АННОТАЦИЯ

ABSTRACT

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В настоящее время активно ведутся исследования и разработки в области беспилотных транспортных средств, в частности, беспилотных автомобилей. Бесплотные 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 [**company_1**], Яндекс [**company_yandex**], 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 [**overview_private_ovnership**], 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 управление 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 использования 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 управление 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 Chellenge (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 ялвяется 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 [**sensor_hdl64_manual**]. 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) [**darpa_junior**], Talos (Массачусетский технологический институт)[**darpa_mit**], BOSS (несколько участников из университета Карнеги-Меллона, компаний General Motors, Caterpillar, Continental, Intel) [**darpa_boss**], AnnieWAY (технологический институт Карлсруэ) [**darpa_annieway**] 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 радарами.

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 объекты (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 мне 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 радов 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 [**sensor_radar_overview**], [**sensor_radar_overview_2**]. 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 [**darpa_boss**] в Darpa Urban Challenge применяла радар совместно с ЛИДАРом 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 **??** приведено сравнение основных свойств и областей применения рассмотренных выше датчиков.

Таблица 1 – Сравнение применяемых датчиков

Датчик	Получаемая информации	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, дорогу, 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 [**darpa_junior**], [**darpa_mit**], [**darpa_boss**], [**darpa_annieway**], так и описана в более современных статьях, таких как [**developing_car_1**], [**car_proud**], [**car_race**].

Архитектура обобщенной системы управления автономным автомобилем, построенная на основе анализа ряда работ [**motion_planning_review**], представлена на рисунке **??**.

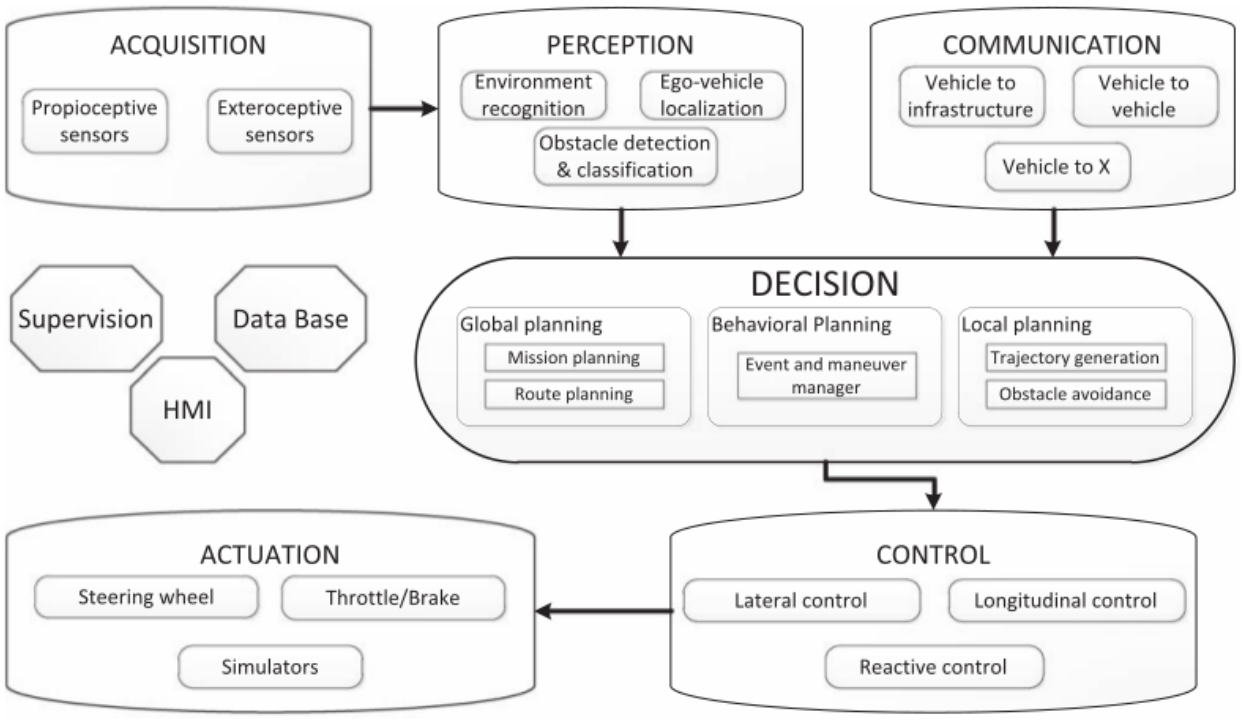

Рисунок 1 – Обобщенная абстракция архитектуры управления автономными автомобилями [**motion_planning_review**]

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 [**motion_planning_survey**]. 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 **??**.

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 маршрутка может быть сформулирована как задача нахождения пути на графе с наименьшей стоимостью. Участники Darpa Urban Challenge решали эту задачу самостоятельно, использую 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 правильно поведения 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 остальные участников дорожного движения. Проблема предсказания намерений участников движения активно изучается и существуют различные подходы, от простой линейно экстраполяции [**darpa_junior_frenet**] 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 спомощью 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 све-

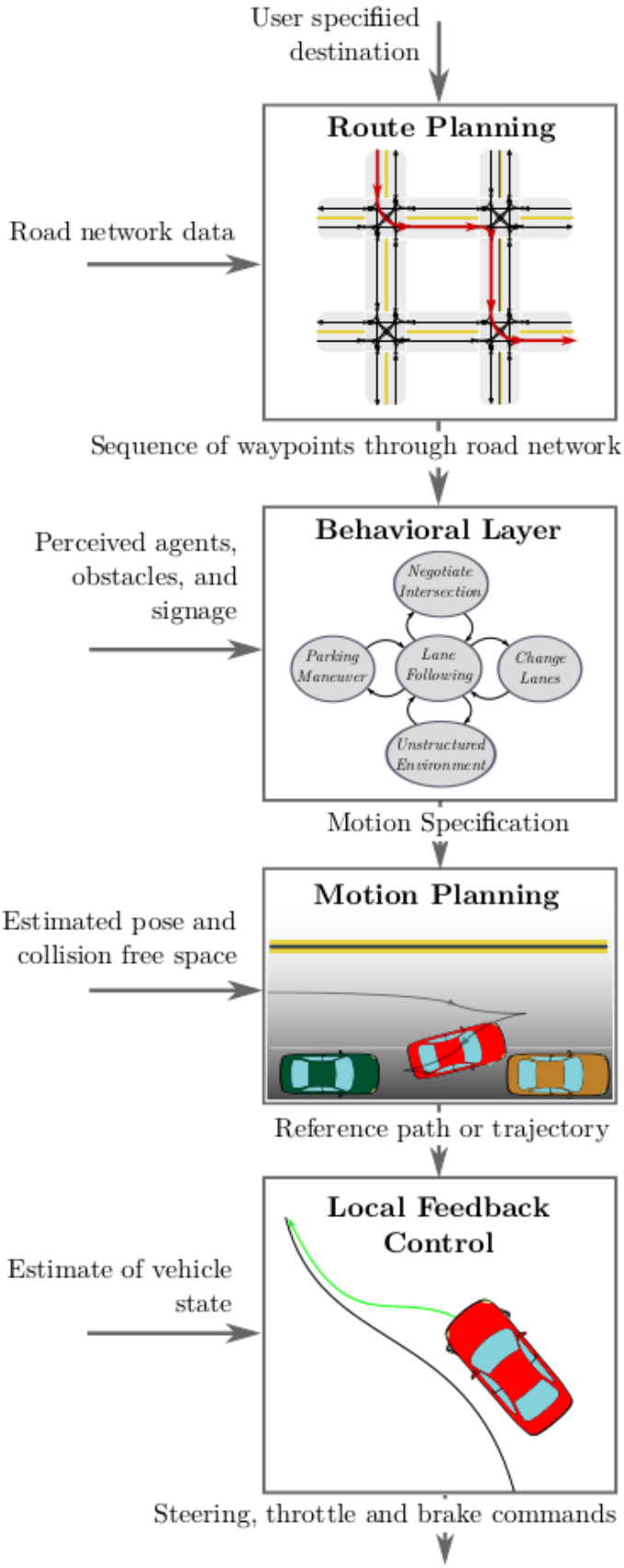

Рисунок 2 – Иерархия системы управления

тофоре, 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 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 положение 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**], [**motion_planning_review**], [**motion_planning_overview_obstacles**], [**motion_planning_overview_modern**], [**motion_planning_review_2**].

Различные способы планирования движения имеют свои преимущества и недостатки. Например, одни методы позволяют более поло 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, неголономные связи [**car_frund**]. Многие исследователи применяют различные упрощенные модели динамики автомобиля [**car_dynamics_1**]. Очень распространенной является "велосипедная"модель динамики автомобиля, которая применяется в ряде работ [**car_dynamics_bycicle_model_1**], [**car_dynamics_bycicle_model_2**], [**darpa_annieway_navigation**].

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 $\mathcal{C}$ системы — это совокупность всех ее обобщенных координат.

Дано:

- пространство конфигураций $\mathcal{C}$ робота;
- множество конфигураций, соответствующее препятствиям $\mathcal{C}_o \in \mathcal{C}$, соответственно множество свободных конфигураций $\mathcal{C}_{free} = C \setminus C_o$;
- 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 $\tau(\alpha) : [0, 1] \to \mathcal{C}_{free}$, 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 [**darpa_boss**], или делегировать эту задачу нижележащим подсистемам.

В случае планирования траектории, явно учитывается время движение. В таком случае требуется найти функцию $\pi(t) : [0, T] \to C$, 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) [**motion_planning_overview_obstacles**]. 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. [**motion_planning_overview_obstacles**] 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 построение графа, который сильно распространен в задачах планирования [**darpa_annieway_navigation**], [**motion_planning_overview_obstacles**], 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 [**darpa_boss**], [**darpa_boss_2**], [**darpa_boss_3**]. 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.

1.3.3 State Lattice

Отдельного упоминания заслуживает разновидность поиска на графах, которая получила название в англоязычной литературе "State Lattice". 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* [**motion_planning_hybrid_a_star**], предложенный командой "Junior"DUC [**darpa_junior**]. 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 **??**).

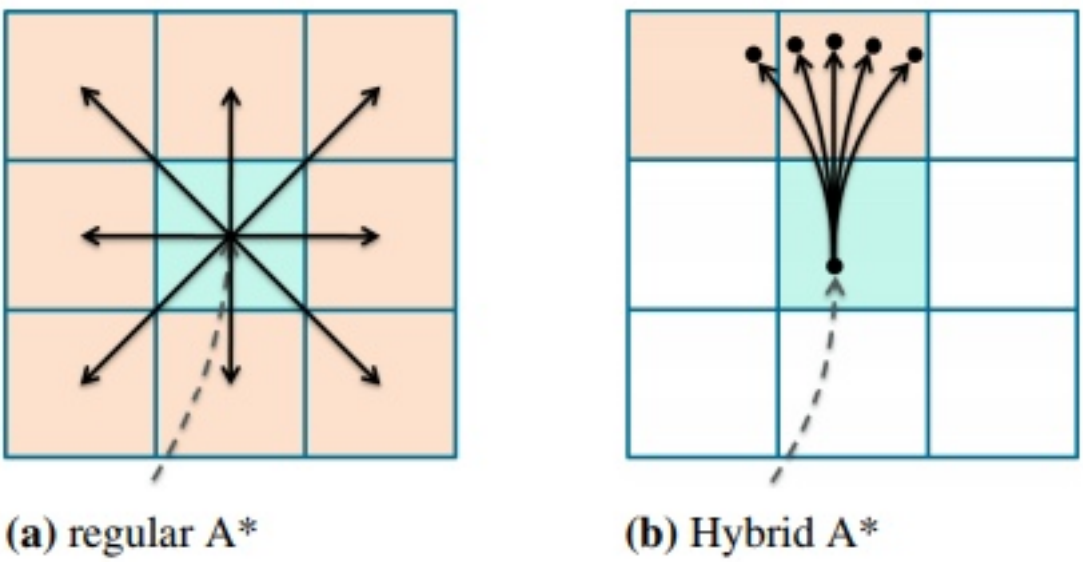

Рисунок 3 – Hybrid A* в сравнении с A* [**motion_planning_hybrid_a_star**]

1.3.4 Случайные методы

При решении задач планирования движения в пространствах высоких размерностей, которые часто возникают в современной робототехнике (например, планирование движение 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) [**motion_planning_rrt**], Optimal Rapidly-exploring random tree (RRT*)[**motion_planning_rrt_star**].

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 **??**, а иллюстрация метода $EXTEND$ на рисунке **??**.

Algorithm 1 Rapidly-exploring random tree

function RRT(x_{init})
  $\tau.init(x_{init})$
  for $k = 1$ to K **do**
    $x_{rand} \leftarrow RANDOM\_STATE()$
    EXTEND(τ, x_{rand})
  end for
end function
function EXTEND(τ, 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 ifreturn TRAPPED
end function

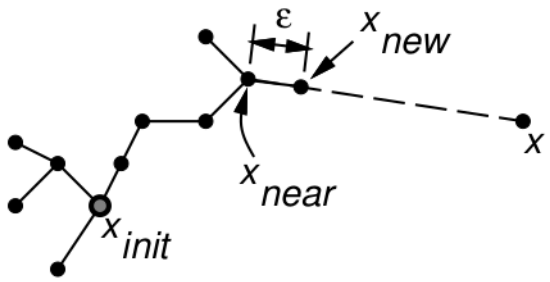

Рисунок 4 – Пример работы метода EXTEND алгоритма RRT

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 планирования отводится 0.5 секунды, и превышение этого времени может привести к нарушению функ-

ционирования 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* [**motion_planning_rrt_star**], являющийся асимптотически-оптимальной версией алгоритма RRT. Он основан на 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) [**motion_planning_kinodynamic_rrt**], 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 модули 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 $PROPAGATE$ (**??**) формирует следующее состояние системы после применения к ней заданного управляющего воздействия, например, путем численного интегрирования.

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

где x_i — текущее состояние,
u_i — управление,
Δt — время интегрирования.

Функция $STEER$ (**??**) значительно сложнее. Согласно алгоритму RRT необходимо осуществлять шаг из некого 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 ите-

раций, чтобы прийти к решению.

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

где x — случайно выбранное состояние,
x_{newar} — состояние, близкое к случайному,
u^* — примерное управление,
Δt — время интегрирования.

Интересное решение этой проблемы было предложено командой MIT Darpa Urban Challenge [**darpa_mit**], [**darpa_mit_2**], [**darpa_mit_3**], [**darpa_mit_3**]. 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 сэмплирования 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 **??**). Этот подход позволяет уменьшить количество вершин дерева, которое приходится открывать алгоритму RRT, тем самым уменьшив вычислительную сложность.

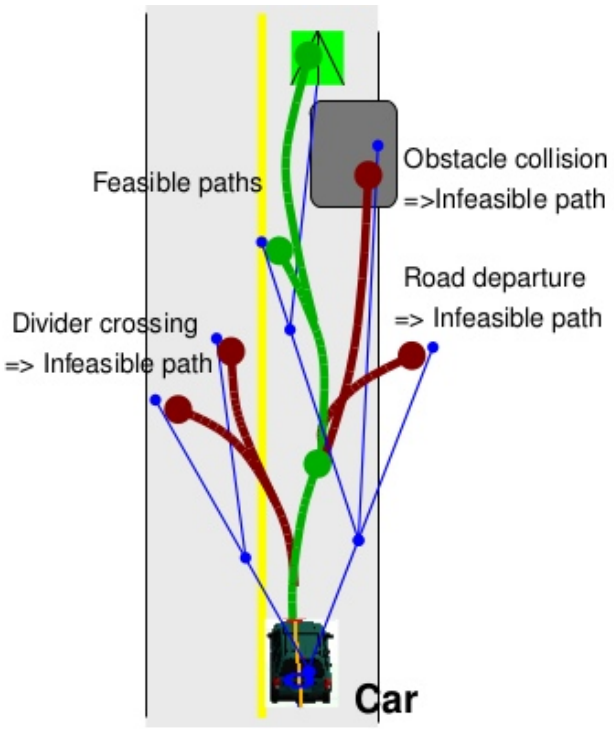

Рисунок 5 – Планирование движения с помощью RRT и регулятора с обратной связью

Существует Open-Source библиотека The Open Motion Planning Library [**motion_planning_ompl**], 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 вышеназванные 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

1.3.5 Методы интерполяции кривыми

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

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 [**darpa_junior_path_planning**], [**darpa_junior_frenet**], опубликованных по результатам участия команды Стэндфордского университета в Darpa Ubran 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 **??**, 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.

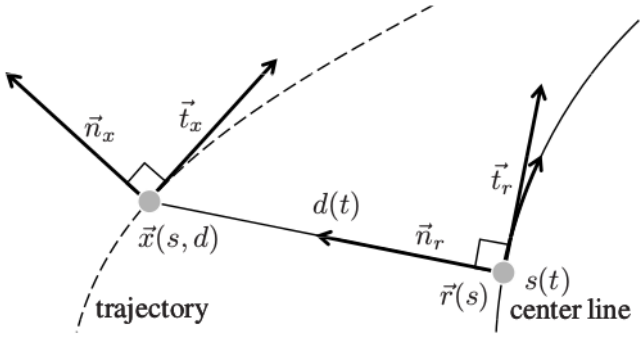

Рисунок 6 – Система координат для планирование траектории

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):

$$J = \int_{t_0}^{t_1} \dddot{p}^2 d\tau \tag{3}$$

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:

$$D_0 = [d_0, \dot{d}_0, \ddot{d}_1] \tag{4}$$

$$D_{1_{ij}} = [d_i, 0, 0, T_j] \tag{5}$$

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 траектории представлен на рисунке **??**.

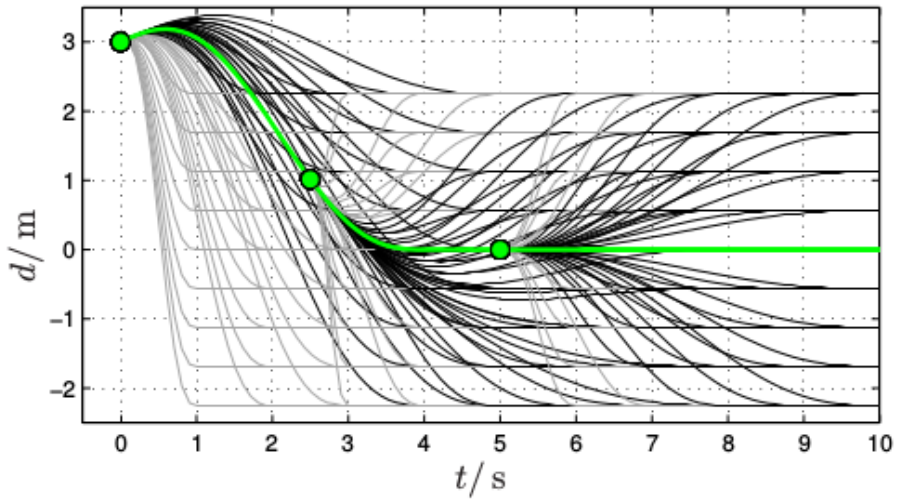

Рисунок 7 – Планирование поперечного движения, зеленая — оптимальная траектория, черная — валидные траектории, серые — невалидные траектории

В работе [**motion_planning_path_optimization**] 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.

1.3.6 Методы численной оптимизации

1.4 Обзор некоторых систем управления беспилотными автомобилями

1.4.1 Команда Junior в Darpa Urban Challenge

1.4.2 Команда Talos в Darpa Urban Challenge

1.4.3 Команда AnnieWAY в Darpa Urban Challenge

1.4.4 Команда BOSS 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