.3.5 Случайные методы . . . . .	14
1.3.6 Методы интерполяции кривыми . . . . .	17
1.4 Обзор некоторых систем управления беспилотными автомобилями . . . . .	18
1.4.1 Команда Junior в Darpa Urban Challenge . . . . .	18
1.4.2 Команда BOSS в Darpa Urban Challenge . . . . .	20
1.4.3 Команда Talos в Darpa Urban Challenge . . . . .	21
1.4.4 Команда AnnieWAY в Darpa Urban Challenge . . . . .	21
2 ПРОЕКТИРОВАНИЕ СИСТЕМЫ УПРАВЛЕНИЯ ДВИЖЕНИЕМ БЕСПИЛОТНОГО АВТОМОБИЛЯ . . . . .	22
2.1 Проектирование общей архитектуры системы управления беспилотным автомобилем . . . . .	22
2.2 Проектирование подсистемы планирования поведения . . . . .	22
2.3 Проектирование подсистемы планирования траектории . . . . .	22
2.4 Проектирование подсистемы следования траектории . . . . .	22
3 РЕАЛИЗАЦИЯ СИСТЕМЫ УПРАВЛЕНИЯ ДВИЖЕНИЕМ БЕСПИЛОТНОГО АВТОМОБИЛЯ . . . . .	23
3.1 Использование ROS в качестве основы для системы управления . . . . .	23
3.2 Реализация подсистемы цифровой карты . . . . .	23
3.3 Реализация подсистемы планирования поведения . . . . .	23
3.4 Реализация подсистемы планирования траектории . . . . .	23
3.5 Реализация подсистемы следования траектории . . . . .	23
4 ЭКСПЕРИМЕНТЫ И ОЦЕНКА РЕЗУЛЬТАТОВ РАБОТЫ . . . . .	24
4.1 ??? . . . . .	24
ЗАКЛЮЧЕНИЕ . . . . .	25

## ВВЕДЕНИЕ

В настоящее время активно ведутся исследования и разработки в области беспилотных транспортных средств, в частности, беспилотных автомобилей. Беспилотные транспортные средства являются очень перспективной технологией, на которую делают ставку многие крупные автопроизводители и IT-компании. В частности, исследованиями и разработками в этой области занимаются такие компании как Waymo (Google), Tesla, Audi [24], Яндекс [34], GM, Apple, Uber, Intel совместно с BMW и ряд других. Наибольших успехов в этом достигли Waymo, которая запустила тестовый коммерческий сервис беспилотного такси для ограниченного круга испытателей в нескольких городах штата Аризона, Яндекс, запустивший беспилотное такси в Иннополисе и технопарке Сколково, и компания Tesla, выпускающая серийные автомобили с ограниченными функциями автопилота. **добавить источники**

Массовое применение беспилотных автомобилей на дорогах общего пользования может привести к ряду положительных изменений, таких как

- уменьшение человеческого фактора, что уменьшит количество ДТП и увеличит безопасность на дорогах;
- увеличению пропускной способности дорог и уменьшение пробок, потому что беспилотные автомобили смогут поддерживать меньшее расстояние друг с другом, быстрее реагировать на изменения дорожной ситуации;
- увеличение плотности парковок и иных инфраструктурных объектов по тем же причинам, что позволит сэкономить площадь;
- уменьшение количества личных автомобилей, по причине того, что, согласно анализу, один общественный беспилотный автомобиль может заменить девять–тринадцать личных автомобилей без компромиссов в текущих сценариях использования [11], что еще больше сократит нагрузку на инфраструктуру, уменьшит пробки и сократит вредные выбросы.

Эти и ряд других возможностей делают беспилотные транспортные средства крайне полезными и перспективными в будущем.

**И вот как-то надо перейти к тому, какого хера мы это делаем**

**Поэтому актуальной задачей является сделать вид, что мы тоже не лохи и изобразить какое-то подобие беспилотного автомобиля, чтобы потом еще десять лет показывать на каких-нибудь выставках и днях открытых дверей.**

Целью данной диссертационной работы является разработка и реализация методов управления движением беспилотного автомобиля. Для достижения данной цели необходимо решить следующие задачи:

- 1) анализ существующих подходов к задаче управления беспилотными автомобилями в целом, выделение типичных подсистем,
- 2) проектирование подсистем управления движением беспилотным автомобилем,
- 3) реализация подсистем управления движением,
- 4) проведение экспериментов и оценка результатов работы.

В первой главе настоящей работы приведен обзор и анализ существующих подходов к построению систем управления беспилотными автомобилями в целом, анализ возможных структур систем управления, выделение типичных подсистем.

Во второй главе рассматривается проектирование подсистем, отвечающих за управление движением автономного автомобиля.

В третьей главе рассматривается реализация рассмотренных подсистем, а также методика их

применения для управления беспилотным автомобилем.

В четвертой главе рассматриваются экспериментальная часть работы, связанная с использованием разработанной системы для проведения исследования движений автомобиля.

В разделе "Выводы" подводятся итоги создания подсистемы управления беспилотным автомобилем и рассматриваются результаты экспериментов.

Научная новизна и актуальность. Хотя, актуальность вроде расписал, но не сказал явно это слово, надо сказать

## 1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ УПРАВЛЕНИЯ БЕСПИЛОТНЫМ АВТОМОБИЛЕМ

В этой главе рассматривается обзор существующих систем управления беспилотными автомобилями. Определены основные принципы и выделены основные подсистемы системы управления. Произведен обзор способов построения траекторий движения и способов удержания этой траектории.

### 1.1 Обзор применяемого аппаратного обеспечения

Беспилотные автомобили должны функционировать в условиях сложной окружающей среды с большим количеством различных неподвижных и движущихся препятствий различных конфигураций, в окружении других участников дорожного движения, пешеходов и т.п. Для обеспечения высокого уровня безопасности для пассажиров и других представителей дорожного движения беспилотный автомобиль должен оперативно и адекватно реагировать на изменяющиеся условия окружения, для этого автомобиль должен быть оснащен различными датчиками и реализовывать надежные и эффективные алгоритмы обработки данных.

Важным этапом в развитии беспилотных автомобилей были соревнования Darpa Grand Challenge (2004г, 2005г) и Darpa Urban Challenge (2007г), в ходе которых команды соревновались в разработке и постройке беспилотного автомобиля, который сможет самостоятельно проехать в трассу в загородных условиях (2004г, 2005г) и в городских условиях (2007г). Некоторые команды опубликовали ряд работ, в которых описаны подходы, использованные ими при разработке беспилотного автомобиля.

Главным датчиком большинства беспилотных автомобилей является 3D LIDAR (Light Identification Detection and Ranging) — вращающийся многолучевой лазерный дальномер, позволяющий получать данные о конфигурации окружающего пространства в трехмерном пространстве на большом расстоянии. Так, например, ЛИДАР Velodyne HDL-64E способен обнаруживать объекты с высокой отражающей способностью (например, автомобили) на расстоянии до 120 м, а объекты с низкой отражающей способностью на расстоянии до 50 метров [14]. ЛИДАР позволяет получить большое количество информации об окружающем пространстве и с его помощью значительно легче и надежнее реализовать такие задачи, как обнаружение препятствий, определение дороги и т.п. Из рассмотренных команд, участвовавших в Darpa Urban Challenge: Junior (Стэнфордский университет) [16], Talos (Массачусетский технологический институт)[1], BOSS (несколько участников из университета Карнеги-Меллона, компаний General Motors, Caterpillar, Continental, Intel) [8], AnnieWAY (технологический институт Карлсруэ) [32] все команды использовали шестидесятилучевой лучевой ЛИДАР Velodyne HDL-64E в качестве основного источника информации об окружающем пространстве. Дополнительно применялись однолучевые 2D-ЛИДАРЫ в различных конфигурациях, для того, чтобы получить лучшее покрытие слепых зон, расположенные по бокам, на переднем и/или заднем бамперах, на крыше. ЛИДАРЫ не лишены недостатков. Главным их недостатком является высокая стоимость, что делает их не лучшим кандидатом для применения в массовых коммерческих автомобилях, так например шестнадцатилучевой ЛИДАР Velodyne VLP-16 стоит дороже ряда бюджетных автомобилей. [ССЫЛКА](#). Помимо этого, работа ЛИДАРа затруднена или вовсе невозможна в дождь, снег, туман.

Несмотря на большой объем и сравнительно высокую точность данных, получаемых с помощью ЛИДАРа, одного ЛИДАРа не достаточно для построения системы компьютерного зрения беспилотного автомобиля. В дополнение к ЛИДАРАм, все системы комплектуются одной, но чаще несколькими, цветными и/или черно-белыми камерами, а также радарам.

Камеры, наряду с ЛИДАРОм, являются важнейшими датчиками системы компьютерного зрения

беспилотного автомобиля. С помощью камер производится распознавание дорожных знаков, светофоров, дорожной разметки. В настоящее время, в связи со значительным прогрессом в области искусственных нейронных сетей, можно с высокой точностью производить детектирование различных объектов (автомобилей, пешеходов, велосипедистов и других классов объектов), а также сегментацию, т.е. определять, какие пиксели изображения относятся к тем и или иным к классам, таким образом можно детектировать дорогу и ровное пространство, доступное для движения. Тут можно накидать ссылок на какие-нибудь статьи про это дело, тысячи их. Прогресс в этой области делает возможным реализацию системы компьютерного зрения для беспилотного автомобиля без использования ЛИДАРа, что существенно снижает сложность и стоимость такой системы, что важно для серийных автомобилей. В настоящее время автомобили Tesla, оснащенные автопилотом с ограниченной функциональностью, используют только камеры. пруф. Тем не менее, использование ЛИДАРа значительно увеличивает точность и надежность системы, и ЛИДАРЫ используются во всех пруф проектах беспилотных автомобилей. Комбинируя данные с ЛИДАРа и камер (так называемая задача sensor fusion), можно существенно повысить точность таких задач, как распознавание и отслеживание препятствий, определение дороги и т.п. Тут можно кинуть ссылки на статьи про fusion на нейронках.

Другим важным датчиком, применяемым в беспилотных автомобилях, является радар. За последние годы применение радаров в автотранспорте (automotive radar) активно развивалось. Применение радаров не ограничивается полностью беспилотными автомобилями, они находят место и в обычных автомобилях в системах помощи водителю (advanced driver-assistance system, ADAS) и в системах активной безопасности [7], [6]. Радары обладают рядом возможностей, недоступных другим датчикам, таких как возможность работы вне зависимости от погодных условий и уровня освещенности, большая (до 200м) пруф дальность, возможность определять положение объектов и непосредственно определять их скорость с помощью эффекта Доплера.

Например, команда BOSS [8] в Darpa Urban Challenge применяла радар совместно с ЛИДА-Ром для отслеживания объектов. Объекты отслеживались независимо, используя особенности датчиков (непосредственное измерение скорости объектов с помощью радара играло важную роль в алгоритме отслеживания) и затем результаты объединялись.

В таблице 1 приведено сравнение основных свойств и областей применения рассмотренных выше датчиков.

Таблица 1 – Сравнение применяемых датчиков

Датчик	Получаемая информация	Дальность	Влияние освещенности	Влияние погодных условий
Камера	Цветное или ч/б изображение	Не позволяет определить расстояние	Да	Да
Стереокамера	Облако точек	???	Да	Да
3D ЛИДАР	Облако точек 360°	до 120 м	Малое	Да
2D ЛИДАР	Положение препятствий (2D)	???	Малое	Да
Радар	Положение и скорость препятствий (2D)	до 200 м	Нет	Нет

Источники про влияние, дальность итп

TODO: Написать не только про сенсоры, но и про управление

## 1.2 Обзор общей структуры системы управления беспилотными автомобилями

Несмотря на то, что было представлено большое количество различных систем управления беспилотными автомобилями, все они обладают схожей структурой. В общем виде, систему управления беспилотным автомобилем можно разделить на следующие подсистемы:

- интерфейс сенсоров, позволяющий получать данные от сенсоров;
- подсистема восприятия (perception), осуществляющая построение комплексной информации об окружающем пространстве, на основе данных от сенсоров, определяя положение автомобиля, дороги, статические и динамические препятствия, распознавания дорожные знаки, светофоры, разметку, и формирующая в конечном итоге цифровую карту окружающего пространства, которая в дальнейшем используются другими подсистемами;
- подсистема управления движением, осуществляющая принятие решений и построение безопасной и достижимой траектории и осуществляющая движение по траектории, формирования управляющих сигналов, таких как угол поворота руля, газ, тормоз.

Подобная высокоуровневая архитектура применена как командами Darpa Urban Challenge [16], [1], [8], [32], так и описана в более современных статьях, таких как [10], [28], [4].

Архитектура обобщенной системы управления автономным автомобилем, построенная на основе анализа ряда работ [3], представлена на рисунке 1.

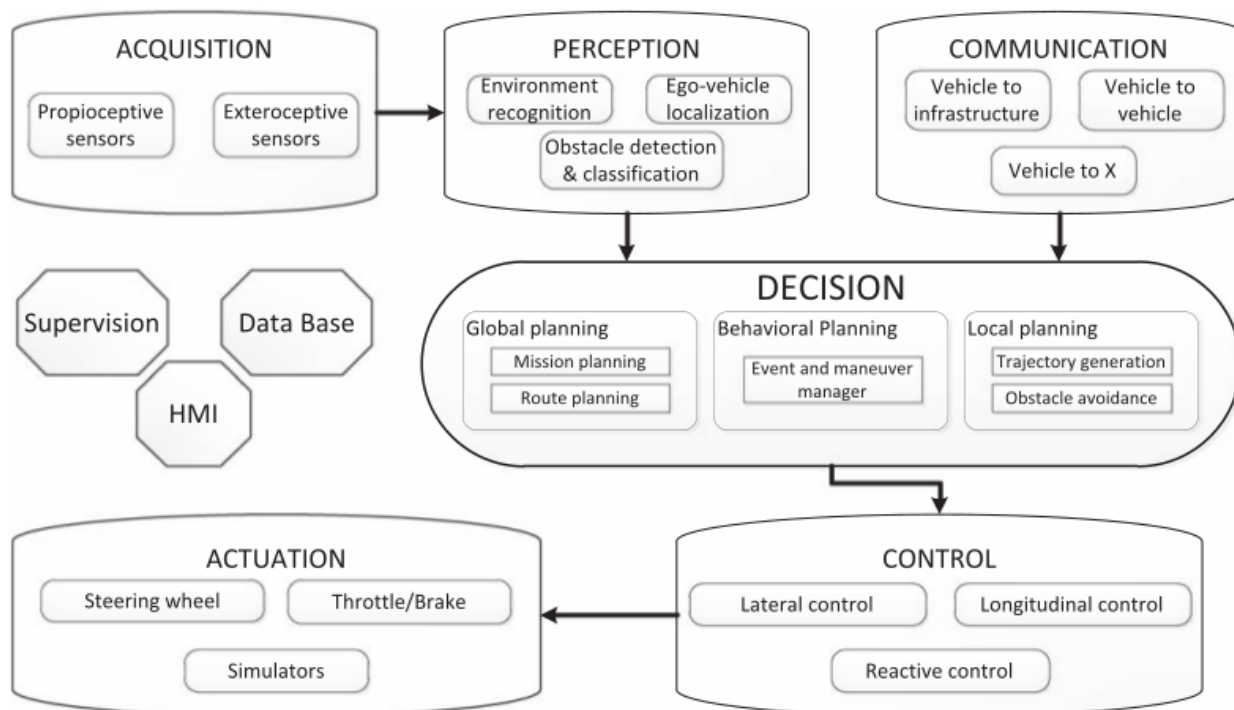


Рисунок 1 – Обобщенная абстракция архитектуры управления автономными автомобилями [3]

Подсистема восприятия обычно состоит из следующих компонентов, состав и взаимодействие которых, в прочем, могут отличаться в зависимости от назначения и уровня автономности беспилотного автомобиля:

- подсистема локализации, осуществляющая определения положения автомобиля с помощью GPS и системы одновременной картографии и навигации (SLAM),
- подсистема выделения дороги (ground detection/road detection), определяющая, какие области пространства вокруг доступны для передвижения,
- подсистема детектирования (object detection) и отслеживания (object tracking) объектов, позволяющая определять положение динамических препятствий вокруг автомобиля и их скорость

и направление движения,

- подсистемы распознавания дорожных знаков, светофоров, дорожной разметки и т.п.

Процесс планирования движения и принятия решений в современных беспилотных автомобилях обычно представлен в виде иерархии: планирование маршрута (route planning), принятие решений (behaviour planning, decision making), локально планирование (local motion planning) и управление с обратной связью [29]. Точное разделение на уровни обычно размыто и может отличаться в различных работах, но общая структура сохраняется. Пример того, как разделяются задачи по уровням иерархии представлен на рисунке 2.

На верхнем уровне осуществляется планирование маршрута по дорожной сети. Затем следуют уровень планирования поведения, который принимает решения и формирует локальные навигационные задачи, которые приближают автомобиль к выполнению высокоуровневой задачи и удовлетворяют правилам дорожного движения. Затем локальный планировщик формирует непрерывный путь в окружающем пространстве, который выполняет локальную навигационную задачу. Система управления с обратной связью осуществляет выполнение запланированного движения и коррекцию ошибок.

Система планирования маршрута должна выбирать как можно более оптимальный путь по дорожной сети от текущего положения к точке назначения. Типичным решением является представление дорожной сети как ориентированного графа, ребрам которого назначаются веса, соответствующие стоимости движения по данному сегменту дороги (в простейшем случае — расстояние). Тогда задача построения маршрута может быть сформулирована как задача нахождения пути на графе с наименьшей стоимостью. Участники Darpa Urban Challenge решали эту задачу самостоятельно, используя хорошо известные алгоритмы, такие как алгоритмы Дейкстры, A\* или их модификации. Это было возможно, потому что область проведения Darpa Urban Challenge была небольшая. В реальных сценариях применение простейших алгоритмов невозможно, потому что дорожная сеть может содержать миллионы узлов. Но это и не требуется, потому что в настоящее время в настоящее время существует большое количество картографических сервисов, реализующих эффективные алгоритмы построения маршрутов вплоть до континентального масштаба и предоставляющих подобную функциональность в виде API. Поэтому в данной работе не рассматривается глобальное планирование маршрута, так как реализовывать его самостоятельно не имеет смысла.

После того, как глобальный маршрут был построен, автономный автомобиль должен уметь двигаться по этому маршруту и взаимодействовать с другими участниками дорожного движения в соответствии с правилами дорожного движения. Планировщик поведения отвечает за выбор подходящего поведения в каждый момент времени, основываясь на наблюдаемой дорожной ситуации. Например, когда автомобиль достигает стоп-линии, планировщик поведения формирует команду остановки. Правила дорожного движения определяют правильное поведение в различных ситуациях. Варианты поведения и возможные дорожные ситуации представляют собой конечные множества, поэтому естественным будет представление планировщика движения в виде конечного автомата. Большинство участников DARPA Urban Challenge применяли конечные автоматы для задачи принятия решений.

Основной проблемой является неопределенность намерений остальных участников дорожного движения. Проблема предсказания намерений участников движения активно изучается и существуют различные подходы, от простой линейно экстраполяции [26] движения до предсказания движения с помощью Deep Learning [ссылка](#), что является наиболее перспективным способом в настоящее время. Неопределенность поведения участников дорожного движения часто учитывается в планировании поведения с помощью вероятностного формализма, например, применяя марковский процесс принятия решений [ссылку](#) [бы](#).

После того, как планировщик поведения определил поведение, которое должно выполняться в данной дорожной ситуации, например, удерживать полосу, выполнит обгон или остановиться на светофоре, выбранное поведение должно быть представлено в виде пути или траектории, которые, в свою очередь, могут быть выполнены низкоуровневым регулятором с обратной связью. Эта траектория должна



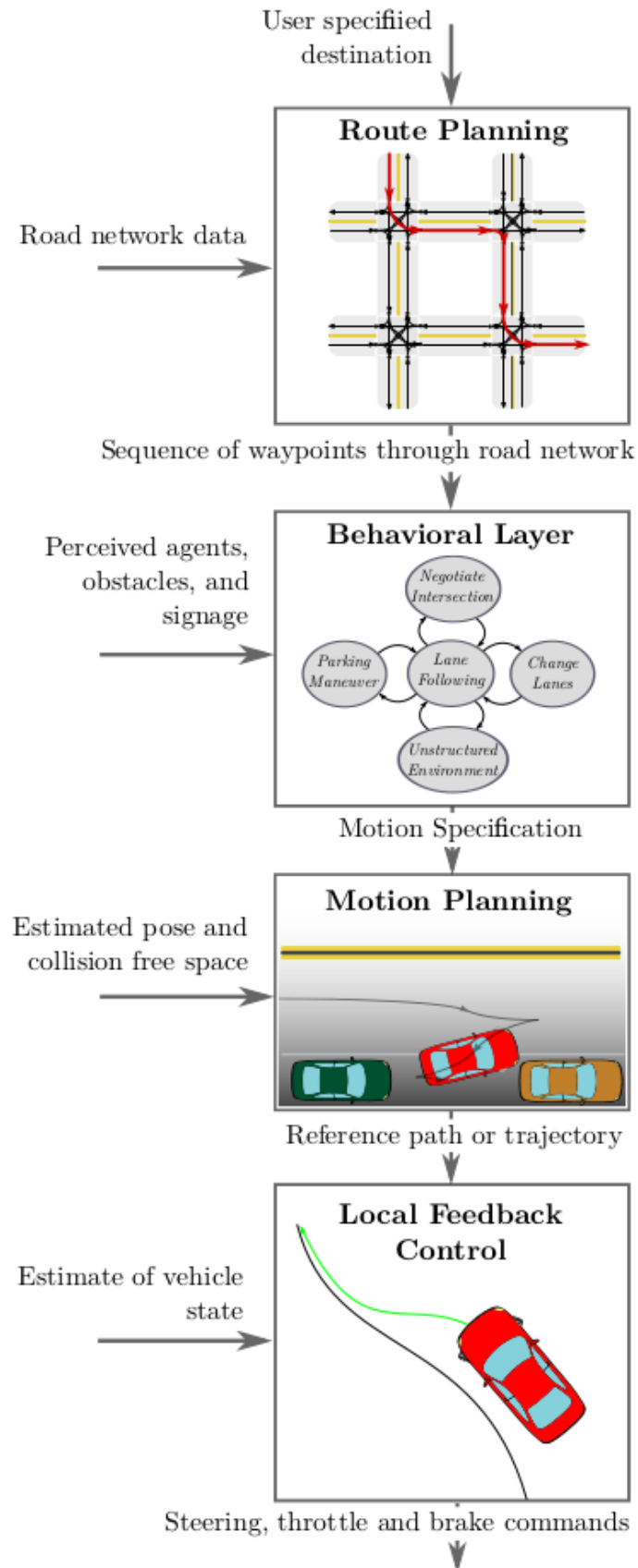


Рисунок 2 – Иерархия системы управления

избегать препятствий, быть достижимой для автомобиля согласно его кинематике и динамике и, наконец, быть комфортной для пассажиров. Эта задача соответствует стандартной задаче планирования движения (motion planning) в робототехнике.

И, наконец, регулятор с обратной связью отвечает за то, чтобы сформировать управляющие сигналы, такие как поворот руля, газ и торможение, чтобы точно следовать программной траектории.

### 1.3 Обзор методов планирования движения

Локальный планировщик движения отвечает за формирование безопасной, достижимой и комфортной для пассажиров траектории из текущего положения автомобиля до следующей цели, определенной планировщиком поведения. В зависимости от ситуации, это могут быть различные цели, такие как удержание полосы, следование на заданном расстоянии от впереди идущего автомобиля, остановка и др. Планировщик движения учитывает информацию о статических и динамических препятствиях, чтобы сгенерировать траекторию, которая будет избегать препятствий и будет выполнима с точки зрения кинематики и/или динамики автомобиля. В некоторых подходах рассматривается формирование оптимальной по отношению к некой целевой функции траектории.

Получаемое программное движение может представлено в виде пути или траектории. В случае планирования пути, результатом будет путь в конфигурационном пространстве автомобиля,

Задача планирования движения (motion planning) в робототехнике — процесс разбиения требуемого движения на дискретные действия, которые удовлетворяют ограничениям. Планирование движения является важной задачей робототехники и активно исследуется и разрабатывается в течении последних десятилетий. Разработано большое количество алгоритмов планирования движения, как для решения задачи в общем случае, так и для конкретных приложений. В этом разделе дается обзор методов, позволяющих осуществить задачу локального планирования беспилотного движения автономного автомобиля.

Существует большое количество методов, позволяющих решить данную задачу, такие как методы поиска на графах, случайные методы поиска (sample-based motion planning), методы интерполяции движения кривыми, методы численной оптимизации и другие. Обзор методов планирования движения в приложении к беспилотным автомобилям рассмотрен в работах [motion\_planning\_survey], [3], [37], [38], [29].

Различные способы планирования движения имеют свои преимущества и недостатки. Например, одни методы позволяют более полно решать задачу планирования движения, но являются более вычислительно сложными, что затрудняет их применение в реальном времени на дорогах с быстро меняющейся ситуацией. Чтобы решить эту проблему, многие команды в DARPA Urban Challenge не ограничивались одним способом планирования движения. Распространенным подходом было применять один планировщик движения при движении по дорогам, а другой - при движении в неструктурированном окружении, например, при задаче парковки.

Задача планирования движения в применении к автомобилю в общем виде является нетривиальной, потому что динамика автомобиля описывается сложными дифференциальными уравнениями, имеющими, что немаловажно, негOLONOMНЫЕ связи [36]. Многие исследователи применяют различные упрощенные модели динамики автомобиля [30]. Очень распространенной является "велосипедная" модель динамики автомобиля, которая применяется в ряде работ [21], [25], [35].

#### 1.3.1 Формулирование задачи планирования движения

Перед рассмотрением алгоритмов планирования движения необходимо определить несколько понятий, которые применяются в теории автоматизированного управления и сформулировать задачу планирования движения.

Конфигурационное пространство  $C$  системы — это совокупность всех ее обобщенных координат. Дано:

- пространство конфигураций  $C$  робота;
- множество конфигураций, соответствующее препятствиям  $C_o \in C$ , соответственно множество свободных конфигураций  $C_{free} = C \setminus C_o$ ;
- начальные и конечные конфигурации  $q_s, q_g$ .

Задача планирования движения может быть сформулирована как планирование пути или планирование траектории. В первом случае, требуется найти функцию  $\tau(\alpha) : [0, 1] \rightarrow C_{free}$ , представляющую путь робота в конфигурационном пространстве, где  $\tau(0) = q_s, \tau(1) = q_g$ . В этом случае, решение не определяет, как этот пройден автомобилем. Планировщик движения может выбрать профиль скорости для этого пути, как это сделано в [8], или делегировать эту задачу нижележащим подсистемам.

В случае планирования траектории, явно учитывается время движение. В таком случае требуется найти функцию  $\pi(t) : [0, T] \rightarrow C$ , где  $T$  - горизонт планирования. В отличие от планирования пути, планирование траектории явно указывает, как должна меняться конфигурация с течением времени.

Требуется это более внятно и корректно сформулировать. Спросить Горобцова. И буквы W, C, должны быть "гнутыми но я не знаю, как они называются

### 1.3.2 Методы поиска на графах

Для решения задачи планирования движения можно применять алгоритмы поиска на графах, такие как алгоритм Дейкстры, алгоритм  $A^*$  или их модификации. Для применения алгоритмов поиска на графах к задаче планирования движения необходимо сначала представить конфигурационное пространство в виде графа.

В данном методе большую роль играет способ построения графа, а именно выбор опорных точек и ребер, которые их соединяют.

Простейшими, но крайне широко распространенными вариантами представления конфигурационного пространства в виде графа являются методы клеточной декомпозиции. Наиболее распространенным вариантом является приближенная клеточная декомпозиция, когда пространство разбивается с помощью равномерной сетки (grid map) [37]. Преимуществом этого метода является то, что его легко реализовать и легко представить конфигурацию окружающего пространства в виде сетки. Этот метод широко применяется в ряде существующих реализаций поиска пути, таких как navigation stack в ROS [ссылка](#). Недостатком этого метода является существенное увеличение размера графа при уменьшении шага дискретизации, увеличении области, в которой осуществляется планирование и, особенно, при увеличении размерности конфигурационного пространства. Подобные методы в основном применяются для геометрического планирования на плоскости, с увеличением числа степеней свободы (повороты, трехмерное пространство, трехмерное пространство с поворотами) количество вершин в графе становится слишком большим для эффективного применения данного метода.

Другим способом выбора опорных точек графа является использование диаграммы видимости. [37] Диаграмма видимости определяется как неориентированный граф, вершины которого включают множество вершин препятствий, а ребра соединяют некоторые вершины таким образом, что никакое ребро не пересекается с препятствиями. Для использования диаграммы видимости необходимо, чтобы препятствия имели форму многоугольника или многогранника. Поскольку путь строится по вершинам препятствий, и часть пути совпадает с краями препятствий, существует определенная опасность столкновения с препятствиями. Кроме того, при увеличении количества препятствий возрастает сложность графа, причем этот рост очень быстрый. Подобный подход редко применяется для беспилотных автомобилей [пруф](#).

Еще одним методом построения графа, который сильно распространен в задачах планирования [35], [37], являются диаграммы Вороного. Диаграмма Вороного конечного множества точек на плоскости

представляет собой разбиение плоскости таким образом, что каждая область образует геометрическое место точек (локус), каждая из которых более близка к одному элементу множества, чем к другому. Преимуществом диаграмм Вороного для планирования движения является то, что с их помощью можно построить путь, наиболее удаленный от всех препятствий, и, следовательно, наиболее безопасный.

Команда "BOSS" Darpa Urban Challenge применяла метод поиска на графах для планирования движения [8], [12], [9]. Для графа они применяют т.н. решетку состояний (state lattice). Узлы графа представляют собой дискретизированные состояния в пространстве конфигураций, а каждое ребро графа представляет достижимый путь. В отличие от более распространенных подходов, применяющих сетки, клетки которых просто имеют 4 или 8 соединенных соседей, в этом подходе гарантируется, что полученный путь будет достижимым.

Наиболее применительны методы поиска на графах для планирования движения в неструктурированном окружении.

Далее будут рассмотрены некоторые модификации известных алгоритмов поиска пути на графах, ориентированные на нахождение кинематически и динамически достижимых путей.

### 1.3.3 Hybrid A\*

Из-за нелинейной динамики автомобиля, путь, построенный с помощью поиска на регулярной прямоугольной дискретной сетке будет кинематически и динамически недостижим для автомобиля. Было предложено много вариантов решения этой проблемы, так например, алгоритм Hybrid A\* [5], предложенный командой "Junior" DUC [16]. В отличие от алгоритма A\*, примененного к регулярной сетке, имеющий 4 или 8 дискретных соединений между соседними ячейками, Hybrid A\* формирует набор траекторий, учитывающих нелинейную динамику и непрерывную природу движения автомобиля (рисунок 3).

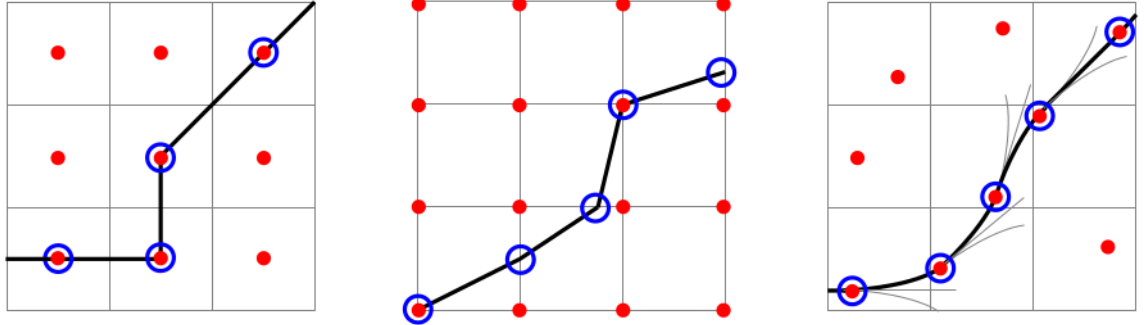


Рисунок 3 – Hybrid A\* в сравнении с A\* [16]

Алгоритм работает с дискретными сетками в трехмерном пространстве состояний  $x, y, \theta$ , где  $x, y$  - положение автомобиля, а  $\theta$  - ориентация. Рассмотрим работу алгоритма. Допустим, что в некий момент времени состояние автомобиля  $\langle x, y, \theta \rangle$  и оно соответствует ячейке  $c_i$  в дискретизированном пространстве состояний. Сопоставим с этой ячейкой непрерывные координаты:  $x_i = x, y_i = y, \theta_i = \theta$ . Далее решаем задачу прямой динамики, интегрируя движение автомобиля при подаче управления  $u$  в течении заданного времени  $t$ . Конечным состоянием будет  $\langle x', y', \theta' \rangle$ , попадающее в дискретную клетку  $c_j$ . Аналогичным образом сопоставим непрерывные координаты с этой клеткой:  $x_j = x', y_j = y', \theta_j = \theta'$ . Это позволяет гарантировать, что мы имеем непрерывную траекторию из клетки  $c_i$  и  $c_j$ , для которой гарантируется кинематическая и динамическая достижимость (разумеется, при использовании достаточно точной динамической модели), а также соответствующее управление  $u$ , что не гарантируется для обычного A\* алгоритма. Дальнейший поиск по этому графу осуществляется аналогично обычному алгоритму A\*.

### 1.3.4 State Lattice

Другой широко распространенной модификацией графовых алгоритмов является семейство алгоритмов, называемых в англоязычной литературе "State Lattice" (решетка состояний). Алгоритм был предложен в работе [27] и близок по своей идее к алгоритму Hybrid A\*, рассмотренному ранее. В работе представлено пространство поиска, называемое решеткой состояний, позволяющее представить планирование неголономного движения как задачу поиска на графе.

Решетка состояний представляет собой дискретное множество всех достижимых конфигураций системы. Ее построение осуществляется путем дискретизирования пространства состояний в виде многомерной решетки, состоящих из достижимых состояний, и попытки соединить начальное состояние со всеми узлами решетки с помощью достижимых путей в качестве ребер. В общем случае, решетка состояний содержит все достижимые пути (с заданной дискретизацией), что подразумевает, что если автомобиль можете переместиться из одного состояния в другое, то решетка состояний содержит последовательность путей для осуществления этого маневра.

Подобно обычной сетке, решетка состояний преобразует задачу планирования в непрерывном пространстве в задачу формирования последовательности выбора из нескольких дискретных альтернативных состояний, но в отличие от обычной сетки, решетка состояний строится таким образом, что ее связи являются достижимыми путями.

Построение решетки состояний требует решение обратной задачи для нахождения достижимого пути между двумя заданными состояниями. Существует ряд методов, позволяющих решить эту задачу. Несмотря на то, что метод решетки состояний по-прежнему требует решения обратной задачи, что, как было замечено выше, является нетривиальной задачей, этот метод существенно упрощает задачу планирования движения, потому что позволяет осуществлять планирование движения в виде графа в условиях наличия различных препятствий, позволяя формировать сложную траекторию, что существенно более затруднено при применении чисто математических методов решения этой задачи. В данном случае требуется решать обратную задачу лишь для некоего ограниченного множества возможных комбинаций начальных и конечных состояний.

Другим немаловажным свойством этого метода является то, что в силу использования регулярной сетки, можно выделить набор состояний, инвариантный к пространственному перемещению (рисунок 4). Это позволяет заранее рассчитать ограниченный набор состояний и соответствующих траекторий и управлений с помощью точных моделей и ресурсозатратных алгоритмов, а в процессе планирования движения в реальном времени пользоваться этой заранее рассчитанной таблицей. Этот подход широко применяется в ряде разработок, например, в автомобиле "BOSS" DUC [8].

Важным понятием в этом методе является понятие эквивалентности путей. Путь  $\tau_1$  считается эквивалентным пути  $\tau_2$ , если  $\tau_1$  содержится в определенном регионе  $Q$  вокруг  $\tau_2$  (рисунок 5). Все пути, являющиеся эквивалентными, представляются одним путем.

Для достижения эффективности, как было сказано выше, возможно дублирование некоего базового набора состояний и траекторий для разных начальных состояний, таким образом позволяя возможность эффективно конструировать граф необходимого размера в реальном времени. Для реализации такой возможности во всем множестве путей, которое представляет собой решетка состояний, необходимо выделить базовое множество. Например, для обычной регулярной сетки, каждая ячейка которой соединена с четырьмя соседними, базовым множеством путей будет движение на одну клетку вперед, назад, влево и вправо. Этого простого набора путей достаточно, чтобы описать любые перемещения в этом пространстве состояний. Для решетки состояний применяется подобная идея. Любой путь через решетку может быть декомпозирован в последовательность путей из некоего минимального набора таким образом, что эта последовательность будет эквивалентной исходному пути.

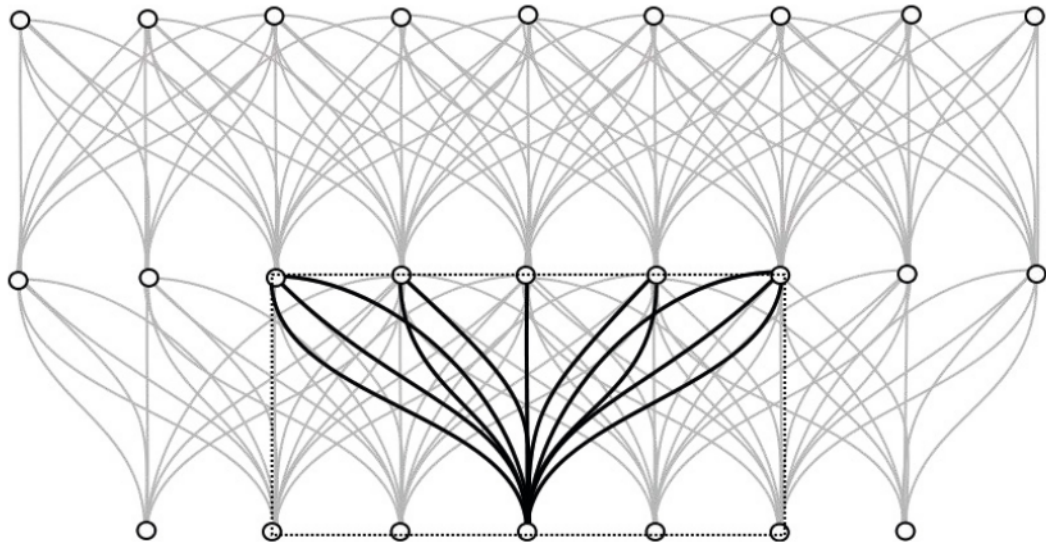


Рисунок 4 – Пример регулярной решетки состояний. Базовое множество управлений и результирующих путей (черный) не зависит расположения и может быть продублировано. Это позволяет очень эффективно представлять множество траекторий и решать проблему планирования движения [13]

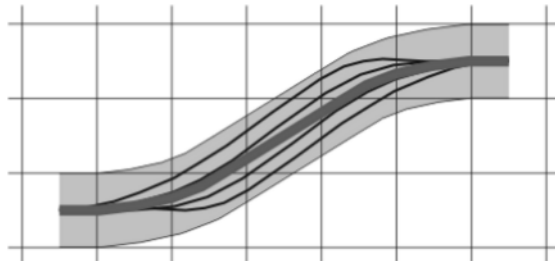


Рисунок 5 – Эквивалентность путей. Множество различных путей (черные линии), которые содержатся в пределах заданного региона (серая область), считаются эквивалентными

### 1.3.5 Случайные методы

При решении задач планирования движения в пространствах высоких размерностей, которые часто возникают в современной робототехнике (например, планирование движения твердого тела в трехмерном пространстве с шестью степенями свободы или планирование движения многозвенного манипулятора), классические методы, такие как методы поиска на графах, обладают очень большой вычислительной сложностью. Для решения таких задач перспективными являются случайные (sample-based) методы, такие как Probabilistic roadmap, Rapidly-exploring random graph (RRG), Rapidly-exploring random tree (RRT) [19], Optimal Rapidly-exploring random tree (RRT\*)[17].

Идея метода заключается в том, что начиная из начального состояния строится дерево, которое может достичь конечного состояния, путем выбора случайных точек в конфигурационном пространстве. Алгоритм RRT представлен в листинге 1, а иллюстрация метода *EXTEND* на рисунке 6.

Особенность этого семейства алгоритмов, как и многих других итерационных методов, является их сходимость в пределе **я встречал более правильный термин, но потерял**, т.е. решение будет получено при количестве итераций стремящемся к бесконечности. Тем не менее, практическое использование этого семейства алгоритмов позволяет быстро находить решение для высоких размерностей. Это свойство является преимуществом для систем реального времени, которым является беспилотный автомобиль. Для систем реального времени ключевой является способность выполнять работу в строго определенный интервал времени. К примеру, под планирование локальной траектории вперед на заданный горизонт

```

function RRT( $x_{init}$ )
   $\tau.init(x_{init})$ 
  for  $k = 1$  to  $K$  do
     $x_{rand} \leftarrow RANDOM\_STATE()$ 
    EXTEND( $\tau, x_{rand}$ )
  end for
end function
function EXTEND( $\tau, x$ )
   $x_{near} \leftarrow NEAREST\_NEIGHBOUR(x, \tau)$ 
  if NOT_COLLIDED( $x, x_{near}, x_{new}$ ) then
     $\tau.add\_vertex(x_{new})$ 
     $\tau.add\_edge(x_{near}, x_{new}, u_{new})$ 
    if  $x_{new} = x$  then
      return REACHED
    else
      return ADVANCED
    end if
  end if return TRAPPED
end function

```

---

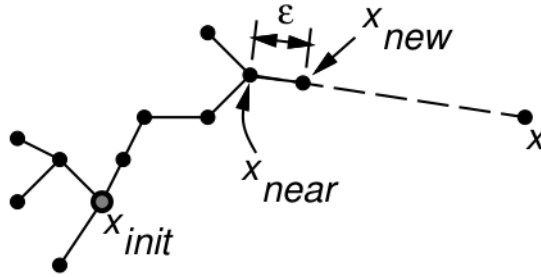


Рисунок 6 – Пример работы метода EXTEND алгоритма RRT

планирования отводится 0.5 секунды, и превышение этого времени может привести к нарушению функционирования беспилотного автомобиля, что может создать опасную ситуацию. В случае с RRT алгоритмами, указывается максимальное время работы, после которого итерационный процесс прекращается. В этом случае, полное решение может быть не найдено, но планирование может быть повторено через некоторое время, чтобы построить недостающий участок пути.

Для применения этого алгоритма достаточно определить процедуру проверки на пересечение с препятствием NOT\_COLLIDED.

Существует большое количество модификаций алгоритма RRT, особенно необходимо отметить алгоритм RRT\* [17], являющийся асимптотически-оптимальной версией алгоритма RRT. Он основан на алгоритме rapidly-exploring random graph (RRG) и имеет отличающуюся процедуру добавления новой точки в граф. В отличие от алгоритма RRT, который прекращает свою работу, когда решение было найдено, алгоритм RRT\* продолжает работу, постепенно находя все более оптимальные решения.

Отдельно стоит выделить направление под названием кинодинамическое планирование (kinodynamic planning), которое является актуальным в последнее время. В этом случае планирование осуществляется не в конфигурационном пространстве, а в пространстве состояний (фазовом пространстве) [18], которое определяется следующим образом. Пусть  $\mathcal{C}$  — конфигурационное пространство, каждая конфигурация  $q \in \mathcal{C}$  представляет положение робота в пространстве. Обозначим пространство состояний как  $\mathcal{X}$ , в котором состояние  $x \in \mathcal{X}$  определяется как  $x = (q, \dot{q})$ . (Состояние может определяться и с использованием производных высших порядков).

Это позволяет осуществлять планирование не только с учетом кинематической модели автомобиля, но и с учетом динамической модели с дифференциальными ограничениями и неголономными

связями. Таким образом, результирующие пути будут достижимы с точки зрения динамики автомобиля, что не всегда возможно для других методов планирования, что приводит к необходимости вводить различные искусственные ограничения. Результатом такого метода планирования может быть не только траектория в пространстве состояний, которая подается на вход низкоуровневого регулятора с обратной связью, но и непосредственно последовательность управляющих сигналов, которые приведут систему к движению с такой траекторией.

Для реализации этих алгоритмов необходимо определить функции *PROPAGATE* и *STEER*. Функция *PROPAGATE* (1) формирует следующее состояние системы после применения к ней заданного управляющего воздействия, например, путем численного интегрирования.

$$x_{i+1} = \text{PROPAGATE}(x_i, u_i, \Delta t) \quad (1)$$

где  $x_i$  — текущее состояние,  
 $u_i$  — управление,  
 $\Delta t$  — время интегрирования.

Функция *STEER* (2) значительно сложнее. Согласно алгоритму RRT необходимо осуществлять шаг из некоего состояния по направлению к случайно выбранному состоянию, то в случае с кинодинамическим планированием это невозможно сделать напрямую, так как для перехода между состояниями используется функция *PROPAGATE*, реализующая динамическую модель. Для этого потребуется определить, какие управляющие сигналы могут привести в требуемое состояние, т.е. решить обратную задачу динамики. В общем случае это невозможно. В зависимости от задачи и модели, может выбираться различная функция *STEER*, реализующая обратную задачу с той или иной точностью. Благодаря итерационному алгоритму, решение может быть найдено даже с использованием приближенной функции. В простейшем случае, в качестве функции *STEER*, может использоваться сэмплирование случайного управляющего вектора. В этом случае, сходимость алгоритма существенно ухудшается и потребуется очень много итераций, чтобы прийти к решению.

$$u^* = \text{STEER}(x_{near}, x, \Delta t) \quad (2)$$

где  $x$  — случайно выбранное состояние,  
 $x_{near}$  — состояние, близкое к случайному,  
 $u^*$  — примерное управление,  
 $\Delta t$  — время интегрирования.

Интересное решение этой проблемы было предложено командой MIT Darpa Urban Challenge [1], [22], [23]. Они применяли RRT для планирования системы модель автомобиль-регулятор с обратной связью, т.е. использовали модель управления по замкнутому контуру для планирования движений, в то время как в большинстве решений применяется управление по открытому контуру, т.е. учитывается только модель автомобиля.

Таким образом, sample-based методы планирования движения являются очень перспективными и способными к решению сложных задач планирования движения, где не справляются другие методы. Тем не менее, эти методы не лишены ряда существенных недостатков, главным из которых является высокая вычислительная сложность. В зависимости от сложности модели, размерности пространства, конфигурации препятствий, могут потребоваться десятки или сотни тысяч итераций, чтобы найти решение, что может не укладываться в жесткие временные рамки для системы управления беспилотным автомобилем. При типичном применении кинодинамического RRT, происходит выбор (сэмплирование) точек в пространстве управления автомобиля, т.е. входов для органов управления автомобилем. В этом подходе применяется сэмплирование входов для регулятора с обратной связью. Алгоритм RRT формирует дерево со сравнительно низкой плотностью сэмплированных точек, затем происходит моделирование системы автомобиль-регулятор, формируя реальную траекторию автомобиля (рисунок 7). Этот подход



позволяет уменьшить количество вершин дерева, которое приходится открывать алгоритму RRT, тем самым уменьшив вычислительную сложность.

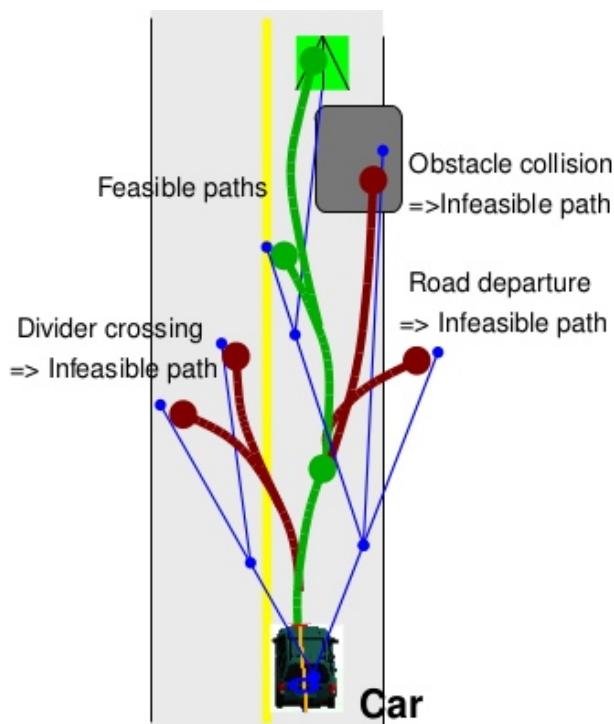


Рисунок 7 – Планирование движения с помощью RRT и регулятора с обратной связью

Существует Open-Source библиотека The Open Motion Planning Library [31], которая реализует sample-based алгоритмы поиска, таких как RRT, RRT\* и ряд других, основанных на них, как для планирования в пространстве конфигураций, так и для планирования в пространстве состояний (kinodynamic planning). Библиотека легко интегрируется с любой задачей планирования движения, поскольку не имеет никаких ограничений на тип задачи, способы представления препятствий, используемые динамические модели. Программисту достаточно определить вышеперечисленные функции *NOT\_COLLIDED*, *PROPAGATE*, *STEER* в соответствии с используемым способом представления препятствий и моделью робота. Библиотека имеет ряд готовых реализаций для широко распространенных пространств конфигурации, состояний и управления, таких как  $SO(2)$ ,  $SO(3)$ ,  $SE(2)$ ,  $SE(3)$ , но возможно определить и свои.

Возможно, куда-то надо написать еще про Dubins и про потенциалы

### 1.3.6 Методы интерполяции кривыми

Следующей группой методов, применяемых для реализации задачи планирования движения беспилотного автомобиля, является интерполяция траектории движения с помощью каких-либо кривых или иных линий.

Эти алгоритмы принимают набор точек (например, данный набор точек пути, описывающих глобальную дорожную карту), генерируя новый набор данных (более плавный путь) удовлетворяющий требованиям на непрерывность траектории, ограничения транспортного средства и учитывая динамическую среду, в которой движется автомобиль.

В отличие от методов, рассмотренных ранее, которые являются универсальными и могут применяться к решению различных задач планирования движения, применительно к беспилотному автомобилю, это задача движения в неструктурированном окружении, эти методы широко применяются для построения траектории при движении по дороге.

Планировщики интерполяционных кривых реализуют различные методы сглаживания траектории и генерации кривых. Могут применяться различные кривые для интерполяции, такие как линии и окружности, клотоиды, кривые Безье или полиномы. Полиномы зачастую применяются для того, чтобы построить участок траектории, удовлетворяющий граничным условиям (например, положению и скорости).

Так в работах [33], [26], опубликованных по результатам участия команды Стэнфордского университета в Darpa Urban Challenge, рассматривается планирование траектории с помощью полиномов пятого порядка в подвижной системе координат, связанной с желаемой траекторией, основанной на трехграннике Френе. В этой работе осуществляется независимое формирование продольной и поперечных траекторий, обе представлены в форме полиномов пятого порядка. Алгоритм генерирует набор кривых путем варьирования конечных условий, а затем осуществляется выбор оптимальной траектории с использованием различных функций стоимости.

Стоит отметить, что формирование большого набора траекторий с последующим выбором оптимальной является распространенным подходом в решении задачи планирования движения.

В работе [2] представлен похожий подход, отличающийся тем, что после нахождения оптимальной траектории из набора дискретных траекторий, осуществляется дополнительная процедура оптимизации параметров полиномов с целью получить более оптимальную траекторию.

В работе [20] применяются кривые Безье пятого порядка для формирования траектории движения автомобиля с моделью Аккермана. Алгоритм получает на вход набор путевых точек и формирует путь, проходящий через них с помощью набора кривых Безье. Каждый сегмент определяется положением точки, первой и второй производной, т.е. скоростью и ускорением автомобиля. Полученная кривая имеет гладкие стыки между сегментами. Затем осуществляется оптимизация кривой путем изменения положений и первых производных в опорных точках, чтобы оптимизировать длину пути.

#### 1.4 Обзор некоторых систем управления беспилотными автомобилями

В этом разделе будут более детально рассмотрены некоторые решения в области планирования движения беспилотных автомобилей, в основном, на основе публикаций участников Darpa Urban Challenge.

##### 1.4.1 Команда Junior в Darpa Urban Challenge

В работах [33], [26], опубликованных по результатам участия команды Стэнфордского университета в Darpa Urban Challenge, рассматривается планирование траектории с помощью полиномов пятого порядка в подвижной системе координат, связанной с желаемой траекторией, основанной на трехграннике Френе. Иллюстрация этого способа представлена на рисунке 8, где  $s(t)$  — длина дуги, пройденная автомобилем,  $\vec{t}_r$  — направляющий вектор системы координат,  $\vec{n}_r$  — нормальный вектор системы координат,  $\vec{x}(s, d)$ ,  $\vec{n}_x$ ,  $\vec{t}_x$  — положение и направляющие векторы локальной системы координат автомобиля соответственно и  $d(t)$  — отклонение автомобиля от желаемой траектории.

Планирование осуществляется независимо для продольного и поперечного движения, а затем полученные полиномы объединяются в траекторию и переводятся в декартову систему координат. Выбирается оптимальная траектория, минимизирующая функцию стоимости. За основу функции стоимости взят интеграл третьей производной, рывка (jerk):

$$J = \int_{t_0}^{t_1} \ddot{\ddot{x}}^2 d\tau \quad (3)$$

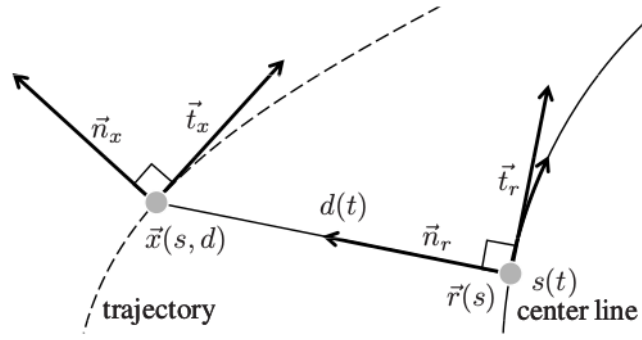


Рисунок 8 – Система координат для планирование траектории

где  $p$  — положение (продольное или поперечное),

$t_0, t_1$  — время совершения маневра.

Полиномы пятого порядка выбраны по причине того, что оптимальным решением для минимизации этого интеграла.

Так как при планировании движения требуется учитывать препятствия, задача оптимизации для нахождения коэффициентов полинома, минимизирующего стоимость крайне затруднительно. Вместо этого в работе предложен другой подход, являющийся типичным для подомных методов. Формируется набор траекторий путем варьирования конечных условий, а затем из них выбирается траектория, которая лучше всего минимизирует функцию стоимости и, при этом, удовлетворяет ограничениям, например:

$$D_0 = [d_0, \dot{d}_0, \ddot{d}_0] \quad (4)$$

$$D_{1ij} = [d_i, 0, 0, T_j] \quad (5)$$

где  $D_0, D_1$  — начальное и конечное состояние,

$d_0, \dot{d}_0, \ddot{d}_0$  — текущие поперечные положение, скорость и ускорение автомобиля,

$d_i$  — конечное поперечное положение,

$T_j$  — время выполнения маневра.

Пример планирования поперечных и продольных траекторий представлены на рисунке 9 и 10 соответственно.

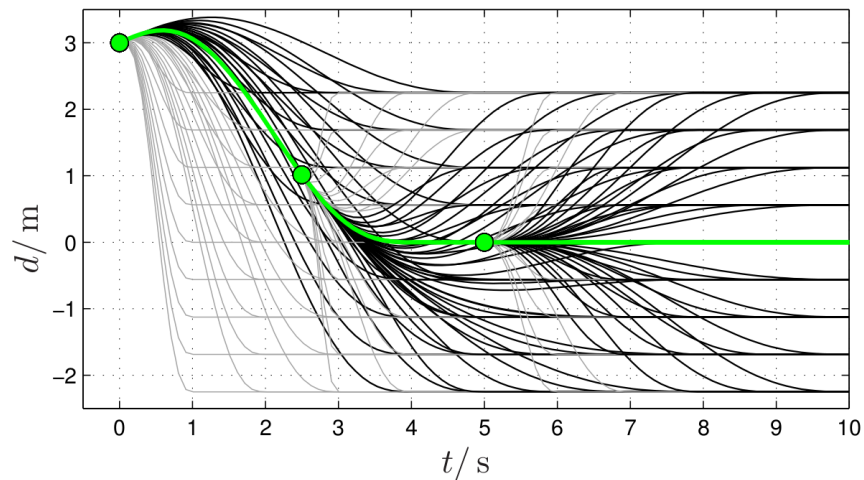


Рисунок 9 – Планирование поперечного движения, зеленая — оптимальная траектория, черная — валидные траектории, серые — невалидные траектории

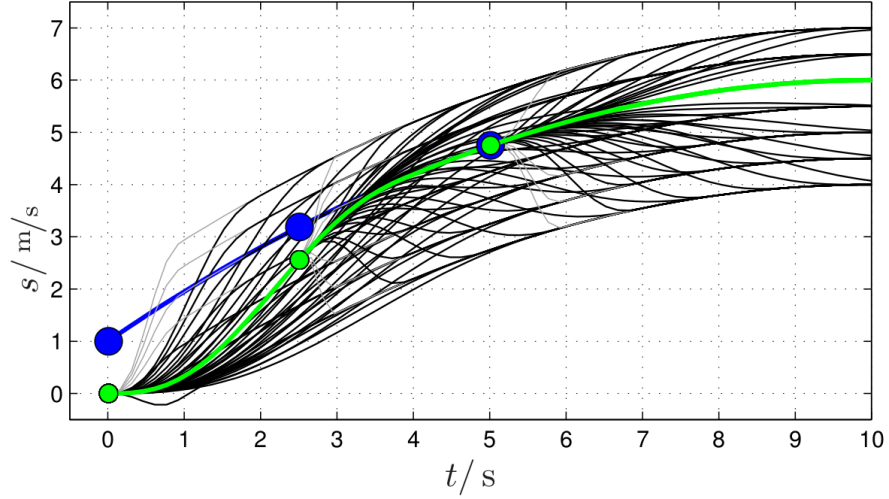


Рисунок 10 – Планирование продольного движения

#### 1.4.2 Команда BOSS в Darpa Urban Challenge

Команда BOSS университета Карнеги-Меллона использовала [8], [12], [9] несколько иной подход, основанный на методе решеток состояния (state lattice). Планировщик траектории формирует набор траекторий-кандидатов, из которых затем выбирается лучшая, удовлетворяющая ограничениям и функции стоимости.

Каждая траектория-кандидат рассчитывается с помощью MPC (Model Predictive Control) генератора [15], который формирует динамически достижимый путь между начальным и конечным состояниями. Этот алгоритм может быть использован для формирования набора параметризованных последовательностей управления  $[u(p, x)]$ , которые удовлетворяют набору ограничений в виде дифференциальных уравнений вида:

$$\dot{x}(x, p) = f(x, u(p, x)) \quad (6)$$

где  $x$  - состояние автомобиля (положение  $x, y$ , ориентация  $\theta$ , кривизна траектории  $k$ , скорость  $v$ ),  $p$  - набор параметров.

Авторы вводят набор ограничений, которые описываются как разница между целевым состоянием  $X_c$  и конечной точкой траектории, получаемой путем интегрирования уравнений динамики автомобиля:

$$x_c = \langle x_c, y_c, \theta_c \rangle^T \quad (7)$$

$$x_f(p, x) = x_i + \int_0^{t_1} \dot{x}(x, p) dt \quad (8)$$

$$C(x, p) = x_c - x_f(p, x) \quad (9)$$

Путем минимизации ошибки  $C(x, p)$  находятся такие коэффициенты  $p$ , что полученная траектория как можно ближе подходит к заданному конечному состоянию.

Траектории определяются функцией скорости  $v(p, t)$  и функцией кривизны  $k(p, s)$ , где  $s$  - покрытая длина дуги. В работе применен ряд различных функций, реализующих различные профили скорости, принимаемые в разных ситуациях (рисунок 11a). Функция кривизны определяется в виде сплайна второго порядка и задается набором трех опорных точек (рисунок 11b). Каждая траектория определяется набором параметров, которые подбираются в процессе оптимизации.

Траектории рассчитываются заранее и формируют look-up-таблицу, с помощью которой опреде-

ляются подходящие параметры траектории для состояния, близкого к текущему состоянию автомобиля. Это существенно ускоряет операцию планирования движения.

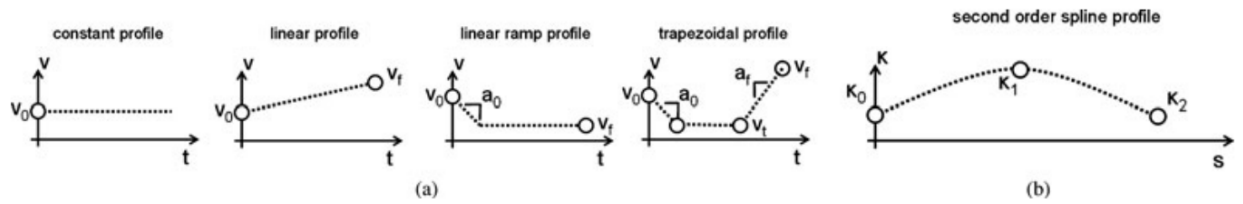


Рисунок 11 – Профили скорости автомобиля BOSS

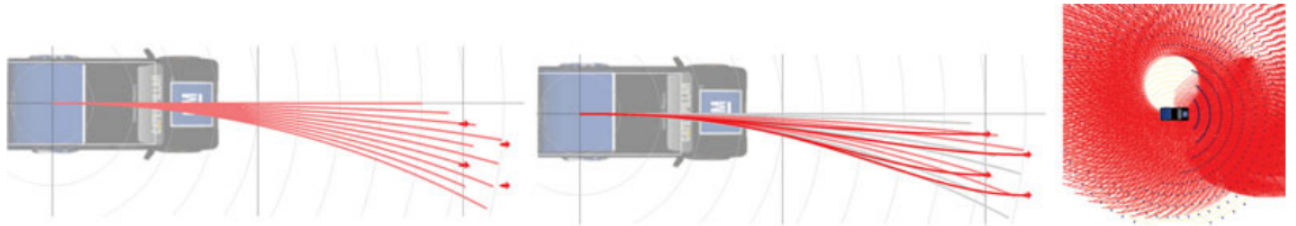


Рисунок 12 – Предварительно рассчитанные траектории автомобиля BOSS

#### 1.4.3 Команда Talos в Darpa Urban Challenge

#### 1.4.4 Команда AnnieWAY в Darpa Urban Challenge

## 2 ПРОЕКТИРОВАНИЕ СИСТЕМЫ УПРАВЛЕНИЯ ДВИЖЕНИЕМ БЕСПИЛОТНОГО АВТОМОБИЛЯ

2.1 Проектирование общей архитектуры системы управления беспилотным автомобилем

2.2 Проектирование подсистемы планирования поведения

2.3 Проектирование подсистемы планирования траектории

2.4 Проектирование подсистемы следования траектории

### 3 РЕАЛИЗАЦИЯ СИСТЕМЫ УПРАВЛЕНИЯ ДВИЖЕНИЕМ БЕСПИЛОТНОГО АВТОМОБИЛЯ

#### 3.1 Использование ROS в качестве основы для системы управления

#### 3.2 Реализация подсистемы цифровой карты

#### 3.3 Реализация подсистемы планирования поведения

#### 3.4 Реализация подсистемы планирования траектории

#### 3.5 Реализация подсистемы следования траектории

## 4 ЭКСПЕРИМЕНТЫ И ОЦЕНКА РЕЗУЛЬТАТОВ РАБОТЫ

### 4.1 ???



## ЗАКЛЮЧЕНИЕ

## Список литературы

1. A Perception-Driven Autonomous Urban Vehicle / J. J. Leonard [и др.] // Journal of Field Robotics. — 2008. — Окт. — Т. 25. — С. 727—774. — DOI: 10.1002/rob.20262.
2. A real-time motion planner with trajectory optimization for autonomous vehicles / Wenda Xu [и др.] // 2012 IEEE International Conference on Robotics and Automation. — 05.2012. — С. 2061—2067. — DOI: 10.1109/ICRA.2012.6225063.
3. A Review of Motion Planning Techniques for Automated Vehicles / D. Gonzalez Bautista [и др.] // IEEE Transactions on Intelligent Transportation Systems. — 2015. — Ноябрь. — С. 1—11. — DOI: 10.1109/TITS.2015.2498841.
4. An Automated Electric Vehicle Prototype Showing New Trends in Automotive Architectures / M. Buechel [и др.] // 2015—October. — 2015. — С. 1274—1279. — DOI: 10.1109/ITSC.2015.209. — URL: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-84950299857&doi=10.1109%2fITSC.2015.209&partnerID=40&md5=05b753f90c370fcc42beca54c13cc704> ; cited By 20.
5. Application of Hybrid A\* to an Autonomous Mobile Robot for Path Planning in Unstructured Outdoor Environments / J. Petereit [и др.] // ROBOTIK 2012; 7th German Conference on Robotics. — 05.2012. — С. 1—6.
6. Automotive radar the key technology for autonomous driving: From detection and ranging to environmental understanding / J. Dickmann [и др.] // 2016. — DOI: 10.1109/RADAR.2016.7485214. — URL: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-84978224598&doi=10.1109%2fRADAR.2016.7485214&partnerID=40&md5=f4ceac54185166a514a6173971a58cd3> ; cited By 18.
7. Automotive Radars: A review of signal processing techniques / S. Patole [и др.] // IEEE Signal Processing Magazine. — 2017. — Т. 34, № 2. — С. 22—35. — DOI: 10.1109/MSP.2016.2628914. — URL: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-85032778479&doi=10.1109%2fMSP.2016.2628914&partnerID=40&md5=49e42c43a5bd6911f206182ffb5ac4ba> ; cited By 35.
8. Autonomous driving in urban environments: Boss and the Urban Challenge // Journal of Field Robotics Special Issue on the 2007 DARPA Urban Challenge, Part I. — 2008. — Июнь. — Т. 25, № 8. — С. 425—466.
9. Brock O., Trinkle J., Ramos F. Planning Long Dynamically-Feasible Maneuvers for Autonomous Vehicles // Robotics: Science and Systems IV. — MITP, 2009. — ISBN 9780262258623. — URL: <https://ieeexplore.ieee.org/document/6284856>.
10. Development of autonomous car-part i: Distributed system architecture and development process / K. Jo [и др.] // IEEE Transactions on Industrial Electronics. — 2014. — Т. 61, № 12. — С. 7131—7140. — DOI: 10.1109/TIE.2014.2321342. — URL: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-84907478492&doi=10.1109%2fTIE.2014.2321342&partnerID=40&md5=0c8d2b4994d7dab581f8c5318fc29c7a> ; cited By 64.
11. Fagnant D. J., Kockelman K. M. The travel and environmental implications of shared autonomous vehicles, using agent-based model scenarios // Transportation Research Part C: Emerging Technologies. — 2014. — Т. 40. — С. 1—13. — ISSN 0968-090X. — DOI: <https://doi.org/10.1016/j.trc.2013.12.001>.
12. Ferguson D., Howard T. M., Likhachev M. Motion Planning in Urban Environments // The DARPA Urban Challenge. — 2009.
13. González-Sieira A., Mucientes M., Bugarín A. A State Lattice Approach for Motion Planning under Control and Sensor Uncertainty // Т. 253. — 11.2013. — DOI: 10.1007/978-3-319-03653-3\_19.

14. HDL-64E user's manual / Velodyne. — URL: <https://www.velodynelidar.com/lidar/products/manual/HDL-64E%20Manual.pdf>.
15. Howard T. M., Kelly A. Optimal Rough Terrain Trajectory Generation for Wheeled Mobile Robots // The International Journal of Robotics Research. — 2007. — Т. 26, № 2. — С. 141–166. — DOI: 10.1177/0278364906075328.
16. Junior: The Stanford Entry in the Urban Challenge / M. Montemerlo [и др.] // Journal of Field Robotics. — 2008. — Сент. — Т. 25. — С. 569–597. — DOI: 10.1002/rob.20258.
17. LaValle S. M., Kuffner J. J. Randomized kinodynamic planning // Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C). Т. 1. — 05.1999. — 473–479 vol.1. — DOI: 10.1109/ROBOT.1999.770022.
18. LaValle S. M., Kuffner J. J. Randomized kinodynamic planning // Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C). Т. 1. — 05.1999. — 473–479 vol.1. — DOI: 10.1109/ROBOT.1999.770022.
19. LaValle S. M. Rapidly-Exploring Random Trees: A New Tool for Path Planning //. — 1998.
20. M. Gloderer A. H. SplinesParametersTrajectoryExperimentalResultsSpline-basedTrajectoryOptimizationforAutonomousVehicleswithAckermannDrive. — 2010. — URL: <https://www.semanticscholar.org/paper/Spline-based-Trajectory-Optimization-for-Autonomous-Gloderer-Hertle/09489079e1fb794258ae3da691c2b368e43e3943>.
21. Model-predictive active steering and obstacle avoidance for autonomous ground vehicles / Y. Yoon [и др.] // Control Engineering Practice. — 2009. — Т. 17, № 7. — С. 741–750. — ISSN 0967-0661. — DOI: <https://doi.org/10.1016/j.conengprac.2008.12.001>. — URL: <http://www.sciencedirect.com/science/article/pii/S0967066108002025>.
22. Motion planning for urban driving using RRT / Y. Kuwata [и др.] // 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems. — 09.2008. — С. 1681–1686. — DOI: 10.1109/IR08.2008.4651075.
23. Motion planning in complex environments using closed-loop prediction / Y. Kuwata [и др.] // Proc. AIAA Guidance, Navigation, and Control Conf. and Exhibit. — 2008.
24. Nagendran N., Kolhe A. Security and safety with facial recognition feature for next generation automobiles // International Journal of Recent Technology and Engineering. — 2019. — Т. 7, № 4. — С. 289–294.
25. Obstacle avoidance of autonomous vehicles based on model predictive control / J.-M. Park [и др.] // Proceedings of The Institution of Mechanical Engineers Part D-Journal of Automobile Engineering - PROC INST MECH ENG D-J AUTO. — 2009. — Дек. — Т. 223. — С. 1499–1516. — DOI: 10.1243/09544070JAUTO1149.
26. Optimal Trajectory Generation for Dynamic Street Scenarios in a Frenet Frame / M. Werling [и др.] //. — 06.2010. — С. 987–993. — DOI: 10.1109/ROBOT.2010.5509799.
27. Pivtoraiko M., Kelly A. Efficient constrained path planning via search in state lattices // The 8th International Symposium on Artificial Intelligence, Robotics and Automation in Space. — 09.2005.
28. PROUD-Public Road Urban Driverless-Car Test / A. Broggi [и др.] // IEEE Transactions on Intelligent Transportation Systems. — 2015. — Т. 16, № 6. — С. 3508–3519. — DOI: 10.1109/TITS.2015.2477556. — URL: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-8495948888&doi=10.1109%2fTITS.2015.2477556&partnerID=40&md5=55565198a5552f02df98a9cdb0ef1f72> ; cited By 21.
29. Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions / C. Katrakazas [и др.] // Transportation Research Part C: Emerging Technologies. — 2015. — Ноябрь. — Т. 60. — С. 416–442. — DOI: 10.1016/j.trc.2015.09.011.

30. *Shiller Z., Gwo Y.-R.* Dynamic motion planning of autonomous vehicles // Robotics and Automation, IEEE Transactions on. — 1991. — Май. — Т. 7. — С. 241–249. — DOI: 10.1109/70.75906.
31. *Şucan I. A., Moll M., Kavraki L. E.* The Open Motion Planning Library // IEEE Robotics & Automation Magazine. — 2012. — Дек. — Т. 19, № 4. — С. 72–82. — DOI: 10.1109/MRA.2012.2205651. — <http://ompl.kavrakilab.org>.
32. Team AnnieWAY’s autonomous system for the 2007 DARPA Urban Challenge / S. Kammel [и др.] // Journal of Field Robotics. — 2008. — Т. 25, № 9. — С. 615–639. — DOI: 10.1002/rob.20252. — eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1002/rob.20252>.
33. Towards fully autonomous driving: Systems and algorithms / J. Levinson [и др.] // 2011 IEEE Intelligent Vehicles Symposium (IV). — 06.2011. — С. 163–168. — DOI: 10.1109/IVS.2011.5940562.
34. Yandex Self-Driving Car project / Yandex. — URL: <https://yandex.ru/promo/taxi/sdc>.
35. *Ziegler J., Werling M., Schröder J.* Navigating car-like robots in unstructured environments using an obstacle sensitive cost function //. — 2008. — С. 787–791. — DOI: 10.1109/IVS.2008.4621302. — URL: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-57749196757&doi=10.1109%2fIVS.2008.4621302&partnerID=40&md5=11accfcd3da5016fd0cbe4e4919fd9b7> ; cited By 30.
36. *А.С. Г. Фрундрование машинки.*
37. *В. Л. МЕТОДЫ ПЛАНИРОВАНИЯ ПУТИ В СРЕДЕ С ПРЕПЯТСТВИЯМИ (ОБЗОР) // МАТЕМАТИКА И МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ.* — 2018. — № 1. — С. 15–58. — DOI: 10.24108/mathm.0118.0000098. — URL: <https://www.mathmelpub.ru/jour/article/view/98#>.
38. *Казаков К.А. С. В.* Обзор современных методов планирования движения // Труды Института системного программирования РАН. — 2016.