МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

МЕХАНИКО-МАТЕМАТИЧЕСКИЙ ФАКУЛЬТЕТ

Кафедра дифференциальных уравнений и системного анализа

КЛИМЕНКО

Кирилл Владимирович

НЕЙРОЭВОЛЮЦИОННЫЕ АЛГОРИТМЫ И ГЛУБОКОЕ ОБУЧЕНИЕ БЕСПИЛОТНЫХ МОДЕЛЕЙ АВТОМОБИЛЕЙ В ГЕНЕРИРУЕМЫХ 2D- И 3D-СИМУЛЯЦИЯХ ДОРОЖНОГО ОКРУЖЕНИЯ

Дипломная работа

Научный руководитель:
кандидат физ.-мат. наук,
доцент А. Э. Малевич

Допущена к защите

«___» _____ 2022 г.

Зав. кафедрой дифференциальных уравнений и системного анализа

канд. физ.-мат. наук, доцент Л. Л. Голубева

Минск, 2022

ОГЛАВЛЕНИЕ

РЕФЕРАТ

В дипломной работе 80 страниц, 36 рисунков, 19 источников, 1 электронное приложение.

НЕЙРОННЫЕ СЕТИ, КОМПЬЮТЕРНАЯ МАТЕМАТИКА, 2D- И 3D-МОДЕЛИРОВАНИЕ, БЕСПИЛОТНЫЕ СИСТЕМЫ, КОМПЬЮТЕРНАЯ СИМУЛЯЦИЯ, МАШИННОЕ ОБУЧЕНИЕ

Объектом исследования являются беспилотные автомобильные системы, методы их проектирования, обучения и моделирования.

Целью дипломной работы является создание эффективных и реалистичных моделей беспилотных автомобилей в симуляциях дорожного окружения и парковочной зоны.

Для достижения цели использовались следующие инструменты: Udacity Driving Simulator, язык Python, библиотеки pygame, numpy, neat, scipy, keras.

В дипломной работе достигнуты следующие результаты:

1) Подробно описаны принципы работы беспилотных автомобилей.
2) Сравнены различные технологии, используемые в обучении и проектировании беспилотных автомобилей.
3) Построены реалистичные виртуальные модели беспилотных автомобилей и симуляции дорожного окружения.

Новизна результатов состоит в уникальной кинематической модели виртуальных беспилотных автомобилей. Результаты дипломной работы могут быть использованы для проектирования и испытания беспилотных автомобильных систем.

Дипломная работа является завершённой, поставленные задачи выполнены в полной мере, присутствует возможность дальнейшего развития исследований. Дипломная работа выполнена автором самостоятельно.

РЭФЕРАТ

У дыпломнай працы 80 старонак, 36 малюнкаў, 19 крыніц, 1 электроннае прыкладанне.

НЕЙРОННЫЯ СЕТКІ, КАМП'ЮТАРНАЯ МАТЭМАТЫКА, 2D- І 3D-МАДЭЛЯВАННЕ, БЕСПІЛОТНЫЯ СІСТЭМЫ, КАМП'ЮТАРНАЯ СІМУЛЯЦЫЯ, МАШЫННАЕ НАВУЧАННЕ

Аб'ектам даследавання з'яўляюцца беспілотныя аўтамабільныя сістэмы, метады іх праектавання, навучання і мадэлявання.

Мэтай дыпломнай працы з'яўляецца стварэнне эфектыўных і рэалістычных мадэлей беспілотных аўтамабіляў у сімуляцыях дарожнага асяроддзя і парковачнай зоны.

Для дасягнення мэты выкарыстоўваліся прылады: Udacity Driving Simulator, мова Python, бібліятэкі pygame, numpy, neat, scipy, keras.

У дыпломнай рабоце дасягнуты наступныя вынікі:

1) Падрабязна апісаны прынцыпы працы беспілотных аўтамабіляў.
2) Параўнаны розныя тэхналогіі, якія выкарыстоўваюцца ў навучанні і праектаванні беспілотных аўтамабіляў.
3) Пабудаваны рэалістычныя віртуальныя мадэлі беспілотных аўтамабіляў і сімуляцыі дарожнага асяроддзя.

Навізна вынікаў складаецца ва ўнікальнай кінематычнай мадэлі віртуальных беспілотнікаў.

Вынікі працы могуць быць скарыстаны для праектавання і выпрабавання беспілотных аўтамабільных сістэм.

Дыпломная работа з'яўляецца завершанай, пастаўленыя задачы выкананы ў поўнай меры, прысутнічае магчымасць далейшага развіцця даследаванняў.

Дыпломная работа выканана аўтарам самастойна.

ABSTRACT

Thesis project is presented in the form of an explanatory note of 80 pages, 36 figures, 19 references, 1 electronic application.

NEURAL NETWORKS, COMPUTER MATHEMATICS, 2D- AND 3D-MODELING, UNMANNED SYSTEMS, COMPUTER SIMULATION, MACHINE LEARNING

The research object of this thesis project is unmanned vehicle systems, methods of their design, training and modeling.

The purpose of this work is to create efficient and realistic models of unmanned vehicles in simulations of the road environment and parking area.

The following tools were used to achieve the goal: Udacity Driving Simulator, Python language, pygame, numpy, neat, scipy, keras libraries.

The main results of the thesis project are as follows:

1) The main principles of unmanned vehicles were described in detail.
2) Different technologies used in training and designing unmanned vehicles were compared.
3) Realistic virtual models of unmanned vehicles and simulations of the road environment were built.

Novelty of the work lies in the unique kinematic model of virtual unmanned vehicles.

The results of the thesis can be used to design and test unmanned vehicle systems.

The thesis project is complete, all tasks have been successfully done, there is a possibility for further research and development.

The thesis work was done by the author independently.

ВВЕДЕНИЕ

Благодаря достижениям в области глубокого обучения и искусственного интеллекта, последнее десятилетие стало свидетелем широкой популярности и всё более быстрого прогресса в разработке и даже использовании технологий беспилотных транспортных средств. В настоящее время происходит активное развитие автоматизации всех видов устройств и гаджетов, способных каким-либо образом передвигаться в любых земных и космических средах: начиная от примитивной робототехники и заканчивая высокотехнологичными транспортными средствами передвижения. Так, например, в военной, космической и авиационной промышленности начали появляться беспилотные летательные дроны, в повседневной жизни начали встречаться беспилотные грузоперевозчики и доставщики, а современную логистику теперь почти невозможно представить без автономных транспортных систем, упрощающих внутрипроизводственное управление различными процессами по всему миру.

Беспилотное вождение, безусловно, изменило способы наших путешествий и исследований разных уголков земного шара и космического пространства, значительно расширив их границы. Нарастающая популярность и успешность беспилотных технологий, а также увеличение соответствующего спроса на мировом рынке способствуют их скорейшему совершенствованию и последующему внедрению в различные сферы человеческой деятельности. В частности, наиболее активно этот прогресс наблюдается в автомобильной промышленности и робототехнике. Говоря конкретно об автомобилях — ручное управление не обеспечивает полную безопасность и не гарантирует избегания смертельных исходов во всех возможных условиях вождения. Одним из потенциальных решений этой проблемы является частичное или полное внедрение технологий автономного вождения в транспортные средства.

В рамках данной работы будет представлен обзор современного состояния некоторых технологий глубокого обучения и нереволюционных алгоритмов,

используемых в автономном вождении. Будут рассмотрены методологии, формирующие основы исследования сцен вождения, алгоритмы восприятия дорожного окружения, планирования пути, арбитража поведения и управления движением, а также архитектуры беспилотных моделей на основе искусственного интеллекта, свёрточные и рекуррентные нейронные сети и парадигмы глубокого обучения с подкреплением. В качестве практической демонстрации технологий беспилотных транспортных средств на базе рассмотренных методологий и алгоритмов будут реализованы основные концепты вождения и парковки кинематических моделей автомобилей при базовых условиях дорожного окружения и базовых правилах дорожного движения в генерируемых 2D- и 3D-симуляциях.

Полная документация и реализация проекта будет доступна в публичном GitHub-репозитории [1] под открытым лицензированием GNU GPL v3.0.

ГЛАВА 1
КОНЦЕПТ БЕСПИЛОТНЫХ АВТОМОБИЛЕЙ

1.1 Зарождение концепта беспилотных автомобилей

История автомобиля восходит к 1885 году, когда был выпущен первый серийный автомобиль, разработанный Карлом Бенцем в Мангейме, Германия. Он назывался «Benz Patent-Motorwagen» (см. Рис. 1.1) и использовал бензиновый двигатель внутреннего сгорания. Уже после появился первый Ford Model T от Ford Motor Company.

Рисунок 1.1 Benz Patent-Motorwagen

В то время было много энтузиазма по поводу частично или полностью автономных автомобилей. Для эффективной навигации и передвижения такого автомобиля технологии из нескольких дисциплин должны быть объединены воедино. Эти дисциплины широко включают информатику, электротехнику и машиностроение.

Так называемое «Линрикканское чудо» 1920-х годов был первым радиоуправляемым автомобилем. В 1939 году впервые были

продемонстрированы электромобили, которые питались от встроенных схем. А уже в 1980 году компания Mercedes-Benz представила роботизированный фургон, в котором использовались первые модели систем компьютерного зрения.

Это стало отправной точкой для зарождения беспилотных технологий, используемых в настоящее время в современных автомобилях. Такие технологии включают в себя помощь в движении по дорожной полосе, предупреждение о движении вне дорожной полосы, адаптивный круиз-контроль и т. д.

Согласно февральскому отчёту Всемирной организации здравоохранения о дорожно-транспортном травматизме, за 2021 год было зафиксировано примерно 1.35 миллиона смертей в результате дорожно-транспортных происшествий. Большинство этих несчастных случаев списывают на счёт человеческих ошибок, которые могут быть вызваны превышением скорости, вождением в нетрезвом виде, психическим состоянием водителя или отвлекающими факторами во время вождения, например, при использовании мобильных телефонов. Другие ошибки включают неиспользование ремней безопасности, шлемов или другого защитного снаряжения [2].

Приведённые выше статистические данные свидетельствуют о необходимости широкого внедрения и продвижения технологий автономных транспортных средств. Без этого количество транспортных травм и смертей, как ожидается, будет только увеличиваться.

Внедрение автономных технологий имеет множество преимуществ, таких как резкое сокращение количества столкновений, более высокая прочность и надёжность, улучшение трафика и уменьшение заторов на дорогах.

Согласно отчёту фонда RAC [3], среднестатистическая машина:

- припаркована дома или в гараже 80% времени;
- припаркована в другом месте 16% времени;
- находится на дороге только 4% времени.

Это подразумевает, что в среднем автомобиль проводит примерно 96% времени на стоянке. Таким образом, появление автономных автомобилей значительно уменьшило бы потребность в частной собственности на автомобили: благодаря совместному использованию или аренде, в массовое потребление может быть внедрена модель автономных роботов-такси. Такая модель каршеринга будет реализована Cruise в своем автономном транспортном средстве Origin (см. Рис. 1.2), которое вообще не будет доступно в качестве частной собственности.

Рисунок 1.2 Автомобиль Cruise Origin

Автономные транспортные средства также могут быть полезны в сфере доставки, которую можно сделать более надёжной и эффективной за счет их использования.

1.2 Системы беспилотных технологий транспортных средств

Беспилотные автомобили — это автономные системы принятия решений, самостоятельно (без какого-либо контроля со стороны человека) воспринимающие и понимающие окружающую обстановку посредством обработки алгоритмами машинного обучения потоков наблюдений, поступающих от различных бортовых источников и приводов, собирающих в режиме реального времени данные об окружающей среде, включая

географические координаты, скорость и направление движения автомобиля, его ускорение и препятствия, с которыми может столкнуться автомобиль. Эти наблюдения используются бортовым компьютером автомобиля для принятия решений при вождении.

Вместе с полностью автоматизированными транспортными средствами существуют и частично автоматизированные автомобили, технологии которых выступают в качестве помощи водителю на дороге в различных ситуациях. Различают шесть уровней технологий автоматизации вождения:

- «Уровень 0: без автоматизации» — все задачи выполняет водитель;
- «Уровень 1: помощь водителю» — присутствуют автономные компоненты автомобиля, такие как электронный ассистент стабилизации или автоматическое торможение;
- «Уровень 2: частичная автоматизация» — присутствуют комбинированные автоматизированные функции, такие как рулевое управление/ускорение, т. е. удержание полосы движения и адаптивный круиз-контроль (тем не менее, водитель всегда должен быть вовлечен в езду и следить за дорогой);
- «Уровень 3: условная автоматизация» — водитель может полностью прекратить управление некоторыми важными функциями транспортного средства в определённых условиях, но должен оставаться готовым взять на себя управление транспортным средством в любое время по предварительному уведомлению;
- «Уровень 4: высокая автоматизация» — автомобиль может выполнять все функции вождения, а возможность передачи управления транспортным средством водителю опциональна;
- «Уровень 5: полная автоматизация» — автомобиль способен выполнять все функции, связанные с вождением, в любых ситуациях и условиях.

Очевидно, что беспилотное транспортное средство представляет собой не одну технологию, а сложную систему технологий, состоящую из множества подсистем. Условно их можно разделить на три основные группы (см. Рис. 1.3):

- алгоритмы сенсорного сканирования, восприятия и принятия решений;
- операционная система и аппаратно-вычислительная платформа;
- облачные платформы для машинного обучения, моделирования условий вождения и хранения данных.

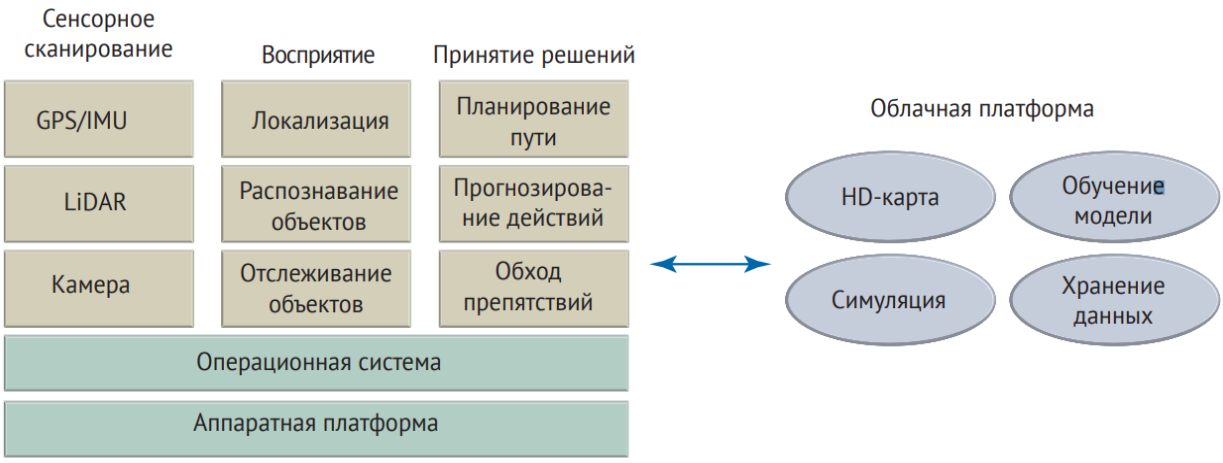

Рисунок 1.3 Архитектура беспилотного транспортного средства [4]

Первая подсистема (подсистема сканирования, восприятия и принятия решений) отвечает за извлечение необходимой информации из необработанных данных, полученных сенсорами, и обеспечивает анализ окружающей среды, на основе которого в дальнейшем строится принятие решений относительно будущих действий беспилотного транспортного средства.

Вторая подсистема (клиентская подсистема) интегрирует алгоритмы первой подсистемы в режиме реального времени и предоставляет слой контроля вычислительных коммуникаций и взаимодействия с конечными пользователем.

Третья подсистема (платформа облачных вычислений) обеспечивает автономные вычисления, хранение данных и совершенствование (тестирование и обучение) алгоритмов и моделей машинного обучения.

1.3 Последние разработки отрасли беспилотных автомобилей

Грандиозная популярность и успешность беспилотных технологий способствуют их скорейшему совершенствованию и прогрессирующему внедрению в различные транспортные средства. Последние разработки в отрасли автоматизированных транспортных средств уже исчисляются тысячами и включают в себя: уведомления об авариях, заблаговременные предупреждения о пробках, строительных работах на дорогах, превышении скорости, сигналах светофора, предупреждения о тумане, наличии гололедицы, предложение определенных услуг в зависимости от местоположения и т. д.

Эти и другие инновационные разработки, наряду с разработками в сфере автомобилестроения, привели к тому, что такие корпорации, как Google, и производители автомобилей, такие как Tesla и Audi, начали развивать, совершенствовать и внедрять беспилотные автомобильные технологии в свои продукты. Помимо этих производителей, другие ведущие автоконцерны, такие как Ford, BMW, Kia, Hyundai, Honda, Toyota, Mercedes-Benz, General Motors, Volvo, Nissan и Volkswagen, уже внедрили в свои автомобили: экстренное торможение, интеллектуальную парковку, предупреждения об авариях и полуавтоматическое вождение.

Сегодня партнерские отношения между производителями автомобилей и технологическими компаниями позволяют проектировать и разрабатывать ещё более инновационные беспилотные технологии. Например, Microsoft сотрудничает с Toyota и Volvo для разработки полностью автономных автомобилей. Помимо Microsoft, в проекты с беспилотными автомобилями активно вовлечены Apple и Uber. В Европе такие производители, как Mercedes-Benz и BMW, лидируют в разработке концепции беспилотных прототипов транспортных средств.

1.3.1 Tesla

Tesla — компания, занимающаяся производством электромобилей и решений для хранения электрической энергии, основанная в 2003 году. На данный момент компания выпускает исключительно полубеспилотники, требующие внимания водителя при включённом автопилоте, однако начиная с 2020 года началась активная стадия тестирования крупномасштабного проекта с автопилотом пятого уровня.

Согласно текущему описанию Tesla Autopilot [5], технология позволяет автомобилю автоматически управлять, разгоняться и тормозить в пределах своей полосы движения. Текущие функции автопилота требуют контроля со стороны водителя и пока что не делают автомобиль полностью беспилотным. Движение на автопилоте предлагает смену полосы движения для оптимизации маршрута и вносит коррективы в поездку, чтобы транспортное средство не застряло за медленными легковыми или грузовыми автомобилями. Когда функция активна, автопилот также автоматически направляет автомобиль к развязкам и съездам с учётом пункта назначения. На рисунке (1.4) показаны датчики и сенсоры, используемые Tesla Autopilot, и их расположение относительно автомобиля.

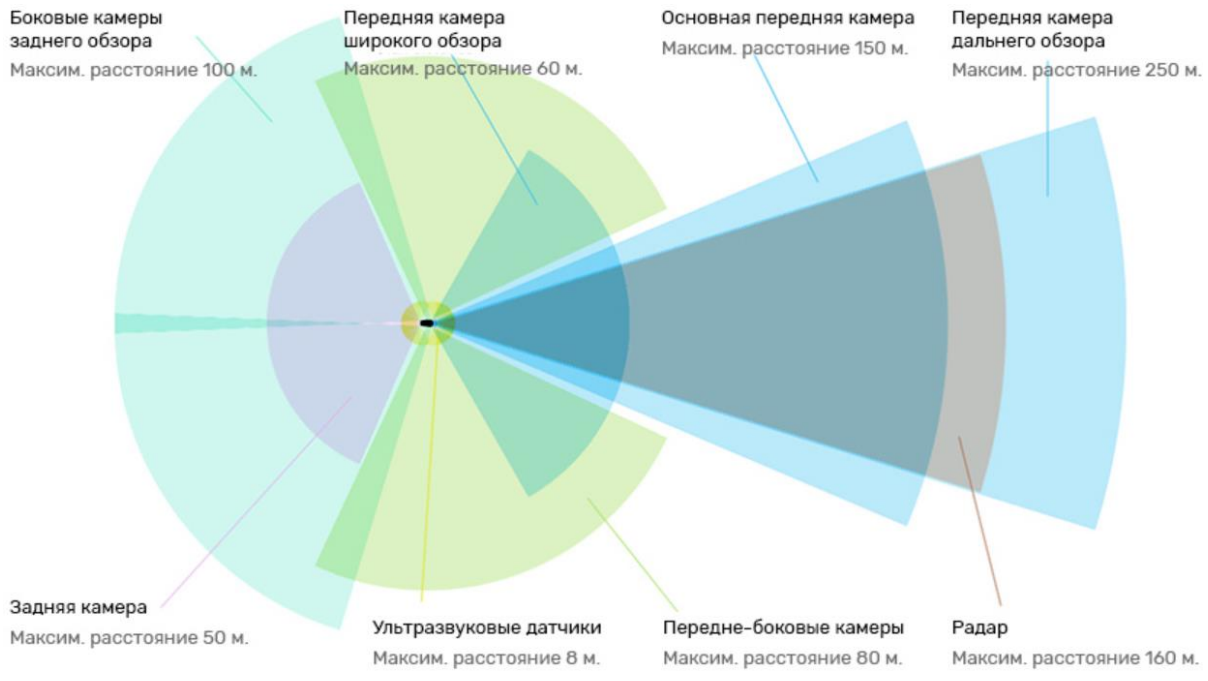

Рисунок 1.4 Схема расположения датчиков Tesla Autopilot

Передовые функции безопасности и удобства Tesla Autopilot разработаны для того, чтобы улучшить существующую функциональность автомобиля, сделать его более функциональным и безопасным, а также помочь водителю справиться с самыми монотонными моментами за рулём: езда по шоссе, в пробке, при парковке и т. д.

1.3.2 Waymo

Waymo — американская компания, занимающаяся технологиями беспилотных автомобилей, основанная в 2009 году и являющаяся дочерней компанией Alphabet Inc. В настоящее время Waymo считается лидером в разработке беспилотных автомобилей, поскольку её автономные автомобили проехали 20 миллионов миль по дорогам и наиболее приближены к пятому уровню автоматизации. Согласно отчету о безопасности Waymo [6], полностью разработанная система автономного вождения Waymo предназначена для работы без участия водителя, в отличие от технологий, продаваемых сегодня в автомобилях, таких как адаптивный круиз-контроль или системы удержания полосы дорожного движения, которые требуют постоянного контроля со стороны водителя.

Технология беспилотных автомобилей Waymo подпадает под систему автоматизированного вождения четвёртого уровня SAE International, поскольку эта технология также может полностью остановить транспортное средство в случае каких-либо сбоев системы.

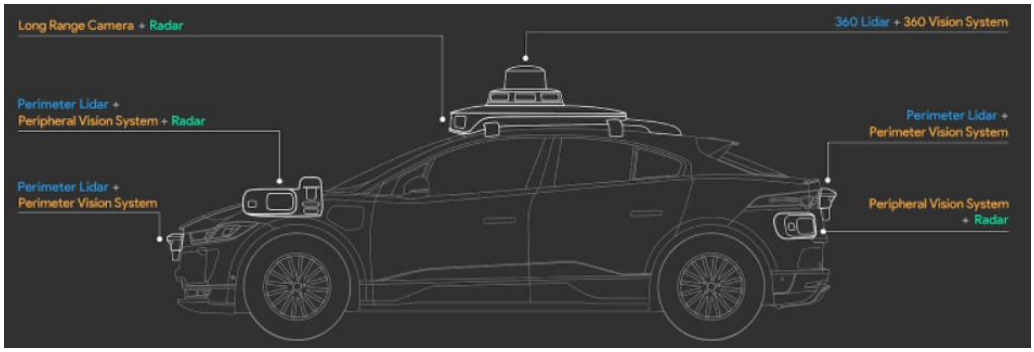

Рисунок 1.5 Схема сенсорных систем Waymo Jaguar I

Используемые датчики (см. Рис. 1.5) позволяют автомобилю видеть на 360 градусов как днём, так и ночью. Набор из нескольких сенсоров помогает создать трёхмерную картину окружения и показывает статические и динамические объекты. К таким объектам относятся транспортные средства, строительное оборудование, предупреждающие конусы, светофоры и пешеходы. Системы LIDAR (Light Detection and Ranging) среднего и дальнего радиусов действия, установленные в блоке на крыше, излучают миллионы лазерных импульсов в секунду на 360 градусов и измеряют время возврата отражения. В системе технического зрения с высоким разрешением используются камеры, которые видят мир в контексте — в поле зрения 360 градусов. Радиолокационная система использует длину волны для отслеживания объектов и движения. Она эффективна при любом освещении и любых погодных условиях, таких как снег, туман и дождь.

В дополнение к вышеупомянутым системам автомобили Waymo также используют датчики, такие как GPS, и систему, которая обнаруживает звук от транспортных средств службы экстренной помощи, которые находятся далеко, порядка сотен километров. Восприятие, прогнозирование поведения и планировщик — три основных компонента программного обеспечения Waymo. В 2019 году Waymo заключила партнерское соглашение с Jaguar Land Rover для создания до 20000 модифицированных полностью электрических автомобилей для использования в качестве роботов-такси [7]. Каждый автомобиль имеет 29 камер, расположенных спереди и сзади, по бокам и сверху автомобиля.

1.3.3 Cruise

Cruise — американская компания по производству беспилотных автомобилей и разработке программного обеспечения для Chevrolet Bolt, основанная в 2013 году и являющаяся дочерней компанией General Motors. Программное обеспечение Chevrolet Bolt используется, чтобы сделать автомобиль максимально автономным.

В январе 2020 года Cruise представила Origin — полностью автономный автомобиль без руля и педалей. Это позволяет электромобилю сразу переходить на пятый уровень автоматизации вождения. Автомобиль имеет симметричную конструкцию с двумя рядами сидений, обращенными друг к другу. Он предназначен для движения по шоссе. Автомобиль модульный и имеет срок службы миллион миль — в шесть раз больше, чем среднестатистический автомобиль. Цифровые дисплеи над головой предоставляют информацию о путешественнике и маршруте. Транспортное средство оснащено так называемой «совой», представляющей собой универсальный гибридный датчик, сочетающий в себе камеру и радар (см. Рис. 1.6) [8].

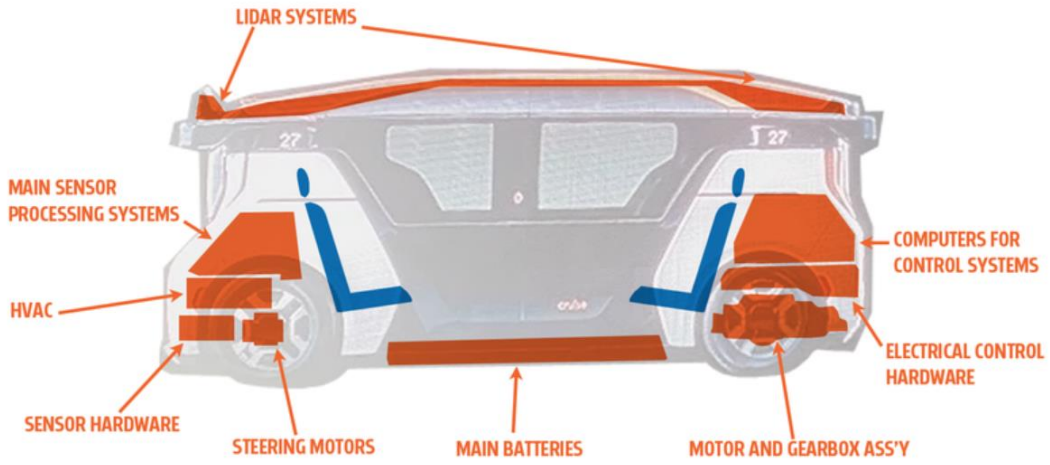

Рисунок 1.6 Схема расположения сенсоров Cruise Origin

1.3.4 Argo AI

Argo AI — компания, разрабатывающая полностью интегрированную систему автономного вождения, предназначенную для совместного использования и доставки товаров. С этой целью Argo AI работает с ведущими автопроизводителями, такими как Ford и Volkswagen. В настоящее время компания разрабатывает технологию строго по четвёртому уровню автоматизации. Компания сосредоточена на разработке полного продукта, включающего программную платформу и все датчики, в том числе LIDAR (Light Detection and Ranging), детекторы света, радар и камеры (см. Рис. 1.7).

Компания также использует платформу автономных транспортных средств, которая представляет собой набор традиционных инструментов, адаптированных для управления транспортным средством с помощью команд, генерируемых компьютером [9].

Чтобы решить две основные технические проблемы — восприятие и принятие решений — Argo AI следует подходу использования алгоритмов машинного обучения для решения проблем с помощью глубоких нейронных сетей и нереволюционных алгоритмов.

Для испытаний компания использует автомобили класса Ford Fusion Hybrid (см. Рис. 1.7). Уже в некоторых городских районах Argo AI разрабатывает и тестирует средства доставки товаров.

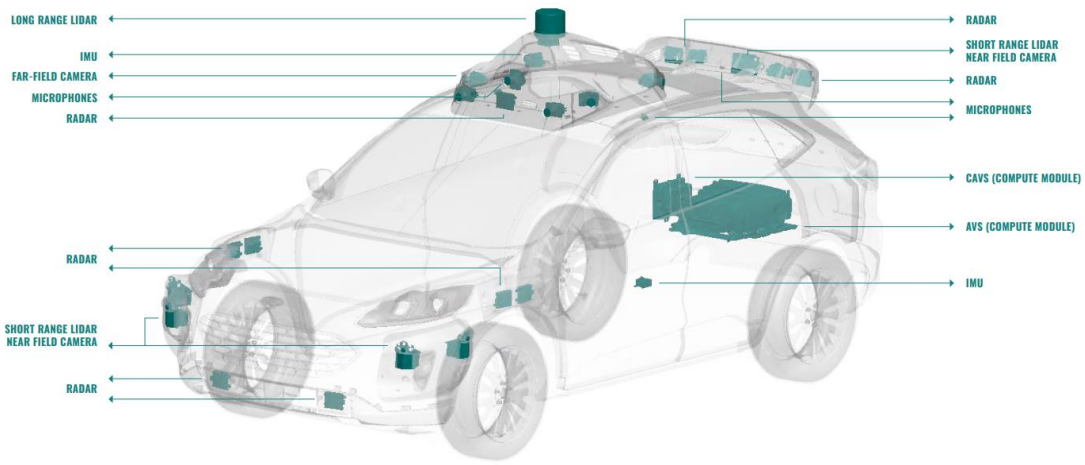

Рисунок 1.7 Схема расположения датчиков Ford Escape Hybrid

На данный момент Argo AI разработала резервные системы для торможения, рулевого управления и контроля мощности двигателя, чтобы альтернативные системы могли взять на себя управление в случае непредвиденной ситуации. Резервные источники электроэнергии доступны для нескольких компонентов, так что в случае отказа эти компоненты, в том числе компьютеры, датчики, тормозная и рулевая системы, по-прежнему получают питание для остановки транспортного средства [9].

1.4 Проблемы беспилотных технологий и возможные решения

Как и любая инновация в нынешнем мире, подверженном всё более нарастающему научно-техническому прогрессу, автономные беспилотные технологии в первую очередь сталкиваются с масштабной волной критики и скептицизма, впоследствии поднимающих различные проблемы, которые необходимо решить, чтобы заслужить доверие потребителей, надёжно укрепиться на мировом рынке в своей отрасли, обеспечить плавную разработку и надлежащую коммерциализацию беспилотных автомобилей. Некоторые технические проблемы и возможные решения подробно рассматриваются ниже.

1.4.1 Безопасность и надёжность

Перед полной коммерциализацией технология должна быть протестирована на нескольких тысячах автомобилей и нескольких миллионах километров. Надежность системы определяется расстоянием, пройденным автомобилем. Согласно требованиям, автономный автомобиль должен проехать около 291 млн миль без человеческих жертв и несчастных случаев, чтобы обеспечить 95-процентную эквивалентность водителю-человеку [10]. Эти требования к милям и полученному количеству лет остаются слишком высокими, даже если мы уменьшим требуемый процент эквивалентности.

Другим фактором для всесторонней проверки безопасности является количество «сложных миль». В некоторых регионах количество сложных дорожных ситуаций, таких как перекрытые полосы движения и левые повороты, может быть больше, чем в остальных. Несмотря на то, что автомобили, протестированные в местах с «легким пробегом», будут накапливать мили быстрее, объём обучения будет менее существенным.

Учитывая эти нюансы, испытательный центр беспилотных автомобилей «Mcity» Мичиганского университета в Анн-Арборе предложил независимый тест на безопасность, который называется Mcity ABC Test [11]. Основная цель

Mcity ABC Test — проверить работу автоматизированных транспортных средств с точки зрения безопасности. Он состоит из трёх основных компонентов:

- ускоренной оценки;
- поведенческой компетентности;
- угловых случаев.

Ускоренная оценка фокусируется на трёх основных сценариях:

- смена полосы движения;
- следование за автомобилем;
- прохождение левого поворота.

При ускоренной оценке изначально собираются натуралистичные данные о вождении. Эти данные отражают то, с чем будут сталкиваться испытуемые автомобили в нормальных условиях на дорогах общего пользования. На следующем этапе поведение водителя искажается, т. е. усиливаются рискованное поведение и внимание к «сложным милям».

Поведенческая компетентность включает в себя тестирование транспортных средств в строгих сценариях, чтобы увидеть, как они работают с точки зрения безопасности. Список Waymo [6] из их отчёта о безопасности также является одним из наборов сценариев. Всего выбрано 50 сценариев. Погода и освещение также находятся в числе факторов, которые учитываются. Условия освещения проверяются в дневное и ночное время. Такие компоненты, как радары, лидары и камеры, тестируются под дождем и снегом. После этого тестирования были окончательно отобраны 35 сценариев. Дальнейшее тестирование показывает, что для низкоскоростных шаттлов, имеющих траектории, определяемые GPS, требуется только 16 сценариев.

Угловыми считаются случаи, которые находятся на крайних точках тестовых условий, например, обнаружение автомобилей темного цвета в темном окружении с помощью камер, зигзагообразное движение бегунов по улице, велосипедисты, поворачивающие налево в условиях оживленного движения.

1.4.2 Валидация и тестирование

Автономные системы требуют всестороннего тестирования из-за своей сложности и в связи с тем, что любое решение, принимаемое программным обеспечением, напрямую влияет на жизнь человека. Для создания исчерпывающей системы, помогающей в принятии решений, используют машинное обучение, включающее в себя множество типов, таких как обучение с учителем, обучение без учителя, глубокое обучение, обучение с критиком, активное и индуктивное обучение. Для целей обнаружения объектов классификаторы алгоритмов машинного обучения необходимо обучать на большом количестве данных, однако это делает процесс тестирования еще более сложным.

Проектирование отказоустойчивой системы также является сложной задачей, поскольку требуются как минимум две независимые резервные подсистемы, чтобы в случае отказа одной части другая могла взять на себя управление.

Также распространённым методом внешнего тестирования является метод внедрения ошибок, при котором используется внешнее оборудование для внесения дефектов в аппаратное обеспечение целевой системы. Эти дефекты могут быть введены контактно (принудительным изменением напряжения или силы тока) или бесконтактно (воздействие магнитного поля или кибератак).

1.4.3 Ориентация относительно окружающей среды

Многие факторы способствуют возникновению проблем с ориентацией, с которыми сталкиваются автономные транспортные средства. Основной причиной этих проблем являются динамические ситуации на дорогах, такие как объезды дорог, строительные площадки и отсутствие дорожных знаков и разметки. Различные компании приняли несколько стратегий для решения этой проблемы.

Обработка изображений в реальном времени и подход к машинному обучению, который был развёрнут в автомобилях Tesla [5], имеет множество преимуществ. Это позволяет автомобилю адаптироваться к динамическим условиям окружающей среды. Несмотря на сложность, этот подход устранил зависимость транспортного средства от устаревших карт.

Такие компании, как General Motors и Mercedes Benz, полагаются на лидары для создания предварительно записанной трехмерной карты окрестностей. Автомобиль обнаруживает изменения в окружающей среде, используя предварительно записанную карту и лидары, и соответствующим образом реагирует на изменения. Хотя стоимость лидаров и записывающих карт высока, этот метод очень надежен.

Другой подход к преодолению ориентации на вызов предполагает создание более разумной среды. Окружающая среда информирует автомобиль об изменениях в окружающей среде. Технология «Vehicle-to-everything (V2X)» также является компонентом этого подхода. Он используется Volkswagen в своих автомобилях. Volkswagen также внедрил тестирование «Vehicle-to-Infrastructure (V2I)» с помощью интеллектуальных светофоров. Такой подход к созданию более интеллектуальной среды снижает сложность систем автономных транспортных средств.

1.4.4 Правовые проблемы

Требование правовой базы и правил является одним из наиболее важных требований для развёртывания автономных транспортных средств. Ещё одной проблемой является вопрос о том, кто будет нести ответственность в случае аварии/столкновения с беспилотным автомобилем.

С развитием автономных транспортных средств происходит четкий перенос ответственности за аварии с водителей на компании, которые проектируют и разрабатывают эти транспортные средства. Поэтому необходимо пересмотреть

законы с учетом наличия автономных транспортных средств на дорогах общего пользования. Необходима четкая и краткая политика, направленная на решение проблем потенциального потребителя.

1.4.5 Морально-этические аспекты

Ещё одна серьёзная проблема связана с принятием решений в случае возникновения чрезвычайных ситуаций. Столкнувшись со сложными дорожными ситуациями, автономные автомобили могут столкнуться с решениями, имеющими моральные последствия, такими как необходимость выбора между спасением жизни пассажиров и столкновением с ближайшим пешеходом или нажатием на тормоз, чтобы избежать столкновения с пешеходом, что может нанести ущерб жизни пассажиров. Принятие обоснованных решений в таких ситуациях может оказаться непростой задачей.

По состоянию на февраль 2020 года единственной страной в мире, в которой действуют рекомендации по принятию решений в отношении автономных транспортных средств, является Германия. В случае неизбежных аварийных ситуаций любые различия, основанные на личных характеристиках (возраст, пол, физическое или психическое состояние), строго запрещены. Также запрещается противопоставлять потерпевших друг другу. Хотя всё это может показаться идеальным, данные предпочтения могут варьироваться из-за различий в моральных установках, например, когда люди склонны отдавать предпочтение спасению молодых перед пожилыми, спасать больше жизней, а не меньше жизней и т. п.

Чтобы преодолеть эти проблемы, все заинтересованные стороны должны быть прозрачны в отношении выбора и обоснования, лежащего в его основе, при анализе связанных с этим рисков и выгод.

1.4.6 Финансовые проблемы

Высокая стоимость разработки и внедрения автономных транспортных средств, является проблемой (см. Рис. 1.8). Сложность технологий и компонентов, таких как датчики и коммуникационные устройства, используемые в автомобилях с более высоким уровнем автоматизации, может привести к тому, что эти функции будут доступны только в серийных автомобилях премиум-класса, что вызывает вопросы о доступности для конечных потребителей.

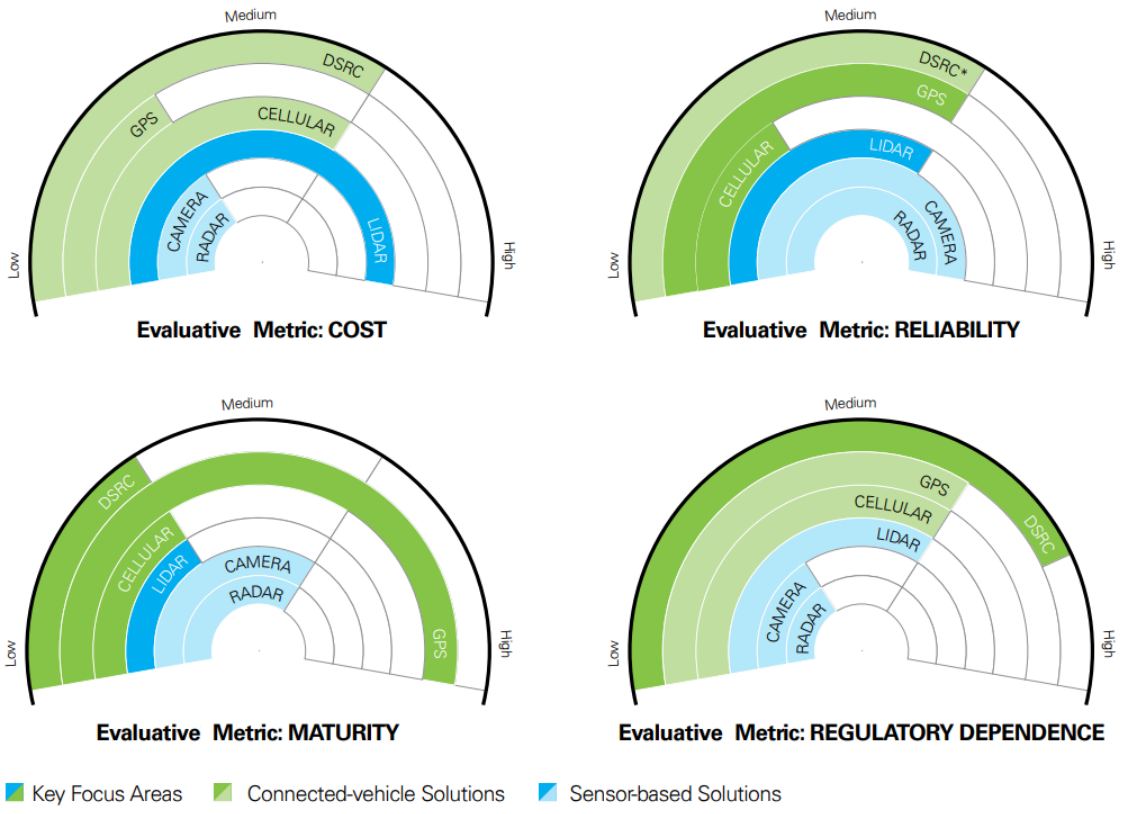

Рисунок 1.8 Метрики оценки компонент беспилотных технологий

Одним из возможных решений проблемы высоких затрат может быть использование модели аренды (каршеринга) транспортных средств, эксплуатируемых коммерческими организациями. Роботы-такси, подобные тем, которые Cruise планирует использовать для своего автомобиля Origin, основаны на этой модели. Эта модель привела бы к распределению затрат на большое количество людей.

Таким образом, чтобы обеспечить широкое внедрение автономных транспортных средств в будущем, технология должна быть доступной.

1.5 Заключение

Хотя автономные автомобильные компании добились феноменального прогресса в технологии, пройдёт много лет, прежде чем полностью автономные автомобили станут общедоступными. На данном этапе может быть невозможно указать конкретный год. По некоторым прогнозам, к 2035 году автомобили могут стать полностью автономными. Несмотря на то, что технология развивается, люди также должны быть готовы ее использовать. Кроме того, проблемы, обсуждавшиеся выше, должны быть преодолены, чтобы обеспечить плавное и гибкое развитие технологии. На круговой диаграмме (см. Рис. 1.9) проиллюстрированы ключевые компоненты, которые на данный момент, в средней и долгосрочной перспективе должны взаимодействовать между собой, чтобы обеспечить успешное внедрение технологий беспилотного вождения:

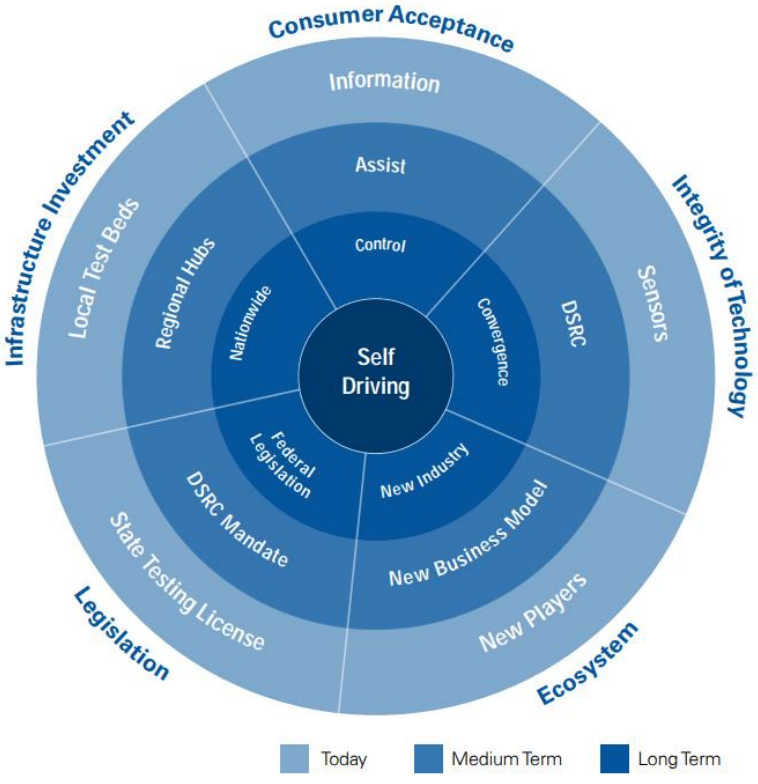

Рисунок 1.9 Компоненты концепта беспилотных автомобилей

ГЛАВА 2
ТЕХНОЛОГИИ БЕСПИЛОТНЫХ АВТОМОБИЛЕЙ

2.1 Алгоритмы беспилотных транспортных средств

Как и большинство инновационных технологий, технология беспилотных автомобилей не может обходиться без систем сбора информации и алгоритмов их обработки. Алгоритмы беспилотных транспортных средств в своём большинстве включают в себя:

- сенсорное сканирование окружающей среды — извлечение важной и необходимой для систем управления автомобилем информации из необработанных данных, полученных датчиками из дорожного окружения в реальном времени;
- восприятие окружения — локализация и оценка движения автомобиля в пространстве, сбор данных об окружающей среде и дорожном окружении;
- принятие решений — выполнение определённых действий и принятие необходимых мер, направленных на обеспечение надежного и безопасного движения автомобиля по спланированным инструкциям для достижения той или иной конечной цели.

Более подробно некоторые из наиболее популярных технологий и алгоритмов, используемых в современных беспилотных транспортных средствах, будут рассмотрены далее в этой главе.

2.2 Сенсорное сканирование окружающей среды

Как правило, на борту беспилотного автомобиля располагается несколько основных сенсоров и датчиков, каждый из которых имеет определённый

функционал из соображений надежности и безопасности, играющий свою собственную роль в общей совокупности сенсорного сканирования окружающей среды. Рассмотрим детально наиболее распространённые из них:

2.2.1 GPS/INS

Автомобильная локализация достигается благодаря навигационной системе, оснащённой глобальной системой позиционирования (GPS) и географической информационной системой для сбора информации о местоположении (с помощью вычисления пространственных координат). Система определения местоположения использует инерциальную навигационную систему (INS) для определения относительного местоположения транспортного средства с высокой частотой.

GPS – довольно точная система, но она имеет низкую частоту обновления (всего около 10 Гц) и поэтому не может предоставлять информацию в режиме реального времени. У сенсоров INS наблюдается тенденция накопления погрешностей, что впоследствии приводит к ухудшению глобальных оценок местоположения. Однако, INS может предоставлять обновления чаще, чем GPS, (с частотой 1 кГц). Комбинация навигационных систем GPS и INS предоставляет точные данные о местоположении транспортных средств в реальном времени с минимальной задержкой.

Также автомобильная локализация может осуществляться посредством GNSS (совокупности нескольких спутниковых систем позиционирования высокой точности: GPS, ГЛОНАСС, Galileo и BeiDou) и спутниковых систем дифференциальной коррекции (SBAS – Satellite Based Augmentation Systems, EGNOS – European Geostationary Navigation Overlay Service) (см. Рис. 2.1).

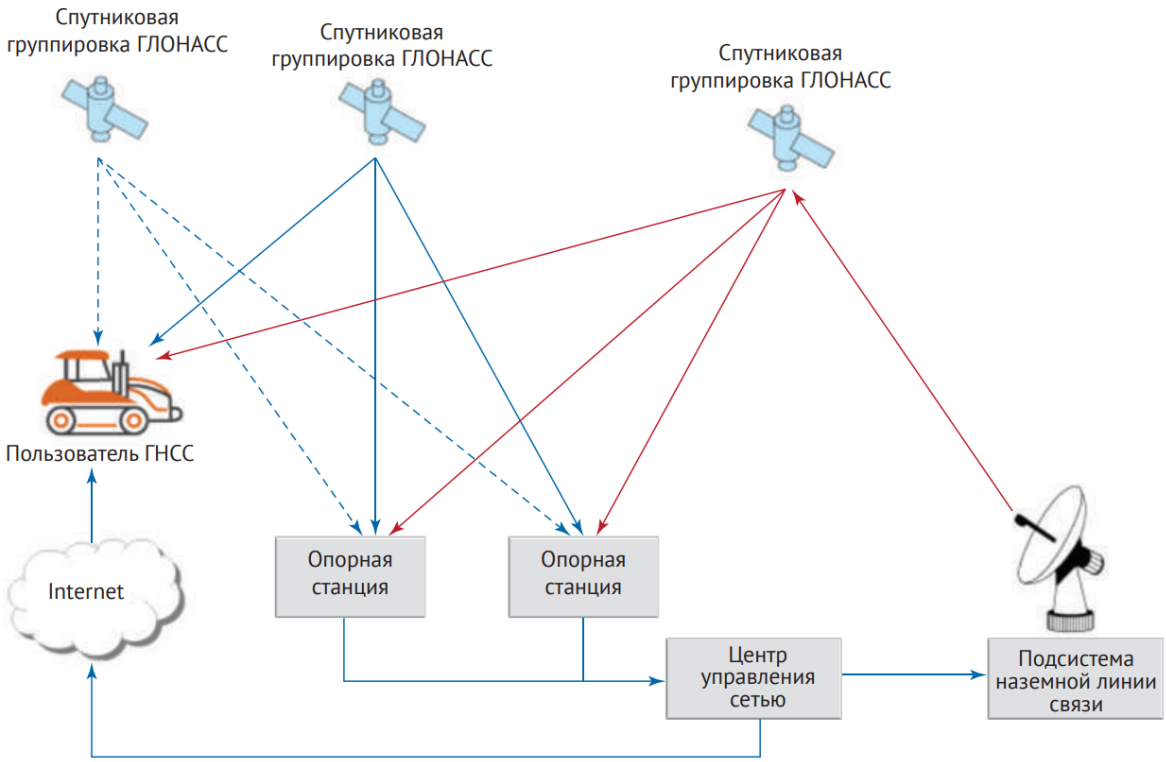

Рисунок 2.1 Спутниковая система навигации GNSS [4]

2.2.2 LIDAR

Для обзора окружающей среды, местности, определения местоположения и обхода препятствий используются три основных метода:

- лазерное восприятие;
- визуальное восприятие;
- радиолокационное восприятие.

Технология лазерного восприятия LIDAR (Light Detection and Ranging) используется для предотвращения столкновений и в ситуациях, требующих экстренного торможения. Системы LIDAR излучают несколько лазерных импульсов в секунду, которые отражаются после взаимодействия с окружающими предметами, что помогает создать трехмерное представление путем вычислений, основанных на скорости света и расстоянии, пройденном лазерным импульсом.

Восприятие радара используется для измерения расстояния путем расчёта времени, необходимого волне, передаваемой радарным датчиком, для возвращения. Связь между транспортными средствами ближнего действия используется беспилотными автомобилями, чтобы они могли общаться с окружающей средой и другими транспортными средствами в режиме реального времени.

Благодаря высокой точности лидары применяются для создания HD-карт, определения местоположения движущегося транспортного средства на HD-картах, обнаружения препятствий и т. п. Как правило, блок лидара содержит вращающийся с частотой 10 оборотов в минуту 64-лучевой лазер (с длиной волны 600–1500 нм), осуществляющий около 1.3 млн считываний в секунду (см. Рис. 2.2).

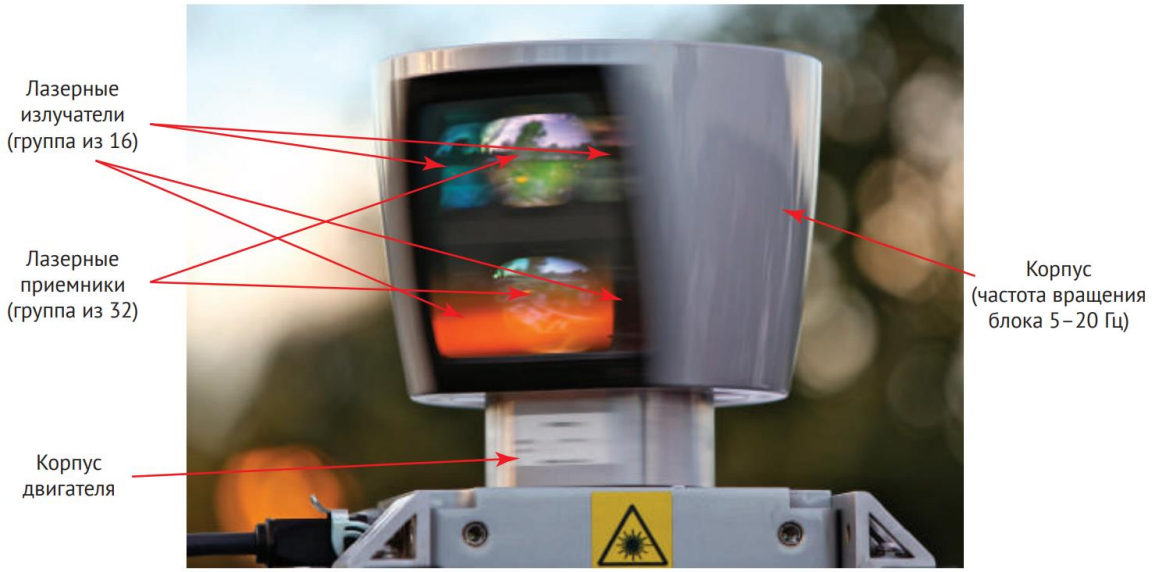

Рисунок 2.2 LIDAR Velodyne HDL-64 [4]

2.2.3 Камеры

В основном камеры используются для отслеживания и распознавания объектов, а также для решения задач типа выбора полосы движения, обнаружения светофоров, пешеходов и т. д. В современных моделях, как правило, по всему периметру кузова беспилотного автомобиля устанавливают

восемь или более камер с высоким разрешением, благодаря которым осуществляется обнаружение, распознавание и отслеживание объектов спереди, сзади и по обе стороны транспортного средства. Такие камеры работают с частотой 60 Гц и в совокупности генерируют около 1.8 Гб необработанных данных в секунду.

2.2.4 Радар и Сонар

Радар и сонар используются в беспилотных автомобилях для обнаружения и уклонения от препятствий. Датчики в режиме реального времени собирают данные об окружающей среде, включая расстояние до ближайшего объекта, скорость и направление движения автомобиля, его ускорение и препятствия, с которыми он может столкнуться.

Когда датчики сообщают системе, что автомобиль приближается к какому-либо объекту, и возникает опасность столкновения, он тормозит или поворачивает, чтобы избежать препятствия, согласно заранее спланированным инструкциям. Генерируемые радаром и сонаром данные практически не требуют предварительной обработки и обычно поступают непосредственно в процессор управления беспилотными системами, что позволяет реализовать такие «экстренные» функции, как резкий поворот, торможение или предварительное натяжение ремней безопасности.

2.3 Восприятие дорожного окружения

Основными задачами этапа восприятия дорожного окружения являются локализация, распознавание и отслеживание объектов с помощью данных, собранных различными сенсорами и датчиками на этапе сканирования.

Для эффективной локализации используют связку GPS/INS, что обеспечивает точную навигацию и позиционирование автомобиля за счёт GPS-систем и предоставляет моментальные обновления за счёт INS-систем.

Чтобы объединить преимущества двух сенсорных систем обычно используют фильтр Калмана. Суть этого метода представлена на рисунке (2.3):

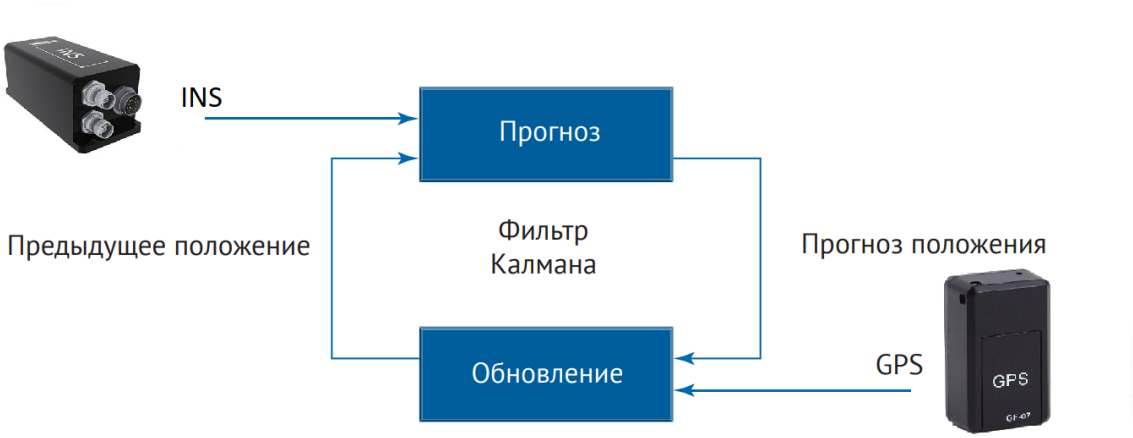

Рисунок 2.3 Локализация при помощи GPS/INS

Система INS каждую миллисекунду обновляет положение автомобиля, но со временем накапливаются погрешности. Тем временем, от системы GPS каждые сто миллисекунд автомобиль получает обновления местоположения, которые помогают исправить эти погрешности. Благодаря этому методу комбинация GPS/INS может обеспечивать быструю и точную локализацию беспилотного автомобиля.

Также для пространственной локализации используются данные с высокочастотных камер (см. Рис. 2.4):

1. путём триангуляции пар стереоизображений составляется карта расхождений между ними, по которой далее извлекается информация о глубине для каждой точки изображения;
2. сопоставляются характерные признаки между последовательными кадрами стереоизображения и устанавливаются корреляции между ними в разных кадрах, по которым оценивается перемещение автомобиля за время, прошедшее между двумя прошлыми кадрами;
3. при помощи сравнения выявленных особенностей и изменений с теми, что есть на известной карте, вычисляются данные о текущем положении автомобиля.

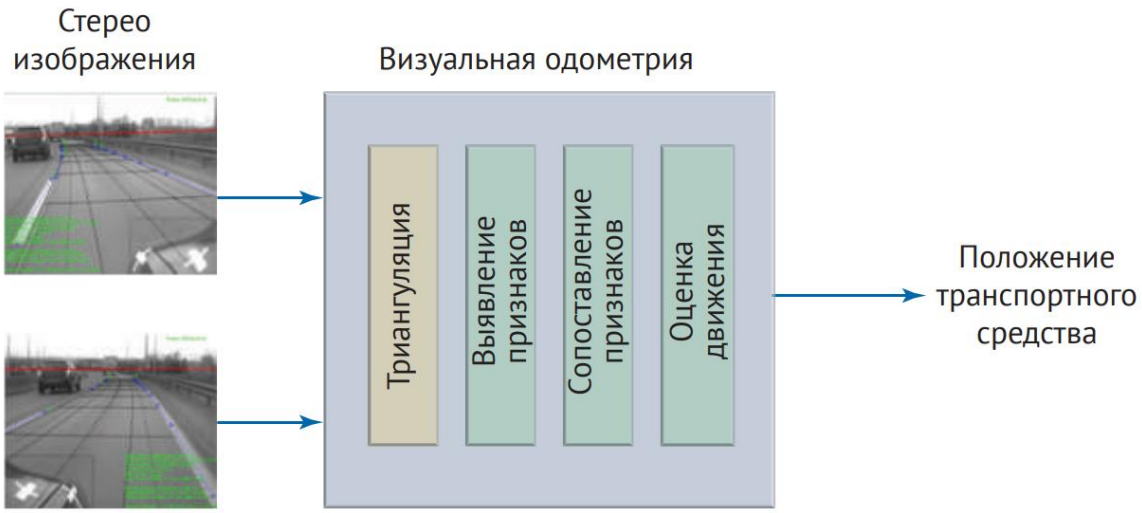

Рисунок 2.4 Стереовизуальная одометрия

Такая локализация является не очень надёжной из-за чувствительности камер к условиям глубины и освещения, поэтому в дополнение к данной одометрии используют монокулярные и инерциальные визуальные одометрии, колёсные одометрии, а также технологию лидаров.

Чтобы избежать зависимости от погодных условий, технология лидаров использует методы фильтра частиц: лидар генерирует облако точек, которое повторяет очертания окружающей среды и сравнивается с уже имеющейся картой с помощью фильтра частиц. Относительно этих карт применяется метод фильтра частиц, который сопоставляет данные лидара с картой, тем самым определяя положение движущегося автомобиля. Данный метод обеспечивает локализацию в реальном времени с точностью до нескольких сантиметров и эффективен при сложных дорожных условиях в различных местностях.

Для достижения надёжной и точной локализации транспортного средства используются все датчики в совокупности, объединяя преимущества и перекрывая недостатки друг друга. Дополнительно, для повышения уровня восприятия окружающей среды, большинство беспилотных автомобилей используют технологии глубокого обучения и нереволюционные алгоритмы, позволяющие относительно точно распознавать и отслеживать объекты. Эти и другие концепции машинного обучения будут рассмотрены в следующих главах.

2.4 Принятие решений

На этапе принятия решений беспилотный автомобиль, основываясь на исследовании данных, полученных из окружающей среды посредством датчиков и сенсоров, прогнозирует и выстраивает безопасный и эффективный план дальнейших своих действий в режиме реального времени.

Для планирования безопасных и эффективных передвижений система принятия решений беспилотного автомобиля генерирует вероятностную модель наборов достижимых исходов того или иного действия и связывает эти наборы достижимости с распределением их вероятностей, тем самым прогнозируя свои действия и поведение дорожного окружения, а также принимая необходимые решения на основе этих прогнозов.

Стоит отметить, что при планировании действий происходит создание сложных многоуровневых сценариев (см. Рис. 2.5) и вычисление индивидуальных решений на их основе алгоритмами машинного обучения. Данные сценарии могут зависеть, например, от положения самого транспортного средства, от поведения идущего впереди транспортного средства, ориентации дорожного движения, светофоров, дорожных переходов, пешеходов, перекрёстков, разделения или слияния полос и т. п. Сведение к окончательному синтетическому решению происходит путём арбитражного синтезирования и объединения всех позитивных индивидуальных решений в целях гарантии безопасности движения беспилотного автомобиля.

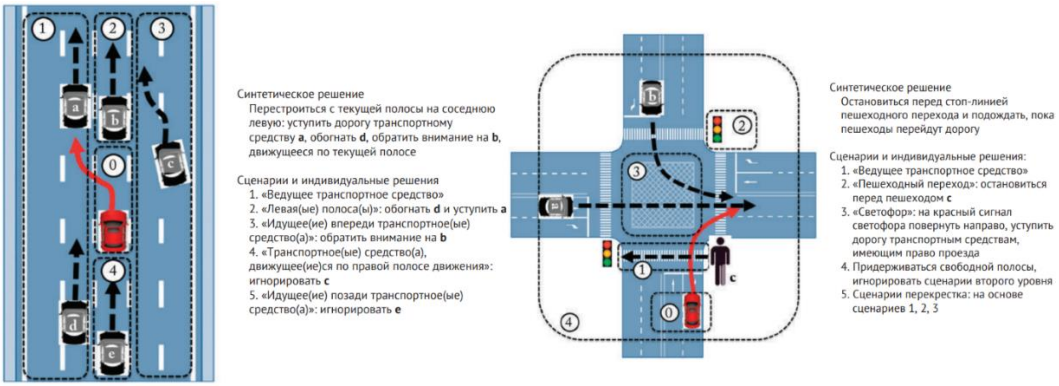

Рисунок 2.5 Примеры многоуровневых сценариев [4]

2.5 Заключение

Беспилотные автомобили представляют собой интеграцию множества технологий, включающих: высокоточные датчики и высокочувствительные сенсоры, обеспечивающие локализацию транспортного средства и восприятие окружающей среды в реальном времени, алгоритмы машинного обучения и искусственного интеллекта, используемые для планирования сложных многоуровневых сценариев вождения и построения индивидуальных решений на их основе, а также клиентские, аппаратные и облачные платформы, выполняющие функции обработки и хранения распределённых данных и объединяющие все технические составляющие беспилотника для удовлетворения требований бесперебойного функционирования и производительности, высокоуровневой безопасности и надежности.

Автономные беспилотные технологии успешно начали своё укрепление на мировом рынке и развитие во многих глобальных сферах производств, поэтому в ближайшем будущем стоит ожидать ускорения процесса совершенствования существующих и внедрения инноваций в беспилотные технологии, а также всё большую их популяризацию в нашей повседневной жизни.

ГЛАВА 3
ГЛУБОКОЕ ОБУЧЕНИЕ В АЛГОРИТМАХ БЕСПИЛОТНЫХ АВТОМОБИЛЕЙ

3.1 Технологии принятия поведенческих решений

Основу технологии беспилотных транспортных средств составляют автономные системы принятия решений, которые обрабатывают потоки наблюдений, поступающих от различных бортовых источников, таких как камеры, радары, лидары, ультразвуковые датчики, устройства глобальной системы позиционирования и инерциальные датчики и т. п. Эти наблюдения используются бортовым компьютером автомобиля для планирования сложных многоуровневых сценариев вождения и построения индивидуальных решений на их арбитражной основе. Индивидуальные решения вычисляются либо в модульном конвейере восприятия-планирования-действия (см. Рис. 3.1 а), либо в режиме обучения End2End (см. Рис. 3.1 б), при котором информация, получаемая системами восприятия дорожного окружения, напрямую сопоставляется с управляющими контроллерами транспортного средства.

Компоненты модульного конвейера могут быть разработаны либо на основе искусственного интеллекта и методологий глубокого обучения, либо с использованием классических подходов, не связанных с обучением. Возможны различные перестановки компонентов, основанных на обучении, и компонентов, не основанных на обучении: например, детектор объектов на основе глубокого обучения предоставляет входные данные для классического алгоритма планирования пути A* или Дейкстры.

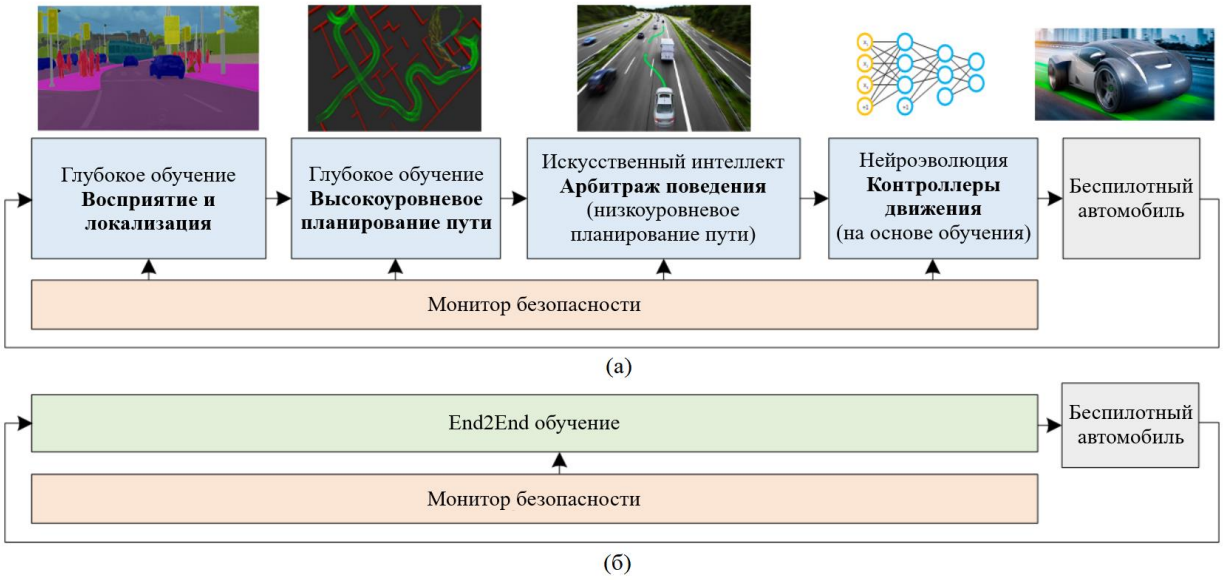

Рисунок 3.1 Беспилотный автомобиль на основе глубокого обучения

Монитор безопасности предназначен для обеспечения безопасности каждого модуля. Модульный конвейер на рисунке (3.1 а) иерархически разложен на четыре компонента, которые можно спроектировать с использованием либо подходов глубокого обучения и искусственного интеллекта, либо классических нереволюционных методов. Эти компоненты включают в себя:

- Восприятие и локализацию на основе глубокого обучения;
- Высокоуровневое планирование пути на основе глубокого обучения;
- Арбитраж поведения и низкоуровневое планирование пути;
- Обучаемые контроллеры движения.

Учитывая маршрут, запланированный к прохождению по дорожному окружению, первая задача беспилотного автомобиля — воспринять и локализовать себя в окружающей среде. На основе этого представления планируется непрерывный путь, и дальнейшие действия автомобиля определяются системой арбитража поведения. Наконец, система управления движением оперативно исправляет ошибки, возникающие при выполнении запланированного действия. Обзор классических методологий проектирования, не связанных с глубоким обучением и искусственными интеллектом для этих четырех компонентов подробным образом изложен в [4].

3.2 Обзор технологий глубокого обучения

Далее в этой главе будут рассмотрены технологии глубокого обучения и искусственного интеллекта, используемые в беспилотных транспортных средствах, а также различные методологии, используемые для разработки иерархического процесса принятия решений, описанного выше.

Наиболее распространенными методологиями глубокого обучения и искусственного интеллекта, применяемыми к беспилотным автомобилям, на сегодняшний день являются:

- свёрточные нейронные сети (CNN);
- рекуррентные нейронные сети (RNN);
- глубокое обучение с подкреплением (DRL);
- нереволюционные алгоритмы.

Введём следующие обозначения для описания последовательностей и векторных переменных, зависящих от времени. Векторы и матрицы будем обозначать жирным шрифтом. Значения таких переменных определяются либо для одного дискретного временного шага t, записываемого как верхний индекс $< t >$, либо как дискретная или векторная последовательность, определенная в интервале времени $< t, t + k >$, где k обозначает длину последовательности. Например, значение переменной некоторого состояния **z** определяется либо в дискретное время t как $\boldsymbol{z}^{<t>}$, либо в пределах временного интервала последовательности $< t, t + k >$ как $\boldsymbol{z}^{<t,t+k>}$.

3.3 Глубокие свёрточные нейронные сети

Свёрточные нейронные сети (CNN) в основном используются для обработки пространственной информации, такой как изображения, и могут рассматриваться как экстракторы признаков изображений и универсальные аппроксиматоры нелинейных функций.

До появления глубокого обучения системы компьютерного зрения реализовывались на основе созданных вручную математических функций, таких как функция HAAR, локальные бинарные шаблоны (LBP) или гистограммы ориентированных градиентов (HoG). По сравнению с этими традиционными функциями, созданными определёнными математическими инструментами, сверточные нейронные сети могут автоматически изучать представление пространства признаков, закодированное в обучающем наборе.

Свёрточные нейронные сети можно в общих чертах воспринимать как очень приблизительные аналогии различных частей зрительной коры млекопитающих [12]. Изображение, сформированное на сетчатке, через таламус отправляется в зрительную кору. Каждое полушарие мозга имеет свою зрительную кору. Зрительная информация поступает в зрительную кору перекрёстным образом: левая зрительная кора получает информацию от правого глаза, а правая — от левого глаза. Информация обрабатывается в соответствии с теорией двойных потоков [12], согласно которой зрительная информация следует двум основным потокам: вентральному потоку, отвечающему за визуальную идентификацию и распознавание объектов, и дорсальному потоку, используемому для установления пространственных отношений между объектами. Нейронная сеть имитирует функционирование вентрального потока, при котором разные области мозга чувствительны к определённым особенностям поля зрения. Более ранние клетки мозга в зрительной коре активируются резкими переходами в поле зрения таким же образом, как детектор границ выделяет резкие переходы между соседними пикселями на изображении. Эти переходы далее используются мозгом для аппроксимации частей объекта и, наконец, для оценки абстрактных представлений объектов.

Свёрточные нейронные сети параметризуются своим тензором весов $\theta = [\mathbf{W}, \mathbf{b}]$, где $\mathbf{W}$ — набор весов, управляющих межнейронными связями, а $\mathbf{b}$ — набор значений смещения нейронов. Набор весов $\mathbf{W}$ организован как фильтры изображений с коэффициентами, вычисленными во время обучения. Свёрточные

слои в нейронной сети используют локальные пространственные корреляции пикселей изображения для изучения трансляционно-инвариантных фильтров свёртки, которые фиксируют дискриминантные признаки изображения.

Рассмотрим многоканальное представление сингала $\boldsymbol{M}_k$ в слое k, которое представляет собой поканальную интеграцию представлений сигнала $\boldsymbol{M}_{k,c}, c \in \mathbb{N}$. Представление сигнала может быть сгенерировано в слое $k+1$ как:

$$\boldsymbol{M}_{k+1,l} = \varphi\left(\boldsymbol{M}_k * \boldsymbol{w}_{k,l} + \boldsymbol{b}_{k,l}\right), \tag{3.1}$$

Где $\boldsymbol{w}_{k,l} \in \boldsymbol{W}$ — фильтр свёртки с тем же количеством каналов, что и $\boldsymbol{M}_k$, $\boldsymbol{b}_{k,l} \in \boldsymbol{b}$ представляет смещение, l — индекс канала, а символ «$*$» обозначает операцию свертки. $\varphi(\cdot)$ — функция активации, применяемая к каждому пикселю входного сигнала. Как правило, функция ReLU является наиболее часто используемой функцией активации в приложениях компьютерного зрения. Последний слой нейронной сети обычно представляет собой полносвязный слой, который действует как дискриминатор объектов в представлении объектов высокого уровня абстракции.

При обучении с учителем отклик $R(\cdot, \theta)$ нейронной сети может быть обучен с использованием тренировочного набора данных $\mathcal{D} = [(\boldsymbol{x}_1, y_1), \dots, (\boldsymbol{x}_m, y_m)]$, в котором $\boldsymbol{x}_i$ — выборка данных, y_i — соответствующая метка, а m — количество обучающих примеров. При обучении регрессионных оценок оптимальные параметры сети можно рассчитать с помощью оценки максимального правдоподобия, например, в частности с помощью простой функции наименьших квадратов:

$$\theta^* = \arg\max_{\theta} \mathcal{L}(\theta, \mathcal{D}) = \arg\min_{\theta} \sum_{i=1}^{m} (R(\boldsymbol{x}_i, \theta) - y_i)^2. \tag{3.2}$$

В целях классификации ошибка метода наименьших квадратов обычно заменяется кросс-энтропией или функциями потерь отрицательного логарифмического правдоподобия. Задача оптимизации (3.2) обычно решается с помощью стохастического градиентного спуска и алгоритма обратного распространения для оценки градиента. На практике используются разные варианты оптимизаторов, такие как Adam или AdaGrad.

3.4 Рекуррентные нейронные сети

Среди методов глубокого обучения рекуррентные нейронные сети (RNN) особенно популярны при обработке данных временной последовательности, таких как тексты или видеопотоки. В отличие от обычных нейронных сетей, рекуррентные нейронные сети содержат в своей ячейке памяти цикл обратной связи, зависящий от времени. Учитывая зависящие от времени входную последовательность $[\boldsymbol{s}^{<t-\tau_i>}, \dots, \boldsymbol{s}^{<t>}]$ и выходную последовательность $[\boldsymbol{z}^{<t+1>}, \dots, \boldsymbol{z}^{<t+\tau_0>}]$, рекуррентная нейронная сеть может быть «развёрнута» $\tau_i + \tau_0$ раз для создания беспетлевой архитектуры сети, соответствующей входной длине (см. Рис. 3.2), где t представляет собой временной индекс, а τ_i и τ_0 — длины входной и выходной последовательностей соответственно. Развернутая сеть имеет $\tau_i + \tau_0 + 1$ идентичных слоев, то есть каждый слой имеет одинаковые обученные веса. После процедуры развёртывания нейронную сеть можно обучить с помощью алгоритма обратного распространения во времени.

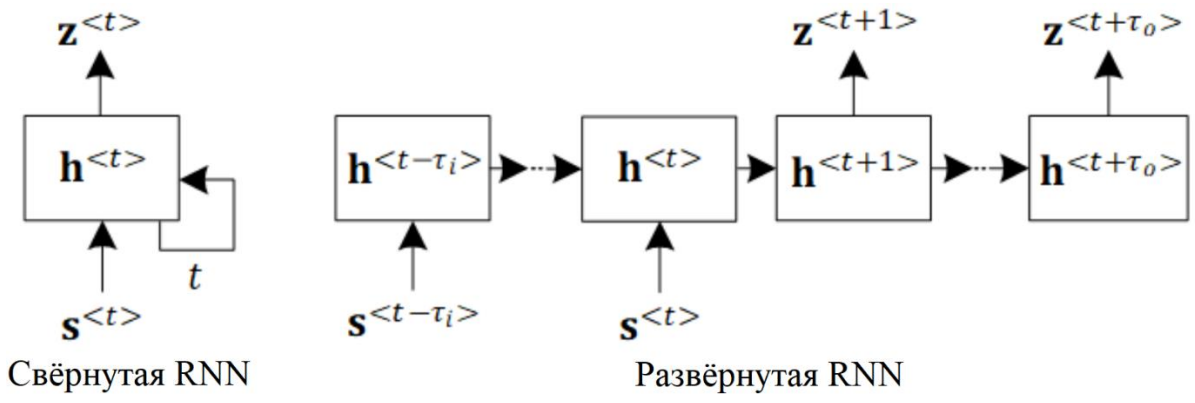

Рисунок 3.2 Свернутая и развернутая рекуррентная нейронная сеть

По сравнению с обычной нейронной сетью единственное отличие рекуррентных нейронных сетей состоит в том, что изученные веса в каждой развернутой копии сети усредняются, что позволяет сети совместно использовать одни и те же веса с течением времени.

Основной проблемой при использовании базовых рекуррентных нейронных сетей является затухающий градиент, возникающий во время обучения: сигнал градиента может быть увеличен в большое количество раз в зависимости от количества временных шагов. Следовательно, традиционная RNN не подходит для предсказания долгосрочных зависимостей. Если сеть очень глубокая или обрабатывает длинные последовательности, градиент выходных данных сети будет с трудом распространяться обратно, чтобы повлиять на веса более ранних слоёв. При исчезновении градиента веса сети не будут эффективно обновляться, что сведётся к очень малым значениям весов.

В качестве улучшенной альтернативы также используют нейронные сети с долгой краткосрочной памятью (LSTM), представляющие собой аппроксиматоры нелинейных функций для оценки временных зависимостей в исходных данных. В отличие от традиционных рекуррентных нейронных сетей, нейронные сети с долгой краткосрочной памятью решают проблему затухающего градиента за счёт внедрения трёх специальных фильтров, которые управляют вводом, выводом и состоянием памяти.

Рекуррентные слои используют временные корреляции исходных данных для изучения нейронных структур, зависящих от времени. Рассмотрим состояние памяти $\boldsymbol{c}^{<t-1>}$ и выходное состояние $\boldsymbol{h}^{<t-1>}$ в сети с долгой краткосрочной памятью, выбранное на временном шаге $t-1$, а также входные данные $\boldsymbol{s}^{<t>}$ при времени t. Включением и отключением фильтров управляет сигмовидная функция $\sigma(\cdot)$ текущего входного сигнала $\boldsymbol{s}^{<t>}$ и выходного сигнала конечного момента времени $\boldsymbol{h}^{<t-1>}$:

$$\begin{cases} \Gamma_u^{<t>} = \sigma(\boldsymbol{W}_u \boldsymbol{s}^{<t>} + \boldsymbol{U}_u \boldsymbol{h}^{<t-1>} + \boldsymbol{b}_u) \\ \Gamma_f^{<t>} = \sigma(\boldsymbol{W}_f \boldsymbol{s}^{<t>} + \boldsymbol{U}_f \boldsymbol{h}^{<t-1>} + \boldsymbol{b}_f), \\ \Gamma_o^{<t>} = \sigma(\boldsymbol{W}_o \boldsymbol{s}^{<t>} + \boldsymbol{U}_o \boldsymbol{h}^{<t-1>} + \boldsymbol{b}_o) \end{cases} \quad (3.3)$$

где $[\Gamma_u^{<t>}, \Gamma_f^{<t>}, \Gamma_o^{<t>}]$ — являются функциями входных фильтров, фильтров забывания и выходных фильтров соответственно (см. Рис. 3.3).

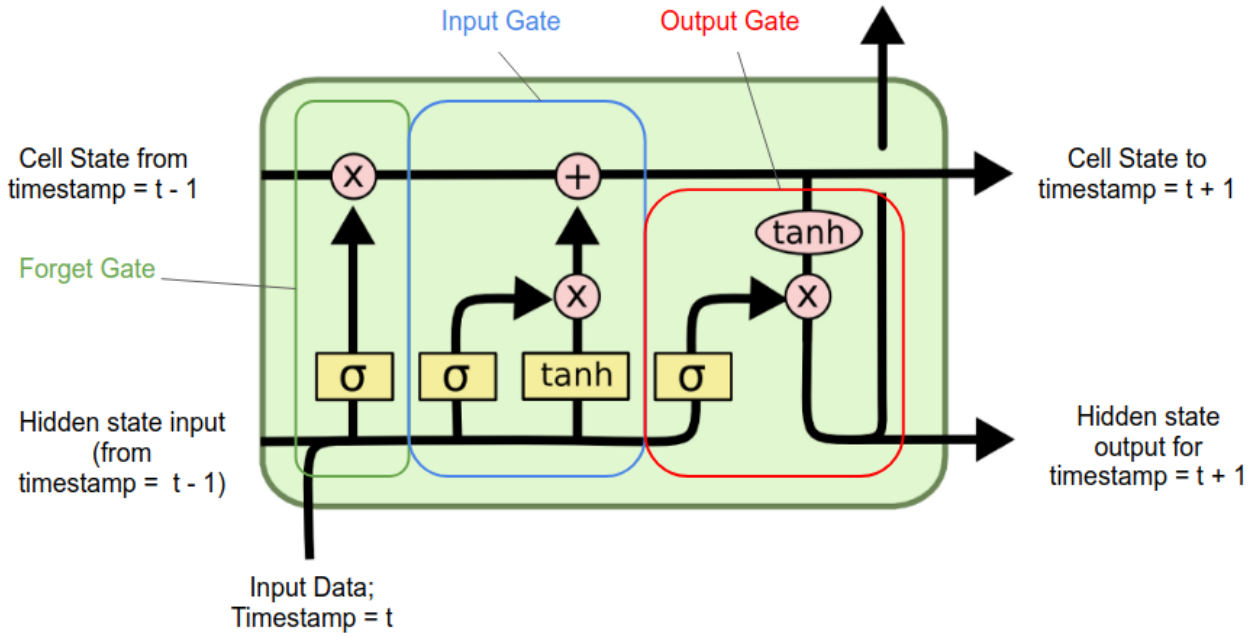

Рисунок 3.3 Ячейка нейронной сети с долгой краткосрочной памятью

Таким образом, учитывая текущее наблюдение, состояние памяти $\boldsymbol{c}^{<t>}$ будет обновлено следующим образом:

$$\boldsymbol{c}^{<t>} = \Gamma_u^{<t>} * \tanh(\boldsymbol{W}_c \boldsymbol{s}^{<t>} + \boldsymbol{U}_c \boldsymbol{h}^{<t-1>} + \boldsymbol{b}_c) + \Gamma_f^{<t>} * \boldsymbol{c}^{<t-1>}. \quad (3.4)$$

Новый выход нейронной сети $\boldsymbol{h}^{<t>}$ будет вычисляться как:

$$\boldsymbol{h}^{<t>} = \Gamma_o^{<t>} * \tanh(\boldsymbol{c}^{<t>}). \quad (3.5)$$

Нейронная сеть с долгой краткосрочной памятью параметризуется как $\theta = [\boldsymbol{W}_i, \boldsymbol{U}_i, \boldsymbol{b}_i]$, где $\boldsymbol{W}_i$ представляет собой веса фильтров сети и ячейки памяти, умноженные на входное состояние, $\boldsymbol{U}_i$ — веса, управляющие активациями, $\boldsymbol{b}_i$ —

набор значений смещений нейронов, а символ «$*$» обозначает поэлементное умножение.

Если при обучении задана обучающая выборка $\mathcal{D} = [(\boldsymbol{s}_1^{<t-\tau_i,t>}, \boldsymbol{z}_1^{<t+1,t+\tau_0>}), \dots, (\boldsymbol{s}_q^{<t-\tau_i,t>}, \boldsymbol{z}_q^{<t+1,t+\tau_0>})]$, то есть q независимых пар наблюдений с метками $\boldsymbol{z}^{<t,t+\tau_0>}$, то нейронная сеть с долгой краткосрочной памятью $Q(\cdot, \theta)$ может быть обучена с использованием оценки максимального правдоподобия:

$$\begin{aligned} \theta^* &= \arg\max_{\theta} \mathcal{L}(\theta, \mathcal{D}) \\ &= \arg\min_{\theta} \sum_{i=1}^{m} l_i(Q(\boldsymbol{s}_i^{<t-\tau_i,t>}, \theta), \boldsymbol{z}_i^{<t+1,t+\tau_0>}) \\ &= \arg\min_{\theta} \sum_{i=1}^{m} \sum_{t=1}^{\tau_0} l_i^{<t>}(Q^{<t>}(\boldsymbol{s}_i^{<t-\tau_i,t>}, \theta), \boldsymbol{z}_i^{<t>}), \end{aligned} \tag{3.6}$$

где входная последовательность наблюдений $\boldsymbol{s}^{<t-\tau_i,t>} = [\boldsymbol{s}^{<t-\tau_i>}, \dots, \boldsymbol{s}^{<t-1>}, \boldsymbol{s}^{<t>}]$ состоит из τ_i последовательных выборок данных, функция $l(\cdot,\cdot)$ — функция потерь логистической регрессии, а t — временной индекс.

В терминологии рекуррентных нейронных сетей процедура оптимизации в уравнении (3.6) обычно используется для обучения архитектур типа «многие ко многим», таких как на рисунке (3.2), где входные и выходные состояния представлены временными последовательностями экземпляров данных τ_i и τ_0 соответственно. Задача оптимизации такого рода обычно решается с использованием градиентных методов, таких как стохастический градиентный спуск и алгоритм обратного распространения во времени для расчета градиентов нейронной сети.

3.5 Глубокое обучение с подкреплением

Далее рассмотрим концепцию глубокого обучения с подкреплением (DRL) как задачу автономного вождения с использованием формализма POMDP (Partially Observable Markov Decision Process).

В системе POMDP агент, которым в нашем случае является беспилотный автомобиль, ощущает окружающую среду с наблюдением $\boldsymbol{I}^{<t>}$, выполняет действие $a^{<t>}$ в состоянии $\boldsymbol{s}^{<t>}$, взаимодействует со своим окружением через полученное вознаграждение $R^{<t+1>}$ и переходит в следующее состояние $\boldsymbol{s}^{<t+1>}$ в соответствии с функцией перехода $T^{\boldsymbol{s}^{<t+1>}}_{\boldsymbol{s}^{<t>},a^{<t>}}$. В автономном вождении на основе обучения с подкреплением задача состоит в том, чтобы изучить оптимальную политику вождения для навигации из состояния $\boldsymbol{s}^{<t>}_{start}$ в состояние назначения $\boldsymbol{s}^{<t+k>}_{dest}$ с учетом наблюдения $\boldsymbol{I}^{<t>}$ в момент времени t и состояния системы $\boldsymbol{s}^{<t>}$. $\boldsymbol{I}^{<t>}$ представляет воспринимаемое дорожное окружение, а k — количество временных шагов, необходимых для достижения состояния назначения $\boldsymbol{s}^{<t+k>}_{dest}$.

В терминологии обучения с подкреплением вышеуказанная проблема может быть смоделирована как $M_{POMDP} \coloneqq (I, S, A, T, R, \gamma)$, где:

- I — множество наблюдений, а $\boldsymbol{I}^{<t>} \in I$ определяется как наблюдение за дорожным окружением в момент времени t;
- S — конечное множество состояний, а $\boldsymbol{s}^{<t>}$ — состояние автомобиля в момент времени t, определяемое как положение, направление и скорость транспортного средства;
- A — конечный набор действий, позволяющий автомобилю перемещаться в окружающей среде $\boldsymbol{I}^{<t>}$, где $a^{<t>}$ — действие, выполняемое транспортным средством в момент времени t;

- $T: S \times A \times S \to [0,1]$ — стохастическая функция перехода, в которой $T_{\boldsymbol{s}^{<t>},a^{<t>}}^{\boldsymbol{s}^{<t+1>}}$ описывает вероятность попадания в состояние $\boldsymbol{s}^{<t+1>}$ после выполнения действия $a^{<t>}$ в состоянии $\boldsymbol{s}^{<t>}$;
- $R: S \times A \times S \to \mathbb{R}$ — скалярная функция вознаграждения, управляющая оценкой действия $a^{<t>}$, определяемая для перехода состояния $\boldsymbol{s}^{<t>} \to \boldsymbol{s}^{<t+1>}$ в момент времени t как $R_{\boldsymbol{s}^{<t>},a^{<t>}}^{\boldsymbol{s}^{<t+1>}} \in \mathbb{R}$, которая количественно определяет, насколько хорошо беспилотный автомобиль действовал при достижении следующего состояния;
- γ — коэффициент подкрепления, контролирующий важность будущих вознаграждений по сравнению с текущими.

Учитывая предложенную функцию вознаграждения и произвольную траекторию состояния $[\boldsymbol{s}^{<0>}, \boldsymbol{s}^{<1>}, \dots, \boldsymbol{s}^{<k>}]$ в пространстве наблюдения, в любой момент времени $t^* \in [0,1, \dots, k]$, соответствующее кумулятивное будущее подкрепляющее вознаграждение определяется как:

$$R^{<t^*>} = \sum_{t=t^*}^{k} \gamma^{<t-t^*>} r^{<t>}, \tag{3.7}$$

где текущая награда в момент времени t определяется как $r^{<t>}$. В теории обучения с подкреплением утверждение в уравнении (3.7) известно как эпизод обучения с конечным горизонтом длины последовательности k [13].

Цель обучения с подкреплением состоит в том, чтобы найти желаемую политику траектории, которая максимизирует связанное с ней совокупное будущее вознаграждение. Определим оптимальную функцию поощрения $Q^*(\cdot,\cdot)$, которая оценивает максимальное будущее подкрепляющее вознаграждение при начальном состоянии $\boldsymbol{s}^{<t>}$ и выполнении действий $[a^{<t>}, \dots, a^{<t+k>}]$:

$$Q^*(\mathbf{s}, \mathrm{a}) = \max_{\pi} \mathbb{E}(R^{<t^*>} | \boldsymbol{s}^{<t^*>} = \boldsymbol{s}, a^{<t^*>} = a, \pi), \tag{3.8}$$

где π — политика действий, рассматриваемая как функция плотности вероятности для набора возможных действий, которые могут иметь место в данном состоянии. Функция оптимального поощрения $Q^*(\cdot,\cdot)$ отображает заданное состояние в оптимальную политику действий беспилотника в любом другом состоянии:

$$\forall \boldsymbol{s} \in S: \pi^*(\boldsymbol{s}) = \arg\max_{a \in A} Q^*(\mathbf{s}, \mathrm{a}). \tag{3.9}$$

Оптимальная функция поощрения $Q^*(\cdot,\cdot)$ удовлетворяет уравнению оптимальности Беллмана (3.10), которое является рекурсивной формулировкой уравнения (3.8):

$$\begin{aligned} Q^*(\mathbf{s}, \mathrm{a}) &= \sum_{\boldsymbol{s}} T_{s,a}^{\boldsymbol{s}'}(R_{s,a}^{\boldsymbol{s}'} + \gamma \cdot \max_{a'} Q^*(\boldsymbol{s}', a')) \\ &= \mathbb{E}_{a'}(R_{s,a}^{\boldsymbol{s}'} + \gamma \cdot \max_{a'} Q^*(\boldsymbol{s}', a')), \end{aligned} \tag{3.10}$$

где $\boldsymbol{s}'$ — возможное состояние, достигнутое после $\boldsymbol{s} = \boldsymbol{s}^{<t>}$, а a' — соответствующая политика действия. Алгоритм выбора политики был представлен в [14] на основе доказательства того, что уравнение Беллмана (3.10) является сжимающим отображением, если его переписать в форме оператора ν:

$$\forall Q \in \mathcal{Q}: \lim_{n \to \infty} \nu^{(n)}(Q) = Q^*. \tag{3.11}$$

Однако описанный выше стандартный метод обучения с подкреплением неприменим в многомерных пространствах состояний. В приложениях для автономного вождения пространство наблюдения в основном состоит из информации, поступающей из систем восприятия дорожного окружения, таких как радары, сенсоры, датчики и т. д. Вместо традиционного подхода нелинейная параметризация Q^* может быть закодирована в слоях глубокой нейронной сети. В литературе, посвящённой обучению с подкреплением, такой нелинейный

аппроксиматор называется Deep Q-Network (DQN) [15] и используется для оценки приближенной функции оптимального поощрения:

$$Q(\boldsymbol{s}^{<t>}, a^{<t>}, \Theta) \approx Q^*(\boldsymbol{s}^{<t>}, a^{<t>}), \tag{3.12}$$

где Θ представляет набор обучающих параметров Deep Q-Network.

Принимая во внимание уравнение оптимальности Беллмана (3.10), можно обучить глубокую Q-сеть методом обучения с подкреплением за счёт минимизации среднеквадратичной ошибки. Оптимальное ожидаемое значение Q может быть оценено в рамках обучающей итерации i на основе набора эталонных параметров Θ_i^*, рассчитанных на предыдущей итерации i':

$$y = R_{s,a}^{\boldsymbol{s}'} + \gamma \cdot \max_{a'} Q(\boldsymbol{s}', a', \Theta_i^*), \tag{3.13}$$

где $\Theta_i^* = \Theta_{i'}$. Новые предполагаемые параметры сети на шаге обучения i оцениваются с использованием следующей функции квадрата ошибки:

$$\nabla J_{\Theta_i^*} = \min_{\Theta_i} \mathbb{E}_{\boldsymbol{s},y,r,\boldsymbol{s}'}[(y - Q(\boldsymbol{s}, a, \Theta_i))^2], \tag{3.14}$$

где $r = R_{s,a}^{\boldsymbol{s}'}$. На основании уравнения (3.14) функция оценки максимального правдоподобия из уравнения (3.6) может применяться для вычисления весов глубокой Q-сети. Градиент аппроксимируется случайными выборками и алгоритмом обратного распространения, использующим для обучения стохастический градиентный спуск:

$$\nabla_{\Theta_i} = \mathbb{E}_{\boldsymbol{s},a,r,\boldsymbol{s}'}[(y - Q(\boldsymbol{s}, a, \Theta_i))\nabla_{\Theta_i}(Q(\boldsymbol{s}, a, \Theta_i))]. \tag{3.15}$$

Стоит отметить, что за последнее время к исходному алгоритму (DQN) [15] добавился ряд комбинированных улучшений, которые значительно расширили функциональность и применение обучения с подкреплением, среди которых:

- устранение предвзятости переоценки и отделение выбора действия от его оценки;
- приоритизация воспроизведения данных, в которых есть важная для обучения информация;
- увеличение производительности нейронных архитектур за счёт внедрения дуэльных сетей;
- повышение скорости тренировки за счёт многоступенчатости обучения;
- улучшение игнорирования бесполезных выходных данных за счёт шумовой фильтрации и разведки с условными состояниями.

Алгоритм DQN [15] был разработан для принятия решений в дискретных пространствах. В случае с беспилотниками, наиболее распространенная стратегия глубокого обучения с подкреплением для непрерывного управления движением автомобиля основана на дискретном вождении: действия будут преобразованы в дискретные команды, такие как поворот, ускорение, замедление и т. п.

Основной проблемой здесь является сама тренировка беспилотника: транспортное средство должно исследовать дорожное окружение, как правило, путем обучения на столкновениях в симуляциях. Такие системы, обученные исключительно на смоделированных данных, зачастую изучают предвзятую версию условий вождения. Решением здесь является использование методов имитационного обучения, таких как обучение с обратным подкреплением или поведенческое клонирование на демонстрациях вождения человека без необходимости изучения небезопасных действий.

3.6 Заключение

Беспилотные автомобили — это сложные высокоуровневые системы, которые должны безопасно доставлять пассажиров или грузы из пункта отправления в пункт назначения.

За последнее десятилетие в технологии автономных транспортных средств произошел значительный прогресс, особенно благодаря достижениям в области искусственного интеллекта и глубокого обучения. Современные методологии искусственного интеллекта в настоящее время либо используются, либо учитываются при разработке различных компонентов для беспилотных автомобилей. Подходы к глубокому обучению повлияли не только на дизайн традиционных конвейеров восприятия-планирования-действия, но также сделали возможными системы обучения, способные напрямую сопоставлять информацию, поступающую из систем восприятия дорожного окружения, с управляющими командами, передаваемыми контроллерам вождения беспилотного транспортного средства.

В этой главе был представлен обзор технологий принятия поведенческих решений на основе наиболее распространённых на сегодняшний день методологий глубокого обучения и искусственного интеллекта, используемых в автономном вождении:

- свёрточных нейронных сетей (CNN);
- рекуррентных нейронных сетей (RNN);
- глубокого обучения с подкреплением (DRL);
- нереволюционных алгоритмов.

ГЛАВА 4
НЕЙРОЭВОЛЮЦИЯ В 2D-СИМУЛЯЦИЯХ ДОРОЖНОГО ОКРУЖЕНИЯ

4.1 Нейроэволюция нарастающих топологий

Нейроэволюция, то есть развитие искусственных нейронных сетей с помощью генетических алгоритмов, оказалась очень эффективной в задачах обучения с подкреплением. В этой главе будут рассмотрены практические применения нейроэволюции на примере метода NeuroEvolution of Augmenting Topologies (NEAT) [16] к беспилотным транспортным средствам в генерируемых 2D-симуляциях дорожного окружения: круговой магистрали с непрерывным движением и перпендикулярного одноуровневого паркинга.

NEAT [16] — эволюционный алгоритм, позволяющий использовать генетические алгоритмы для определения лучшей и минимально необходимой топологии нейронной сети, которая представляет собой ориентированный граф.

Нейроны сети могут быть трех видов: входные (сенсорные), скрытые (анализирующие), и выходные (разрешающие). Нейронные связи между нейронами содержат в себе информацию о своём состоянии (активном и неактивном), своих связях с другими нейронами, соответствующие веса и порядковый номер.

Наборы нейронов и нейронных связей составляют генотип (см. Рис. 4.1), с помощью которого оперируют генетическим алгоритмом путём мутаций и скрещиваний. Например, на рисунке (4.1) третий ген общего генотипа отключен, поэтому задаваемая им связь между вторым и пятым узлами не выражена в фенотипе.

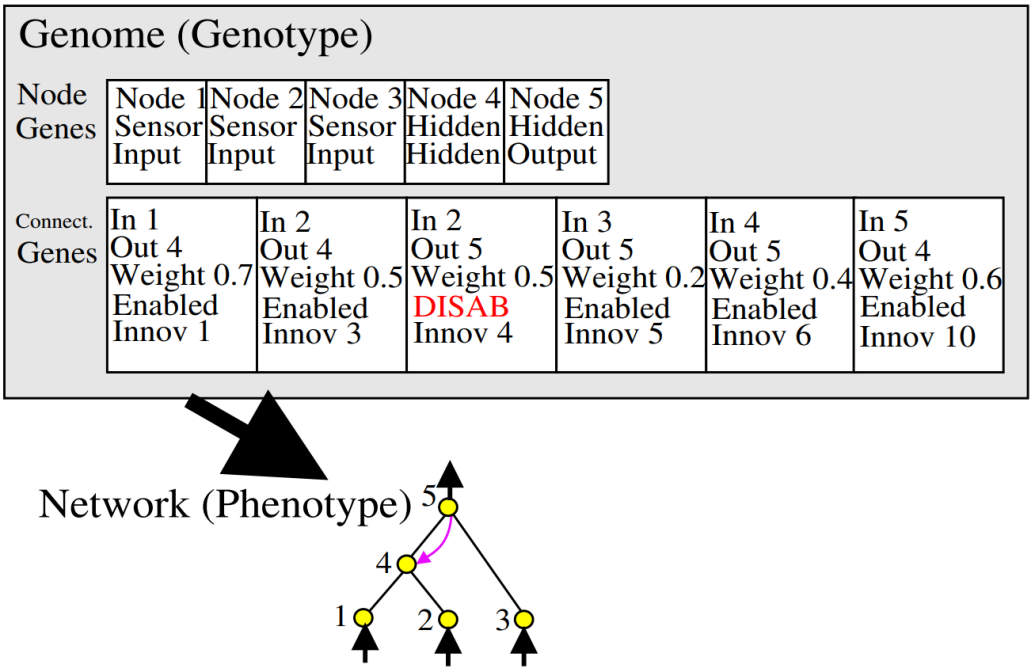

Рисунок 4.1 Пример сопоставления генотипа с фенотипом

Каждая нейронная связь обладает своим уникальным временным идентификатором (историческим маркером), который позволяет упростить скрещивание, а также отследить как возраст самого генома, так и происходившие в нём мутации в любой момент эволюции.

Мутации в эволюции генотипов бывают двух видов (см. Рис. 4.2):

- Добавление нейронных связей к существующим нейронам (4.2a);
- Создание нового нейрона на месте существующей нейронной связи путём разделения старой нейронной связи на две новые (4.2b).

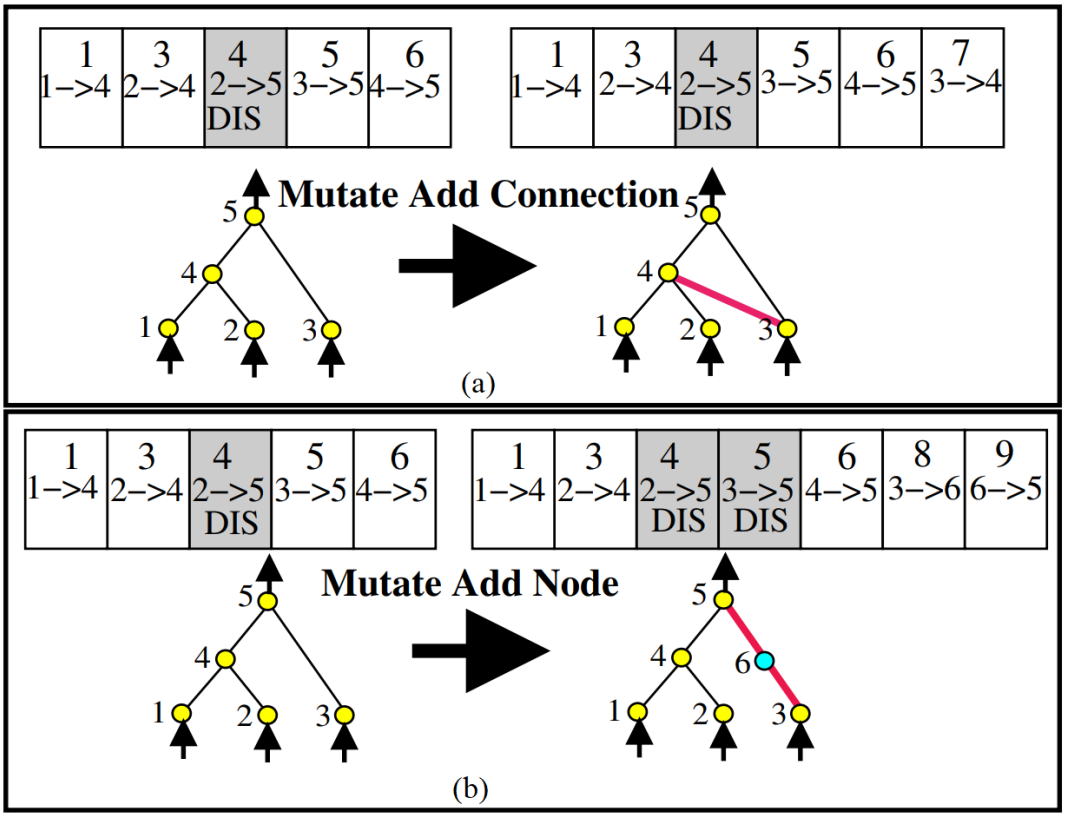

Рисунок 4.2 Два типа генетических мутаций в NEAT

При любых мутациях новым связям будет присвоен уникальный исторический маркер, который также будет использоваться при скрещивании.

Само скрещивание происходит за счёт смешения генотипов: в ген потомка вносятся уникальные исторические маркеры обоих родителей. Например, если нейронная связь неактивна у одного из родителей, то у потомка с определённой долей вероятности она также будет неактивна, а если связь неактивна у обоих родителей, то у неё также есть определённый шанс мутировать и стать активной у потомка (см. Рис. 4.3).

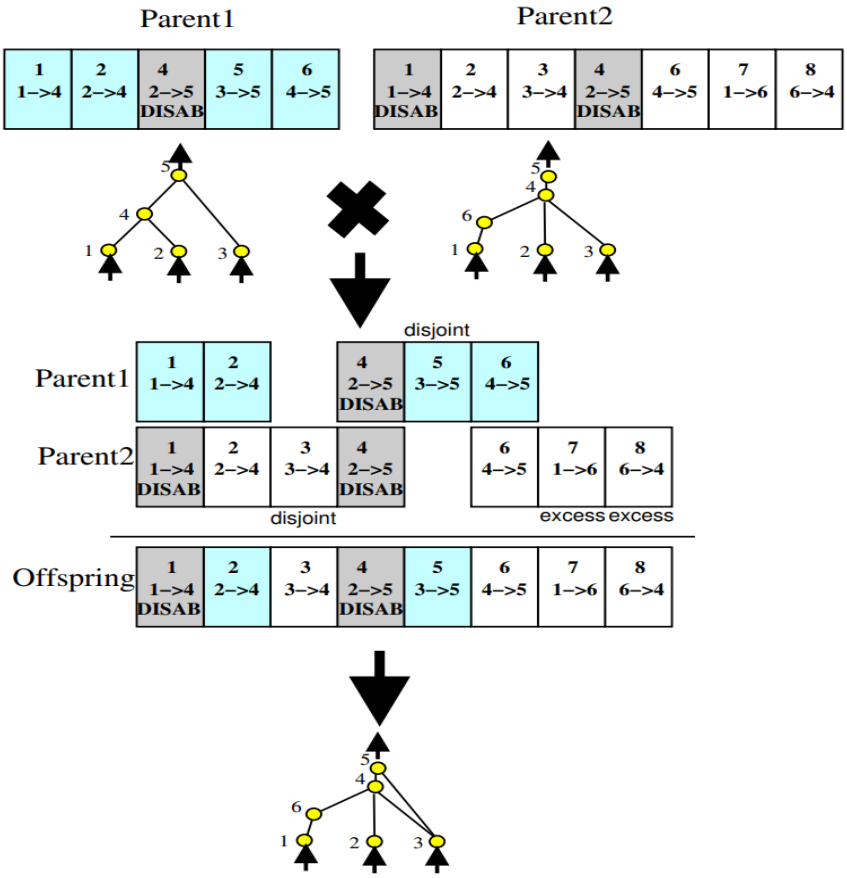

Рисунок 4.3 Пример скрещивания геномов в NEAT

При запуске эволюции генетического алгоритма создаётся начальная популяция особей, каждая из которых состоит только из пронумерованных входных и выходных нейронов, связанных между собой по принципу классического однослойного персептрона.

На протяжении нескольких поколений к каждой особи популяции применяется фитнесс-функция, ранжирующая всю популяцию и определяющая, насколько особи приспособлены для выполнения заданных требований. Чем более приспособленной является особь, тем более вероятно её продолжение рода в следующих поколениях эволюции, однако даже не очень приспособленные особи имеют небольшой статистический шанс не выродиться, что обеспечивает избежание достижения локальных экстремумов — не самого оптимального эволюционирования популяции.

Далее алгоритм по определённому вероятностному принципу отбирает особей для двуполого скрещивания геномов, посредством которого генерируется полностью новая популяция, и происходит их мутация, при которой случайным образом добавляются новые связи или нейроны, а значения весов с определённой вероятностью корректируются.

В итоге аналогичные действия повторяются в следующих поколениях до тех пор, пока популяция не достигнет оптимальных показателей приспособленности в поставленных условиях обучения.

4.2 Проектирование кинематической 2D-модели автомобиля

Кинематическая 2D-модель транспортного средства будет представлять собой полноприводный четырёхколёсный автомобиль с рулевым управлением Аккермана для наиболее точного отражения динамики управления реального автомобиля в симуляции. Полная реализация рассматриваемой кинематической модели автомобиля на языке Python представлена в репозитории автора [1].

На рисунке (4.4) показаны основные характеристики и обозначения моделируемого транспортного средства в 2D.

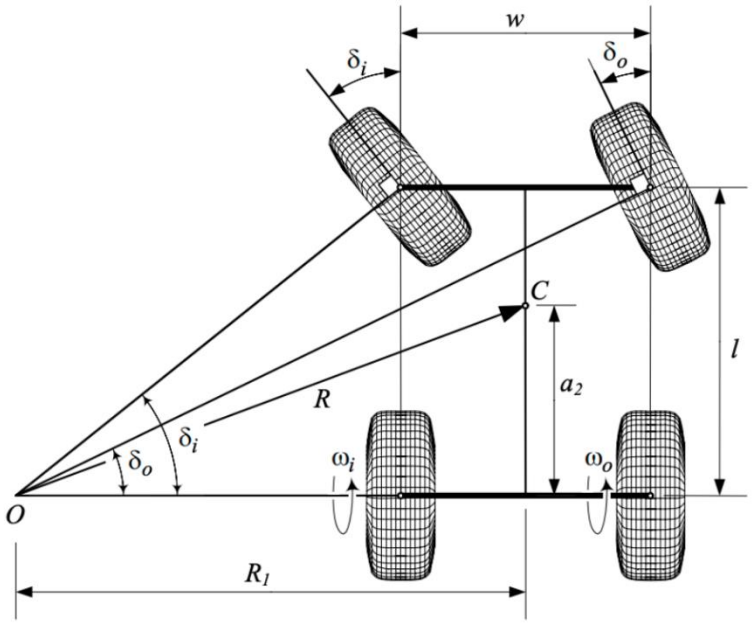

Рисунок 4.4 Кинематическая 2D-модель автомобиля

Модель транспортного средства для воспроизведения наиболее реальных свойств вождения в симуляции будет содержать следующие параметры:

- Экземпляр 2D-спрайта — для отображения автомобиля в симуляции;
- Длина и ширина шасси — для инициализации размеров автомобиля;
- Позиция центра кузова и угол его смещения от вертикали — для локализации автомобиля в симуляции;
- Скорость и ускорение — для регулировки быстроты перемещения автомобиля в анимации;
- Угол поворота колёс — для регулировки радиуса поворота автомобиля в анимации;
- Свободное и тормозное ускорение — для регулировки скорости замедления автомобиля в анимации;
- Максимальная скорость и ускорение — для контроля верхних границ быстроты перемещения автомобиля в анимации;
- Максимальное значение угла поворота колёс — для контроля верхних границ радиуса поворота автомобиля в анимации;
- Положения и статусы точек коллизий — для контроля столкновений автомобиля с дорожным окружением;
- Положения и дальности радаров — для обнаружения объектов дорожного окружения на пути автомобиля;
- Счётчик очков — для контроля качества вождения автомобиля алгоритмом принятия решений.

Предположим, что два передних колеса будут иметь идеальное сцепление с дорожной поверхностью и оставаться взаимно параллельными, следовательно, угол поворота внешнего и внутреннего колеса задаётся соответственно:

$$\delta_i = \delta_o = \delta. \tag{4.1}$$

Центр масс транспортного средства C располагается в физическом центре транспортного средства, поэтому половина длины шасси вычисляется как половина всей длины кузова:

$$a_2 = \frac{l}{2}. \tag{4.2}$$

Без ограничения общности предположим, что два задних колеса вращаются с одинаковой скоростью, тогда угловые скорости внешнего и внутреннего колеса при повороте будут совпадать:

$$\omega_i = \omega_o = \omega. \tag{4.3}$$

Радиус поворота вокруг центра вращения O вычисляется как:

$$R = \sqrt{a_2^2 + l^2 ctg^2(\delta)}. \tag{4.4}$$

С помощью элементарной тригонометрии определим новый угол α, являющийся углом между R_1 и R (см. Рис. 4.5):

$$\alpha = sin^{-1}\left(\frac{a_2}{R}\right). \tag{4.5}$$

Вычислим горизонтальное расстояние R_1 между центром масс автомобиля C и центром его вращения O при повороте:

$$R_1 = R\cos(\alpha). \tag{4.6}$$

Тогда угловая скорость транспортного средства определяется как:

$$\varepsilon = \frac{R\omega}{R_1 + \frac{w}{2}}. \tag{4.7}$$

В момент поворота автомобиль описывает дугу с радиусом R и углом $d\theta$, являющимся функцией угловой скорости ε и времени движения dt (см. Рис. 4.5):

$$d\theta = \varepsilon \cdot dt. \tag{4.8}$$

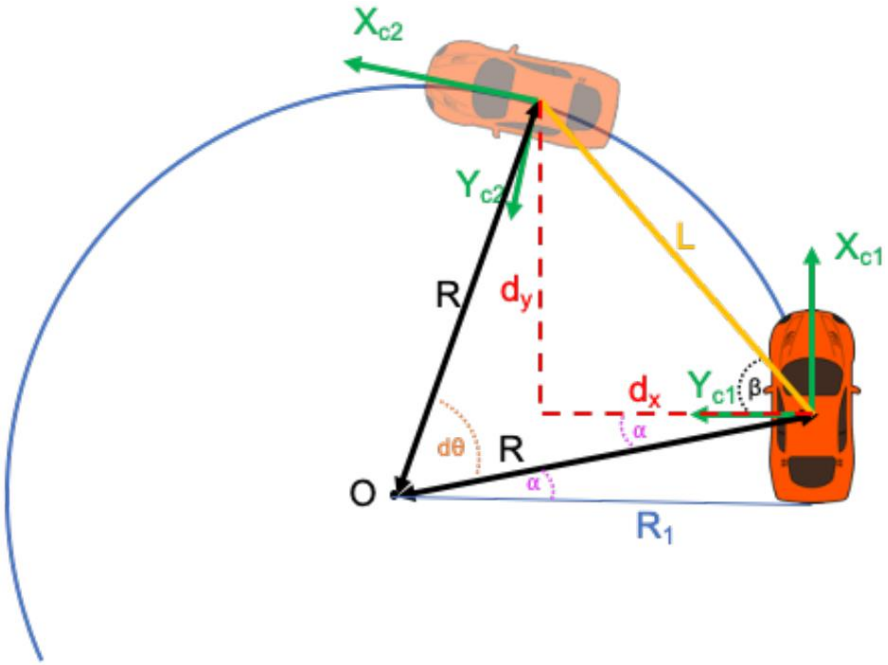

Рисунок 4.5 Диаграмма траектории движения автомобиля при повороте

На рисунке (4.5) показана траектория движения автомобиля, его ключевые составляющие скорости и перемещения при повороте, где L — кратчайшее евклидово расстояние между начальной и конечной позициями:

$$L = 2Rsin(\frac{d\theta}{2}). \tag{4.9}$$

Также необходимо найти компоненты d_x и d_y смещения с учётом перемещения центра вращения вместе с автомобилем. Из треугольника $\triangle RLR$:

$$2(\beta + \alpha) + d\theta = 180° \Longrightarrow \beta = \frac{180° - d\theta}{2} - \alpha. \quad (4.10)$$

Тогда компоненты смещения автомобиля вычисляются как:

$$\begin{cases} d_x = Lsin(\beta) \\ d_y = Lcos(\beta) \end{cases}. \quad (4.11)$$

С учётом движения центра вращения O вместе с автомобилем и угла его поворота θ получим финальные горизонтальные и вертикальные составляющие смещения в глобальной системе координат симуляции:

$$\begin{cases} O_x = vcos(\theta)dt \\ O_y = vsin(\theta)dt \end{cases}. \quad (4.12)$$

$$\begin{cases} x = d_x\cos(\theta) + d_y\cos(\theta) + O_x \\ y = d_ysin(\theta) + d_xsin(\theta) + O_y \end{cases}. \quad (4.13)$$

$$\varphi = \theta + d\theta. \quad (4.14)$$

Ускорение автомобиля определяется как первая производная скорости по времени, следовательно, скорость можно вычислить следующим образом:

$$a = \frac{dv}{dt} \Longrightarrow dv = adt. \quad (4.15)$$

Для восприятия окружающей среды и взаимодействия с ней в симуляции интегрируем в модель автомобиля набор сенсоров и датчиков. На рисунке (4.6) на углах кузова голубым цветом обозначены датчики коллизий, реагирующие на столкновения с дорожными препятствиями, а по радиусу кузова оранжевым цветом — система лидаров, сканирующая дорожное окружение.

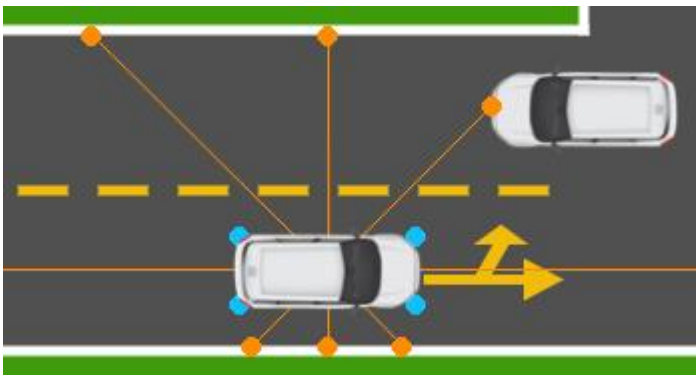

Рисунок 4.6 Датчики коллизий и система лидаров модели автомобиля

Для простоты реализации восприятие окружающей среды сенсорами и датчиками в симуляции будет производиться посредством проверки RGB-составляющих цвета пикселей окружения. Например, система лидаров, воспринимая в радиусе своего обзора цвета пикселей, отличные от цветов дорожной разметки, может распознать близость автомобиля к обочине справа и движение других транспортных средств на встречной полосе (см. Рис. 4.6).

Пусть φ — угол поворота кузова автомобиля, x и y — координаты его центра, а x^c_{old} и y^c_{old} — старые координаты датчиков коллизий на его углах, тогда новые координаты датчиков при движении автомобиля вычисляются как:

$$\begin{cases} x^c_{new} = x + (x^c_{old} - x)\cos(\varphi) - (y^c_{old} - y)\sin(\varphi) \\ y^c_{new} = y + (x^c_{old} - x)\sin(\varphi) + (y^c_{old} - y)\cos(\varphi) \end{cases}. \quad (4.16)$$

В аналогичных обозначениях при радиусе обзора r системы лидаров, выпускающей лучи с шагом Δr вдаль с радиальным отступом $\Delta\varphi$ от угла поворота кузова автомобиля φ, новые координаты лучей вычисляются как:

$$\begin{cases} x^l_{new} = x + \min(\Delta r, r)\cos(90° - \varphi - \Delta\varphi) \\ y^l_{new} = y + \min(\Delta r, r)\sin(90° - \varphi - \Delta\varphi) \end{cases}. \quad (4.17)$$

В результате для перемещения автомобиля в симуляции необходимо: обновить параметры движения (ускорение, скорость, координаты, угол поворота колёс и кузова), параметры датчиков коллизий и системы лидаров, а также выполнить их валидацию и отрисовать по ним новый кадр симуляции.

4.3 Проектирование 2D-симуляции дорожного окружения

Для обучения модели автомобиля автоматическому вождению и парковке спроектируем генерируемые 2D-симуляции дорожного окружения: круговую магистраль с непрерывным движением и перпендикулярный одноуровневый паркинг. Полная реализация рассматриваемых симуляций на языке Python представлена в репозитории автора [1].

4.3.1 Генерация автомагистрали

Автомагистраль будет генерироваться в виде случайно поляризированной замкнутой B-сплайн-кривой и содержать следующие параметры:

- Допустимый диапазон цветов пикселей дороги и её разметки — для ориентации сенсорных систем автомобиля в окружающей среде;
- Позиция геометрического центра трека — для корректного центрирования дорог в анимации;
- Верхняя граница протяжённости трека — для корректного масштабирования дорог в анимации;
- Параметр сложности трека — для контроля количества и крутизны поворотов дорог в анимации;
- Параметр ширины трека — для контроля внутренних границ дорожной полосы в анимации;
- Начальная позиция и угол автомобиля — для контроля исходной позиции и ориентации транспортных средств при запуске анимации.

При старте симуляции дорожного окружения будет генерироваться случайное распределение $\{\mathcal{R} \in \mathbb{R}[a,b]: a,b \in \mathbb{N}, |\mathcal{R}| = \lambda\}$ размерности λ, контролируемой параметром сложности трека, на отрезке $\mathbb{R}[a,b]$ с границами $a,b \in \mathbb{N}$, контролируемыми параметром протяжённости трека, а также последовательность чисел $\{\Psi \in \mathbb{R}[0,2\pi]: |\Psi| = \lambda\}$, имеющая аналогичную

размерность λ. Каркасом $К[\mathcal{R}, \Psi, c_0]$ дорожной магистрали, центрированной в точке $c_0 = (x_0, y_0)$, будет выступать поляризация окружности (4.18), в которой в качестве полярных радиусов будут выступать элементы $\rho_i \in \mathcal{R}$, а в качестве полярных углов — элементы $\gamma_i \in \Psi$.

$$\begin{cases} x_k = x_0 + \rho_i \cos(\gamma_i) \\ y_k = y_0 + \rho_i \sin(\gamma_i) \end{cases}. \tag{4.18}$$

Замкнутая дорожная магистраль будет конструироваться по точкам $(x_k, y_k) \in К[\mathcal{R}, \Psi, c_0]$ полученного каркаса с помощью B-сплайн-интерполяции.

Примеры итоговых круговых магистралей с непрерывным движением различной сложности и протяжённости проиллюстрированы на рисунке (4.7).

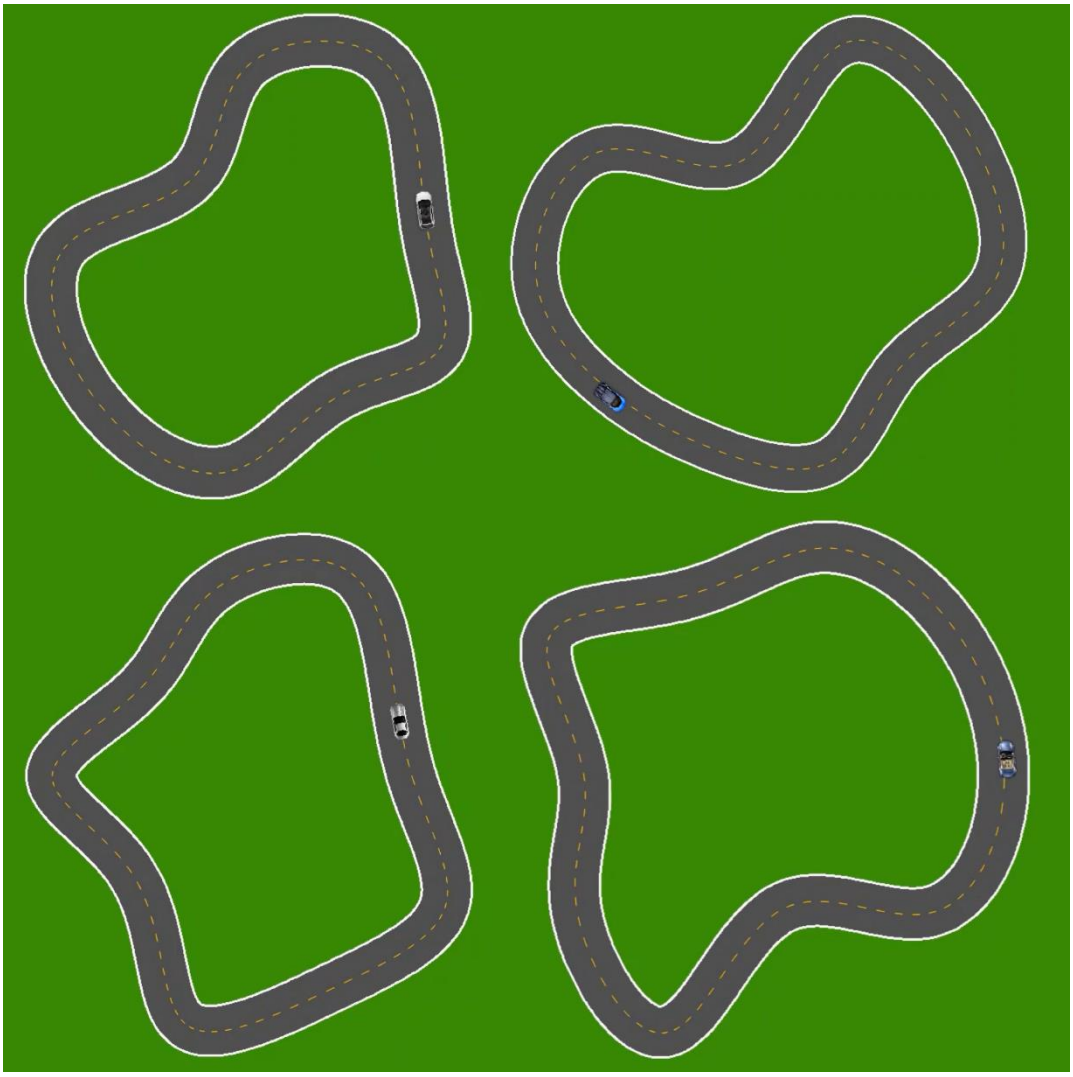

Рисунок 4.7 Круговые магистрали различной сложности и протяжённости

4.3.2 Генерация паркинга

Парковка будет генерироваться на заранее заготовленной 2D-сетке с заранее размеченными и пронумерованными парковочными местами (см. Рис. 4.8), а также содержать следующие параметры:

- Допустимый диапазон цветов пикселей парковки и её разметки — для ориентации сенсорных систем автомобиля в окружающей среде;
- Таблица соответствий позиций парковочных мест с их номерами — для локализации парковочного места по его номеру;
- Список спрайтов автомобилей — для заполнения случайных парковочных мест транспортными средствами;
- Список занятых парковочных мест — для контроля корректности заполнения парковочных;
- Номер и позиция целевого парковочного места — для локализации конечного пункта назначения при автоматической парковке;
- Начальная позиция и угол поворота транспортного средства — для контроля исходной позиции и ориентации автомобиля при въезде на парковку.

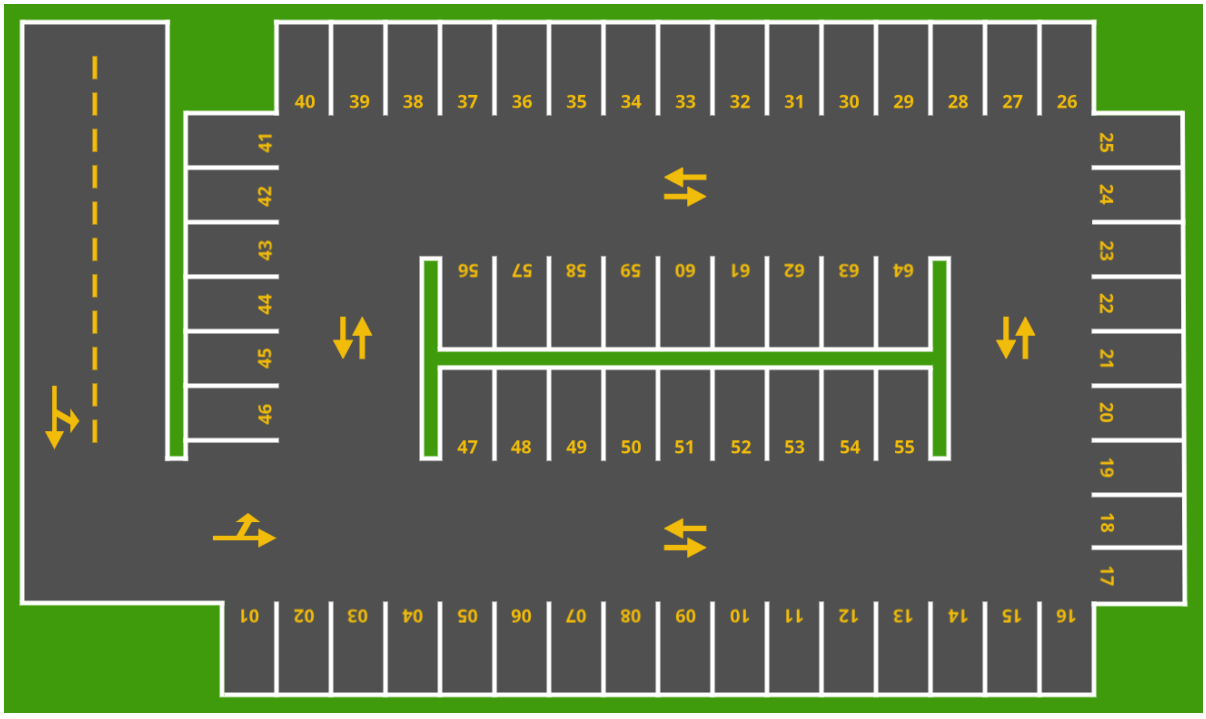

Рисунок 4.8 Шаблон перпендикулярного одноуровневого паркинга

При старте симуляции дорожного окружения на случайных парковочных местах будут генерироваться автомобили согласно заданным в таблице соответствий позициям и ориентациям, а также из незанятых парковочных мест будет случайным образом выбираться и подсвечиваться целевое место стоянки, расположение которого будет передаваться системам восприятия автомобиля в виде числовых координат и евклидова расстояния, обозначающего дистанцию от текущего положения автомобиля до места его парковки.

Пример случайно заполненного перпендикулярного паркинга с выбранным целевым местом №13 проиллюстрирован на рисунке (4.9).

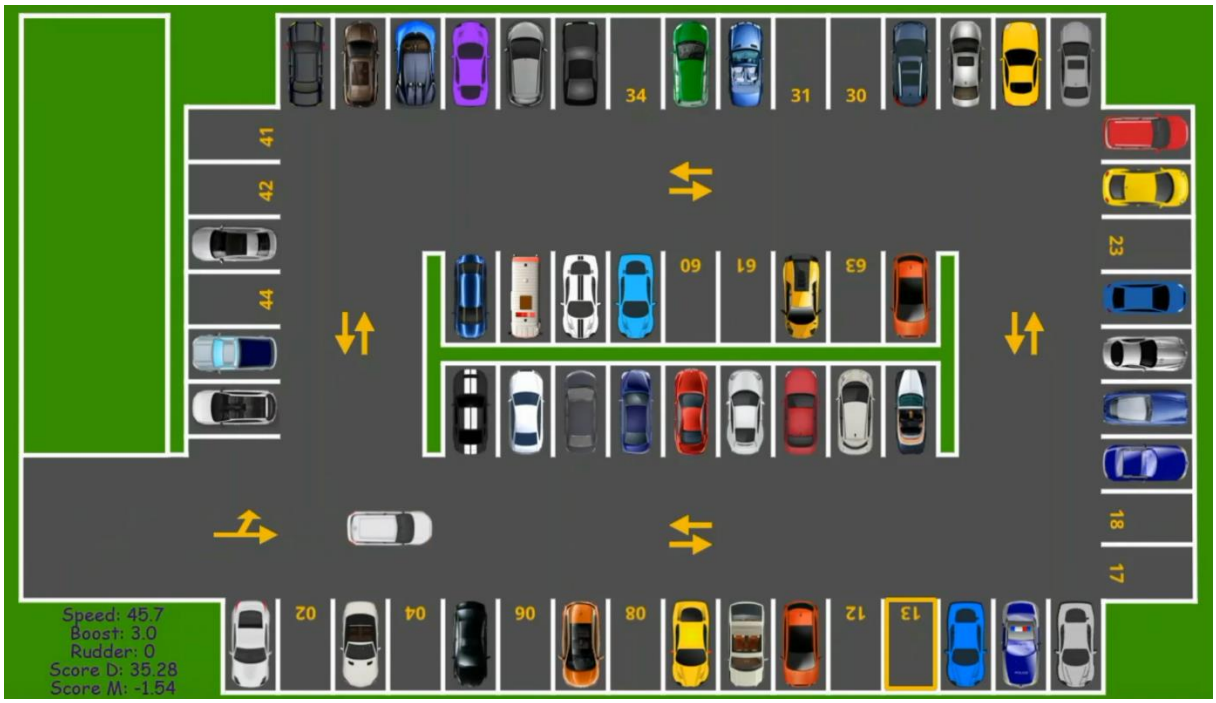

Рисунок 4.9 Заполненный перпендикулярный одноуровневый паркинг

4.4 Интеграция NEAT-алгоритма в 2D-симуляцию

После того, как реализованы симуляции дорожных окружений и заданы законы движения и восприятия окружающей среды кинематической моделью транспортного средства, для обучения самостоятельному движению в автомобиль необходимо интегрировать нереволюционный алгоритм, который в

течение своего эволюционирования на основе данных из датчиков и сенсоров восприятия научится корректному вождению. В качестве вышеописанного нереволюционного алгоритма в pygame-симуляции будет использоваться библиотека NEAT [17] языка Python, которая позволит легко настроить такие параметры обучения, как: критерии фитнесс-функций, критерии исследования, критерии стагнации и элитарности, методы обучения, размеры популяции, правила мутаций и скрещиваний, архитектуры нейронных сетей и многое другое.

4.4.1 Обучение беспилотному вождению

Нейронное эволюционирование кинематических моделей автомобилей в 2D-симуляциях случайно сгенерированных круговых магистралей с непрерывным движением производится по следующим правилам:

- Популяция моделей инициализируется и развивается в соответствии с заданными генетическими конфигурациями [1];
- На входы нейронной сети (см. Рис. 4.10) подаются нормированные расстояния до препятствий от каждого из восьми радаров, расположенных по всему радиусу кузова автомобиля, и текущая скорость автомобиля;
- Выходами нейронной сети (см. Рис. 4.10) являются параметры направления движения (вперёд/назад/свободное) и поворота рулевой колодки (влево/вправо/по центру) автомобиля;
- Каждая модель поколения награждается за положительно ориентированный вектор скорости (за продолжительность движения) и штрафуется за коллизии (столкновение с препятствиями), пересечения дорожной разметки и время бездействия.

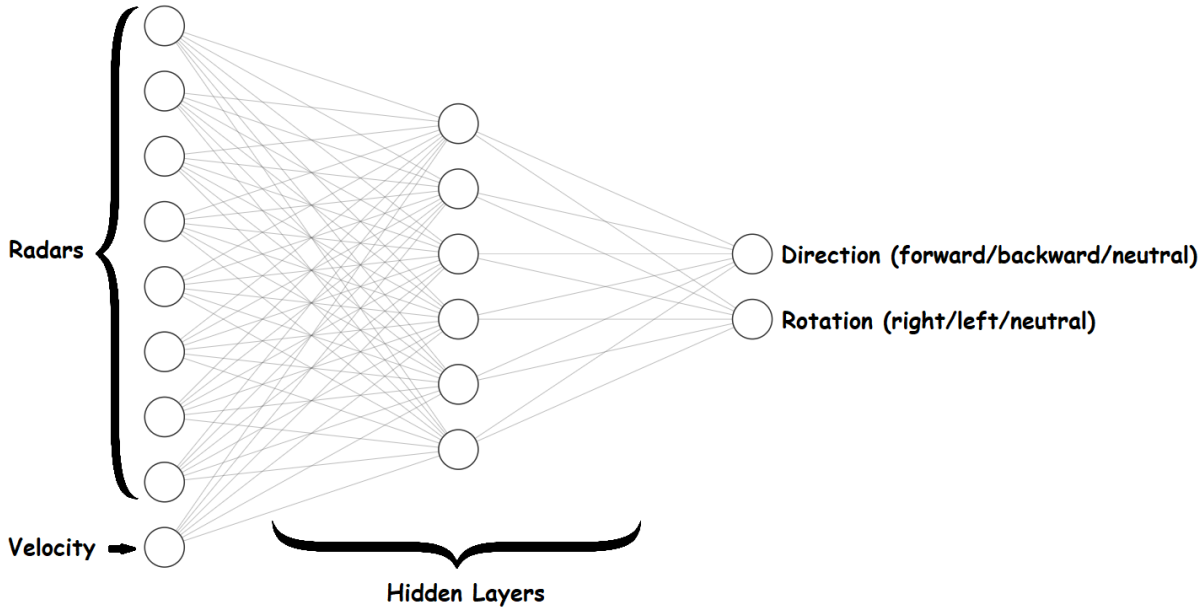

Рисунок 4.10 Архитектура нейронной сети для обучения вождению

Инициализация нового поколения моделей автомобилей и соответствующих им геномов нейронных сетей в симуляции дорожного окружения выглядит следующим образом:

```
1.  def _init_new_generation(self, genomes, config):
2.      """Initializes new generation of networks and cars according to genomes"""
3.      self.nets, self.cars = [], []
4.      self.best_score = 0
5.      self.generation += 1
6.
7.      for _, gen in genomes:
8.          gen.fitness = 0
9.          net = neat.nn.FeedForwardNetwork.create(gen, config)
10.         car = Car(spawn_position=self.highway.start_position,
    spawn_angle=self.highway.start_angle, scale=0.5)
11.         self.nets.append(net)
12.         self.cars.append(car)
```

Эволюционирование геномов и управление перемещением моделей транспортных средств по автомагистрали в терминологии NEAT описывается как:

```
1.  self.cars_left = 0
2.  for net, car, gen in zip(self.nets, self.cars, genomes):
3.      # get movement params from network
4.      output = net.activate(np.append(car.radars_data, car.velocity.x / car.max_velocity))
5.
6.      # movement params mapping
7.      direction, rotation = [0 if -0.33 < out < 0.33 else np.sign(out) for out in output]
8.      movement_params = {"direction": direction, "rotation": rotation}
9.
10.     # move a car
11.     t = self.clock.get_time() * 0.01
12.     car.move(movement_params, t, self.screen, self.highway)
```

```
13.
14.     # update car fitness
15.     self.best_score = max(self.best_score, car.score)
16.     gen[1].fitness = car.score
17.     self.cars_left += 1 if car.is_alive else 0
```

Полная реализация класса симуляции для обучения беспилотного автомобиля вождению посредством NEAT-алгоритма на случайно сгенерированных круговых автомагистралях с непрерывным движением представлена в репозитории автора [1].

4.4.2 Обучение беспилотной парковке

Нейронное эволюционирование кинематических моделей автомобилей в 2D-симуляциях случайно сгенерированных перпендикулярных одноуровневых паркингов производится по следующим правилам:

- Популяция моделей инициализируется и развивается в соответствии с заданными генетическими конфигурациями [1];
- На входы нейронной сети (см. Рис. 4.11) подаются нормированные расстояния до препятствий от каждого из восьми радаров, расположенных по всему периметру кузова автомобиля, навигационные параметры, необходимые для продвижения к целевому месту парковки, и текущее евклидово расстояние от автомобиля до целевого места парковки;
- Выходами нейронной сети (см. Рис. 4.11) являются параметры направления движения (вперёд/назад/свободное) и поворота рулевой колодки (влево/вправо/по центру) автомобиля;
- Каждая модель поколения награждается за близость к целевому месту парковки (критерий удачной парковки — совпадение периметра кузова с границами парковочного места) и штрафуется за коллизии (столкновение с препятствиями), пересечения дорожной разметки и время бездействия.

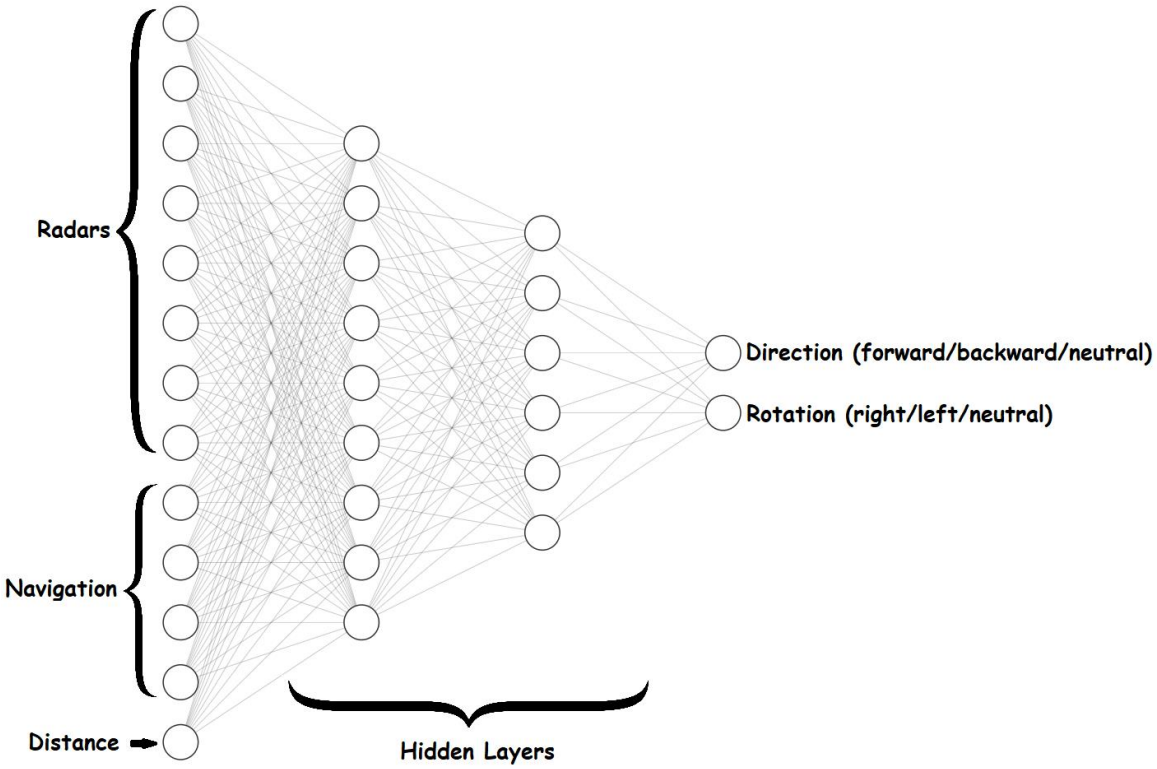

Рисунок 4.11 Архитектура нейронной сети для обучения парковке

Инициализация нового поколения моделей автомобилей и соответствующих им геномов нейронных сетей в симуляции дорожного окружения выглядит следующим образом:

```
def _init_new_generation(self, genomes, config):
    """Initializes new generation of networks and cars according to genomes"""
    self.nets, self.cars = [], []
    self.best_score = 0
    self.generation += 1

    for _, gen in genomes:
        gen.fitness = 0
        net = neat.nn.FeedForwardNetwork.create(gen, config)
        car = Car(self.parking.start_position, self.parking.start_angle)
        self.nets.append(net)
        self.cars.append(car)
```

Эволюционирование геномов и управление перемещением моделей транспортных средств по парковке в терминологии NEAT описывается как:

```
self.cars_left = 0
for net, car, gen in zip(self.nets, self.cars, genomes):
    # get movement params from network
    inputs = np.concatenate((car.radars_data, car.navigation))
    outputs = net.activate(inputs)

    # movement params mapping
    direction, rotation = [0 if -0.33 < out < 0.33 else np.sign(out) for out in outputs]
    movement_params = {"direction": direction, "rotation": rotation}
```

```
10.
11.     # move a car
12.     t = self.clock.get_time() * 0.01
13.     car.move(movement_params, t, self.screen, self.parking)
14.
15.     # update car fitness
16.     self.best_score = max(self.best_score, car.score)
17.     gen[1].fitness = car.score
18.     self.cars_left += 1 if car.is_alive else 0
```

Полная реализация класса симуляции для обучения беспилотного автомобиля парковке посредством NEAT-алгоритма на случайно сгенерированных перпендикулярных одноуровневых паркингах представлена в репозитории автора [1].

4.5 Заключение

В этой главе были рассмотрены практические применения нейроэволюции на примере алгоритма NeuroEvolution of Augmenting Topologies (NEAT) [16] к беспилотным транспортным средствам в генерируемых 2D-симуляциях дорожного окружения: круговой магистрали с непрерывным движением и перпендикулярного одноуровневого паркинга.

Также были описаны и разработаны полноценные API для взаимодействия кинематических моделей автомобилей с генерируемыми симуляциями дорожного окружения для их последующей тренировки и тестирования:

- API для обучения вождению беспилотного автомобиля в симуляции

```
1.  from autopilot import Simulation
2.
3.  sim = Simulation(epochs=100, map_spread=(150, 350), map_complexity=5, time_per_map=3000)
4.  best_genome = sim.train()
5.  sim.save(best_genome)
```

- API для тестов вождения беспилотного автомобиля в симуляции

```
1.  from autopilot import Simulation
2.
3.  sim = Simulation(map_spread=(150, 350), map_complexity=5)
4.  best_genome = sim.load("checkpoints/best.pkl")
5.  sim.test(best_genome)
```

- API для обучения парковке беспилотного автомобиля в симуляции

```
from autopilot import Simulation

sim = Simulation(epochs=1000, time_per_map=500)
best_genome = sim.train()
sim.save(best_genome)
```

- API для тестов парковки беспилотного автомобиля в симуляции

```
from autopilot import Simulation

sim = Simulation()
best_genome = sim.load("checkpoints/best.pkl")
sim.test(best_genome)
```

Из результатов тренировок в различных симуляциях можно сделать вывод, что нереволюционный алгоритм является очень эффективным методом обучения с подкреплением в случае самостоятельного вождения, избегания препятствий, быстрого реагирования и адаптации к различного вида изменениям условий дорожного окружения — для полной поведенческой приспособленности к любому набору случайно сгенерированных условий окружения нереволюционному алгоритму в среднем может понадобиться около пятидесяти поколений, в зависимости от размера популяций и конфигурационных параметров генотипов и процесса эволюции. Однако в случае с самостоятельной парковкой, алгоритму для достижения приемлемых результатов требуется в десятки раз больше времени — более тысячи поколений, а также дополнительные параметры навигации по парковочной местности и локализации целевого парковочного места. На практике парковочные ассистенты больше полагаются на детерминированные парковочные алгоритмы, использующие информацию с лидаров, сенсоров и камер беспилотного автомобиля для обеспечения наиболее безопасного и точного планирования манёвров и вписывания в парковочное место, а также глубокое обучение для контроля за окружением и распознавания потенциальных дорожных препятствий.

Полная реализация вышеописанных API вместе с видео-демонстрациями результатов обучения беспилотных моделей автомобилей самостоятельному вождению и парковке представлены в репозитории автора [1].

ГЛАВА 5
ГЛУБОКОЕ ОБУЧЕНИЕ В 3D-СИМУЛЯЦИЯХ ДОРОЖНОГО ОКРУЖЕНИЯ

5.1 Поведенческое клонирование

Как уже было упомянуто в третьей главе, одной из наиболее распространенных методологий глубокого обучения, применяемых к беспилотным автомобилям, на сегодняшний день являются использование свёрточных нейронных сетей, которые сегодня широко используются для задач распознавания образов при анализе изображений. В частности, вместе со свёрточными сетями для предварительного обучения искусственного интеллекта беспилотных транспортных средств используется метод поведенческого клонирования, с помощью которого навыки вождения реальных людей, записанные на огромное множество видеорегистраторов, интерпретируются и воспроизводятся автомобилями в компьютерных симуляциях и, впоследствии, на тестовых дорожных треках. Метод поведенческого клонирования популярен и широко используется многими производителями беспилотных технологий.

В этой главе будет описано обучение модели автомобиля автономному вождению в 3D-симуляциях различных дорожных трасс с помощью клонирования поведения водителя посредством свёрточной нейронной сети.

Обучающие данные для беспилотного транспортного средства будут генерироваться напрямую в симуляторе, а затем обрабатываться и передаваться в глубокую нейронную архитектуру, которая изучит и попытается воспроизвести поведение и риторику вождения в симуляции по заранее записанным кадрам езды в различных условиях дорожного окружения.

5.2 3D-симулятор дорожного окружения

По причине недостатка вычислительных мощностей и упадка производительности обработки данных и обучения нейросетевых моделей, вместо разработки собственного движка или использования современных игровых симуляторов вождения, для практического рассмотрения глубокого обучения в 3D-симуляциях дорожного окружения было решено использовать легковесный симулятор дорожного окружения Udacity [18].

Данный симулятор предоставляет несколько отличающихся между собой окружением, сложностью и освещённостью трасс:

- Хорошо освещённые ровные дороги с местами крутыми поворотами (см. Рис 5.1а) — для тренировки и валидации беспилотника;
- Слабо освещённые холмистые дороги, с множеством лёгких и крутых поворотов (см. Рис 5.1б) — для тестирования беспилотника.

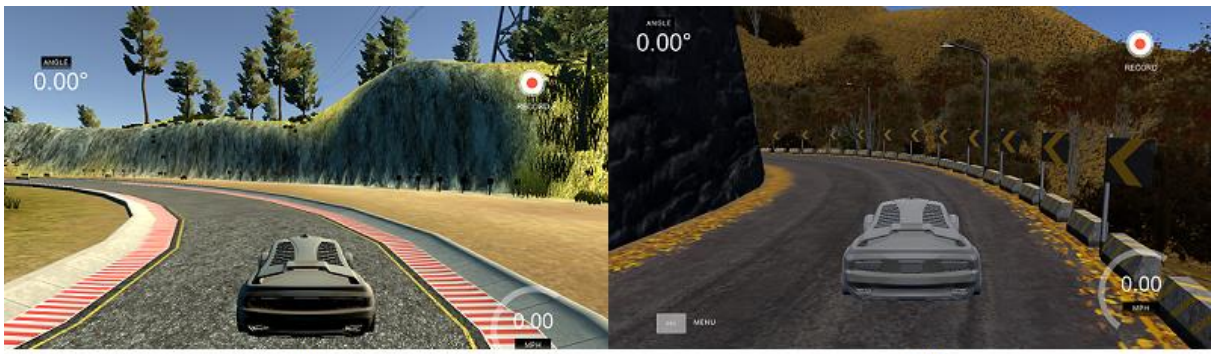

Рисунок 5.1 Udacity's Self-Driving Car Simulator [18]

Симулятор также дает возможность сохранять кадры с трех фронтальных камер автомобиля, для каждого из которых записывая соответствующие статистические данные о вождении, такие как: угол поворота руля, ускорение, торможение и скорость при движении автомобиля по трассе.

В качестве входных данных нейронной сети будут использоваться изображения дорожного окружения с трёх фронтальных камер автомобиля, а в качестве предсказания будет ожидаться угол его поворота в диапазоне $[-1,1]$.

5.3 Сбор и обработка тренировочных данных

С помощью симулятора вождения сгенерируем обучающий набор данных на базе несколькочасовой езды по сгенерированным трассам, выбранным для тренировки беспилотника, включающий как вождение в «плавном» режиме (оставаясь прямо посередине дорожной полосы) для удержания автомобиля на полосе во время крутых поворотов, так и в режиме «восстановления» (съезжая с и возвращаясь на середину дорожной полосы) для устранения предвзятости к прямолинейному вождению. На рисунке (5.2) приведены графики угла поворота рулевой колодки автомобиля в нормализованных тренировочных данных.

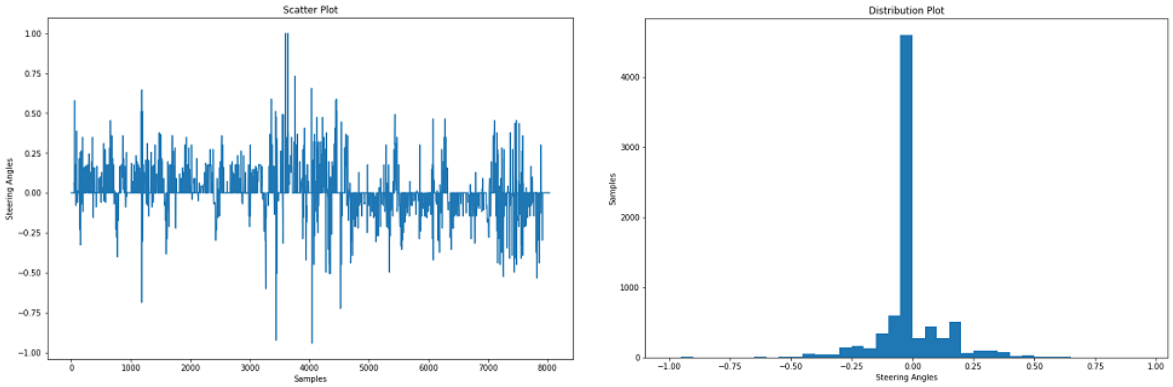

Рисунок 5.2 Распределения углов рулевой колодки в обучающих данных

После балансировки и нормализации обучающих данных было получено около восьми тысяч выборок. Для расширения набора данных были использованы следующие этапы их обработки [1]:

5.3.1 Выбор изображения от одной из трёх фронтальных камер

Симулятор предоставляет три вида фронтальных изображений дорожной полосы, а именно: центральный вид, вид слева и вид справа (см. Рис. 5.3). Добавление изображений с левой и правой фронтальных камер увеличивает исходный обучающий набор данных в три раза.

Рисунок 5.3 Изображение с левой, центральной и правой камер

Поскольку на вход нейронной сети необходимо передавать только одно изображение, его выбор из трёх возможных вариантов будет осуществляться случайным образом. Так как изображение центральной камеры соответствует записанному углу поворота рулевой колодки автомобиля, углы поворота для изображений левой и правой камер получаются путем сложения и вычитания центрального угла поворота на калиброванное значение, равное 0.25:

```
1.  cameras = ["left", "center", "right"]
2.  steering_correction = [.25, 0., -.25]
3.  camera = np.random.randint(len(cameras))
4.  image = mpimg.imread(data[cameras[camera]].values[i])
5.  angle = data.steering.values[i] + steering_correction[camera]
```

5.3.2 Горизонтальный переворот изображений

Обучающие данные могут иметь угловое смещение либо к левому, либо к правому краю дорожного трека. Следовательно, обученная сеть может также демонстрировать смещение в сторону левого или правого поворота. Чтобы избежать этого сценария, половина обучающих изображений переворачиваются по горизонтали, а их угол поворота заменяется на противоположный (см. Рис. 5.4), в результате чего тренировочная выборка увеличивается в 2 раза:

```
1.  length = x.shape[0]
2.  flip_idxs = random.sample(range(length), length // 2)
3.  x[flip_idxs] = x[flip_idxs, :, ::-1, :]
4.  y[flip_idxs] = -y[flip_idxs]
```

Рисунок 5.4 Горизонтальный переворот изображения

5.3.3 Обрезка деталей изображений

Изображения, записанные симулятором, могут содержать несущественные детали, такие как небо, горы и деревья, которые необходимо вырезать из исходных обучающих данных (см. Рис. 5.5), чтобы нейронная сеть не изучала незначительные для процедуры вождения шаблоны:

```
1.  length = image.shape[0]
2.  top = int(random.uniform(.325, .425) * length)
3.  bottom = int(random.uniform(.075, .175) * length)
4.  image = image[top:-bottom, :]
```

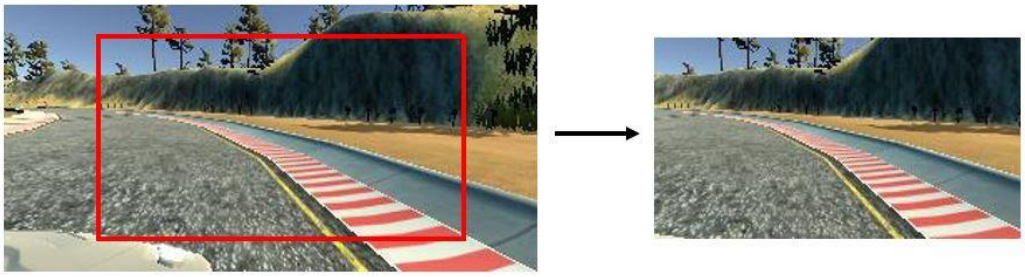

Рисунок 5.5 Обрезка необработанного изображения

5.3.4 Изменение яркости изображений

Чтобы сделать модель нейронной сети инвариантной к реальным теням на дороге, добавим случайным образом вертикальную тень путём уменьшения яркости фрагмента изображения (см. Рис. 5.6):

```
h, w = image.shape[0], image.shape[1]
x1, x2 = np.random.choice(w, 2, replace=False)
k = h / (x2 - x1)
b = -k * x1
for i in range(h):
    c = (i - b) // k
    image[i, :c, :] = (image[i, :c, :] * .5).astype(np.int32)
```

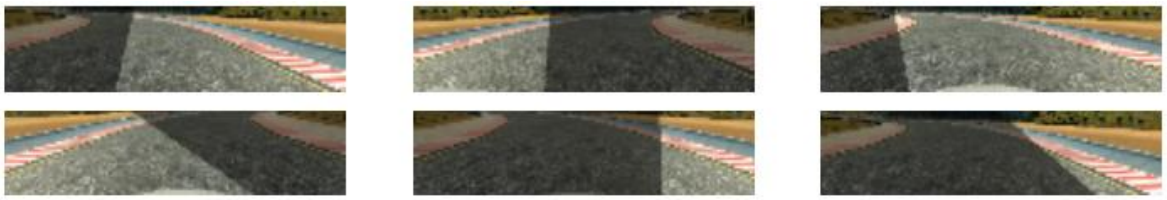

Рисунок 5.6 Случайное затемнение фрагментов изображения

5.4 Проектирование и интеграция нейросети в 3D-симуляцию

После экспериментов с различными видами глубоких свёрточных архитектур для обучения модели автомобиля самостоятельному вождению была выбрана девятислойная архитектура, предложенная Nvidia [19]. В процессе дальнейшей оптимизации и некоторых улучшений, направленных на повышение производительности вычислений и качества вождения, исходная архитектура свелась к довольно простой архитектуре с тремя свёрточными слоями и тремя полносвязными слоями (см. Рис. 5.7), которую можно очень кратко описать в Python с помощью Keras следующим образом [1]:

```
from keras import models
from keras.layers import core, convolutional, pooling

model = models.Sequential()
model.add(convolutional.Convolution2D(16,    3,    3,    input_shape=(32,    128,    3),
activation='relu'))
model.add(pooling.MaxPooling2D(pool_size=(2, 2)))
model.add(convolutional.Convolution2D(32, 3, 3, activation='relu'))
model.add(pooling.MaxPooling2D(pool_size=(2, 2)))
model.add(convolutional.Convolution2D(64, 3, 3, activation='relu'))
model.add(pooling.MaxPooling2D(pool_size=(2, 2)))
model.add(core.Flatten())
model.add(core.Dense(500, activation='relu'))
model.add(core.Dense(100, activation='relu'))
model.add(core.Dense(20, activation='relu'))
model.add(core.Dense(1))
```

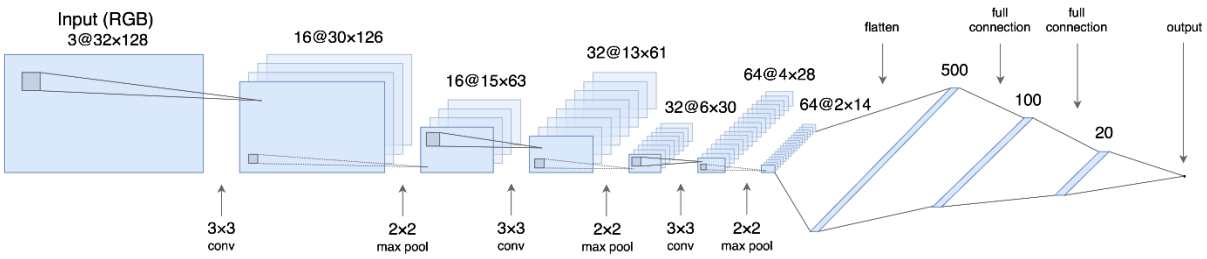

Рисунок 5.7 Архитектура свёрточной нейронной сети на базе Nvidia [19]

Спроектированная модель свёрточной нейронной сети (см. Рис. 5.7) показала наилучшие результаты по всем критериям беспилотного вождения в заданных условиях дорожного окружения на сгенерированных симулятором тестовых трассах, обучаясь с использованием оптимизатора Adam с начальной скоростью обучения равной 10^{-4} и среднеквадратичной ошибкой в качестве функции потерь на 80% сгенерированных обучающих данных. Кривая обучения нейросетевой модели проиллюстрирована на рисунке (5.8).

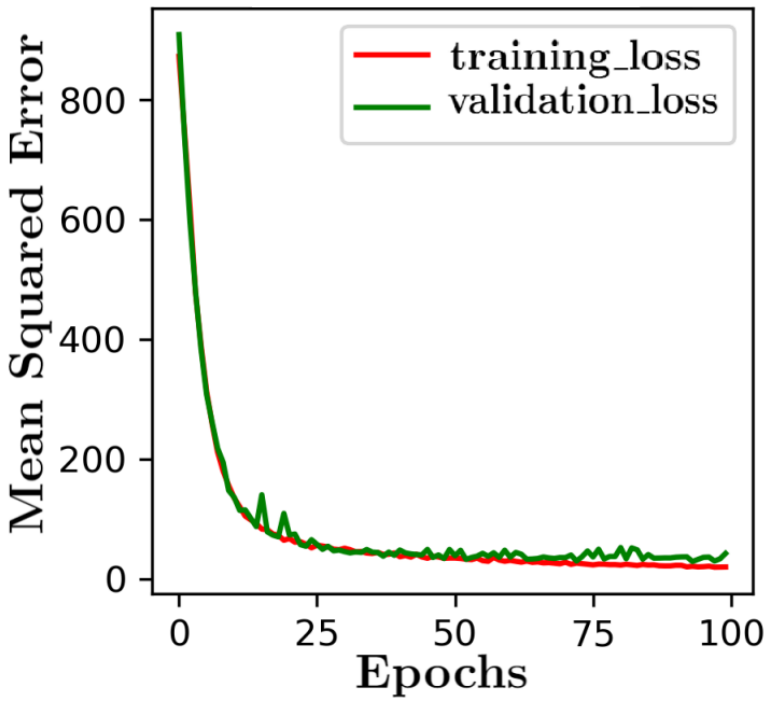

Рисунок 5.8 Зависимость среднеквадратичной ошибкой от эпох обучения

5.5 Заключение

В этой главе было рассмотрено практическое применение глубокого обучения свёрточных нейронных сетей на примере метода поведенческого клонирования к беспилотным транспортным средствам в различных генерируемых 3D-симуляциях дорожного окружения.

В частности, для воспроизведения поведения и риторики вождения автономных автомобилей в симуляторе дорожного окружения Udacity [18] была описана и обучена упрощённая и оптимизированная нейросетевая модель на основе многослойной нейронной архитектуры Nvidia [19], которая в заданных окружающих условиях показала наилучшие результаты по всем критериям беспилотного вождения, включая уверенное и плавное следование дорожной полосе, а также точное маневрирование на поворотах.

Несмотря на то, что обученная модель нейронной сети управляет исключительно углом поворота рулевой колодки, её функционал также можно расширить до управления ускорением и торможением, и даже научить реагировать на внезапные дорожные инциденты и безопасно избегать их — возможности машинного обучения и искусственного интеллекта в применении к беспилотным транспортным средствам на данный момент ограничиваются исключительно стеком технологий, вычислительными мощностями и идеями инженеров-математиков.

Полная реализация генерации и обработки тренировочных и тестовых наборов данных, интеграция модели глубокой свёрточной нейронной сети в 3-D симуляцию дорожного окружения, а также видео-демонстрации результатов обучения беспилотных моделей автомобилей самостоятельному вождению на различного рода трассах представлены в репозитории автора [1].

ЗАКЛЮЧЕНИЕ

В заключение к проведённой исследовательской и практической работе можно сделать несколько основных выводов.

На сегодняшний день представлено множество моделей беспилотных автомобилей и связанных с ними технологий от разных компаний, некоторые модели пока находятся в стадии проектирования или тестирования. Тем не менее, многие сопутствующие технологии пока недостаточно доступны производителям и потребителям. Решение этой проблемы требует времени и работы по совершенствованию методов автоматизации.

Многочисленные технологии принятия поведенческих решений строятся на основе методологий глубокого обучения и искусственного интеллекта: CNN, RNN, DRL и нереволюционных алгоритмов.

Из результатов тренировок в различных симуляциях можно сделать вывод, что нереволюционный алгоритм является очень эффективным методом обучения с подкреплением в случае самостоятельного вождения, избегания препятствий, быстрого реагирования и адаптации к различного вида изменениям условий дорожного окружения.

В случае с самостоятельной парковкой, алгоритму для достижения приемлемых результатов требуется более тысячи поколений, а также дополнительные параметры навигации по парковочной местности и локализации целевого парковочного места.

Описанная и обученная нейросетевая модель поведения и риторики вождения автономных автомобилей в заданных окружающих условиях показала наилучшие результаты по всем критериям беспилотного вождения. Функционал созданной модели можно расширять, совершенствуя кинематическую систему и визуализацию движения автомобиля.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. GitHub Repository: «Self-Driving Car Project», 2022 / Kirill Klimenko [Электронный ресурс] — Режим доступа: https://github.com/Defaultin/car-autopilot
2. World Health Organization, «Road traffic injuries», 2021. [Электронный ресурс] — Режим доступа: https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries
3. RAC Foundation, «Spaced out: perspectives on parking policy» 2012. [Электронный ресурс] — Режим доступа: https://www.racfoundation.org/research/mobility/spaced-out-perspectives-on-parking
4. «Creating Autonomous Vehicle Systems», 2020 / Shaoshan Liu, Liyun Li, Jie Tang, Shuang Wu, Jean-Luc Gaudiot
5. «Tesla Autopilot: Semi-Autonomous Driving», 2018 / Shantanu Ingle, Madhuri Phute
6. «Waymo Safety Report», 2018. [Электронный ресурс] — Режим доступа: https://storage.googleapis.com/sdc-prod/v1/safety-report/Safety%20Report%202018.pdf
7. «Waymo’s self-driving vehicles are now testing on public roads», 2019 / Korosec K. [Электронный ресурс] — Режим доступа: https://techcrunch.com/2019/06/17/waymos-self-driving-jaguar-i-pace-vehicles-are-now-testing-on-public-roads
8. «GM’s Cruise Origin Is an Autonomous Vehicle From the Future», 2020. [Электронный ресурс] — Режим доступа: https://www.extremetech.com/extreme/302323-meet-gms-cruiseorigin-of-the-autonomous-species

9. «Argo AI Safety Report», 2021. [Электронный ресурс] — Режим доступа: https://www.argo.ai/wp-content/uploads/2021/04/ArgoSafetyReport.pdf
10. « Self-Driving Cars Must Drive Hundreds of Millions of Miles», 2016. [Электронный ресурс] — Режим доступа: https://www.driverless-future.com/?page_id=983
11. «MCity ABC Test», 2019. [Электронный ресурс] — Режим доступа: https://mcity.umich.edu/wp-content/uploads/2019/01/mcity-whitepaper-ABC-test.pdf
12. «Shape and Arrangement of Columns in Cats Striate Cortex», 1963 / D. H. Hubel, T. N. Wiesel
13. «Introduction to Reinforcement Learning», 1998 / R. Sutton, A. Barto
14. «Q-Learning, Machine Learning», 1992 / C. Watkins, P. Dayan
15. «Human-level Control Through Deep Reinforcement Learning», 2015 / V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare
16. «Evolving Neural Networks through Augmenting Topologies», 2002 / K. O. Stanley, R. Miikkulainen
17. «NEAT-Python’s documentation». [Электронный ресурс] — Режим доступа: https://neat-python.readthedocs.io/en/latest
18. «Udacity's Self-Driving Car Simulator». [Электронный ресурс] — Режим доступа: https://github.com/udacity/self-driving-car-sim
19. «End to End Learning for Self-Driving Cars», 2016 / Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner. [Электронный ресурс] — Режим доступа: https://arxiv.org/pdf/1604.07316.pdf