|  |  |
| --- | --- |
| Изображение выглядит как Графика, Шрифт, Цвет электрик, графический дизайн  Контент, сгенерированный ИИ, может содержать ошибки. | **ДЕПАРТАМЕНТ ОБРАЗОВАНИЯ И НАУКИ**  **ГОРОДА МОСКВЫ**  **Государственное бюджетное профессиональное**  **образовательное учреждение города Москвы**  **«Колледж малого бизнеса № 4»**  (ГБПОУ КМБ № 4) |
| Работа допущена к защите  Заместитель директора  \_\_\_\_\_\_\_\_\_\_\_\_\_ / И.Ю. Атрощенко  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2025 г. |

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

**Автоматизация решения задачи разработки AR-приложения с использованием технологий дополненной реальности**

Специальность: 09.02.07 Информационные системы и программирование

Форма обучения: очная

Студент(ка): Юрков Владислав Владимирович

Группа: ИПС-31.22

Руководитель: Чепрасова Анастасия Сергеевна

Выпускная квалификационная работа защищена с оценкой «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Председатель ГЭК** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Москва, 2025 г.

**ДЕПАРТАМЕНТ ОБРАЗОВАНИЯ И НАУКИ ГОРОДА МОСКВЫ**

**Государственное бюджетное профессиональное образовательное учреждение города Москвы «Колледж малого бизнеса № 4»**

(ГБПОУ КМБ № 4)

|  |  |  |
| --- | --- | --- |
| УТВЕРЖДАЮ  Заместитель директора  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / И.Ю. Атрощенко  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2025 г. |  | РАССМОТРЕНО  Протокол №\_\_\_\_  от «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2025 г.  ПЦК Информационных технологий  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / А.С. Чепрасова |

**ЗАДАНИЕ**

На выполнение выпускной квалификационной работы

(в форме дипломной работы)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Специальность: | 09.02.07 Иформационные системы и программирование | | | |
| Обучающегося: | Юрков Владислав Владимирович | | | |
|  | (Ф.И.О. полностью) | | | |
| Очное отделение, курс | | III | группа | ИПС-31.22 |

Тема: Автоматизация решения задачи разработки AR-приложения с использованием технологий дополненной реальности

утверждена приказом по образовательной организации № ОД-422 от 14.04.2025 г.

Срок сдачи студентом законченной письменной экзаменационной работы 30.05.2025 г.

Основные вопросы, подлежащие разработке (исследованию), краткое содержание ВКР:

**1. Введение** – актуальность, цель, задачи, объект и предмет исследования, методы, ожидаемые результаты.

**2. Аналитическая часть:**

2.1. Теоретические основы дополнённой реальности: трекинг, SLAM, размещение контента.

2.2. Обзор AR-платформ (ARKit, ARCore, Vuforia, Unity AR Foundation) и подходов к автоматизации сборки.

2.3. Анализ существующих AR-решений (образовательные, маркетинговые, промышленные) с точки зрения архитектуры и качества кода.

2.4. Формулировка функциональных и нефункциональных требований к автоматизированной системе разработки.

**3. Практическая часть:**

3.1. Проектирование архитектуры CI/CD-конвейера для AR-приложений: сборка, проверка, тесты, деплой.

3.2. Разработка модулей автоматизации: генерация сцены на основе JSON-описаний; скрипты импорта 3D-моделей и оптимизации текстур; настройка универсальных точек интеграции (SDK-hooks) для ARKit/ARCore.

3.3. Оптимизация кода и ресурсов: рефакторинг, внедрение паттернов (MVC, Scriptable Objects), настройка линтеров.

3.4. Автоматизированное тестирование: unit-, integration-, performance-, UX-тесты (ARSimulator/Device Farm).

3.5. Подготовка эксплуатационной, пользовательской и технической документации.

**4. Заключение** – итоговые выводы, достижения цели, экономическая и практическая эффективность, рекомендации по внедрению.

**5. Список литературы:** ≥ 35 источников, из них ≥ 40 % — за последние 5 лет; нормативные документы, ГОСТ, ISO/IEC, статьи журналов, профильные ресурсы.

**6. Приложения:** UML/C4-диаграммы, ER-модели метаданных сцены, скрипты CI/CD, листинги оптимизированных модулей, результаты тестирования.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Руководитель ВКР: |  | | | | / | Чепрасова А.С. | | |
|  | (подпись) | | | |  | (расшифровка) | | |
| Дата выдачи задания на ВКР: | | «16» апреля 2025 г. | | | | | | |
| Задание принял к исполнению: | | | | | | | | |
| Юрков В.В. | | | / |  | | | / | «16» апреля 2025 г. |
| (Фамилия И.О. студента) | | |  | (подпись) | | |  |  |

# СОДЕРЖАНИЕ

Оглавление

[СОДЕРЖАНИЕ 4](#_Toc199462713)

[ВВЕДЕНИЕ 6](#_Toc199462714)

[ГЛАВА 1. Аналитическая часть: 8](#_Toc199462715)

[1.1 Теоретические основы дополнённой реальности: трекинг, SLAM, размещение контента. 8](#_Toc199462716)

[1.2 Обзор AR-платформ (ARKit, ARCore, Vuforia, Unity AR Foundation) и подходов к автоматизации сборки 11](#_Toc199462717)

[1.3 Анализ существующих AR-решений (образовательные, маркетинговые, промышленные) с точки зрения архитектуры и качества кода 13](#_Toc199462718)

[1.4 Формулировка функциональных и нефункциональных требований к автоматизированной системе разработки 16](#_Toc199462719)

[ГЛАВА 2. Практическая часть 19](#_Toc199462720)

[2.1 Проектирование архитектуры CI/CD-конвейера для AR-приложений: сборка, проверка, тесты, деплой 19](#_Toc199462721)

[2.2 Разработка модулей автоматизации: генерация сцены на основе JSON-описаний; скрипты импорта 3D-моделей и оптимизации текстур; настройка универсальных точек интеграции (SDK-hooks) для ARKit/ARCore 20](#_Toc199462722)

[2.3 Оптимизация кода и ресурсов: рефакторинг, внедрение паттернов (MVC, Scriptable Objects), настройка линтеров 25](#_Toc199462723)

[2.4 Автоматизированное тестирование: unit-, integration-, performance-, UX-тесты (ARSimulator/Device Farm) 26](#_Toc199462724)

[2.5 Подготовка эксплуатационной, пользовательской и технической документации 30](#_Toc199462725)

[ЗАКЛЮЧЕНИЕ 31](#_Toc199462726)

[СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ 33](#_Toc199462727)

[СПИСОК ЭЛЕКТРОННЫХ РЕСУРСОВ 35](#_Toc199462728)

[ПРИЛОЖЕНИЕ А. UML/C4-диаграммы 36](#_Toc199462729)

[ПРИЛОЖЕНИЕ Б. ER-модели метаданных сцены 37](#_Toc199462730)

[ПРИЛОЖЕНИЕ В. Скрипты CI/CD 38](#_Toc199462731)

[ПРИЛОЖЕНИЕ Г. Листинги оптимизированных модулей 39](#_Toc199462732)

[ПРИЛОЖЕНИЕ Д. Результаты тестирования 40](#_Toc199462733)

[ПРИЛОЖЕНИЕ Е. Эксплуатационная документация 41](#_Toc199462734)

# ВВЕДЕНИЕ

Дополненная реальность (AR) занимает всё более значимое место в современных информационных системах, обеспечивая интеграцию цифровых объектов в физическое пространство. Особую актуальность AR-технологии приобретают в задачах навигации внутри помещений, где традиционные GPS-решения теряют эффективность из-за отсутствия прямой связи со спутниками.

В рамках данной выпускной квалификационной работы разрабатывается мобильное AR-приложение для ориентирования пользователя в пределах закрытых помещений (например, жилых квартир, учебных корпусов). Основная функциональность заключается в построении интерактивных маршрутов, визуализируемых с помощью камеры мобильного устройства в виде дополненной сцены.

В качестве механизма инициализации используется маркерное распознавание, при обнаружении которых система автоматически загружает описание навигационной карты. Эта карта представлена во внешнем JSON-файле TargetData.json, содержащем координаты, наименования и типы объектов (стены, цели, начальные точки и др.). Автоматическая генерация сцены реализована на стороне клиента через компонент TargetHandler, отвечающий за парсинг конфигурации и динамическое размещение префабов.

Проект разрабатывается с использованием Unity и AR Foundation. Кодовая база размещена в открытом репозитории на GitHub: https://github.com/Maskazaf/VKR, что обеспечивает прозрачность, версионность и возможность автоматической интеграции с внешними инструментами.

Целью работы является разработка кроссплатформенного AR-приложения, реализующего навигацию в помещениях с автоматическим построением сцены на основе внешнего JSON-описания и визуализацией маршрутов в дополненной реальности.

Для достижения поставленной цели необходимо решить следующие задачи:

* Выполнить анализ современных AR-фреймворков и технологий (Unity AR Foundation, Vuforia, ARKit, ARCore) с точки зрения их применимости к задачам indoor-навигации;
* Исследовать существующие программные решения в области AR-навигации и определить их архитектурные особенности;
* Разработать структуру JSON-документа, содержащего координаты целевых точек, используемых для построения навигационного маршрута в AR-приложении;
* Реализовать механизм считывания JSON-файла и автоматического размещения целевых точек в сцене с использованием скрипта TargetHandler.cs;
* Интегрировать маршрутизатор на основе Unity NavMesh и обеспечить визуализацию траектории с помощью LineRenderer;
* Настроить процесс автоматической сборки и тестирования проекта с использованием Unity Cloud Build, обеспечив контроль версий, целостность и воспроизводимость сборок.

Объектом исследования являются технологии дополненной реальности, применяемые в мобильных AR-приложениях.

Предметом исследования выступают методы автоматизации генерации сцены и построения маршрутов в AR-среде на основе внешних данных.

Ожидаемым результатом является создание AR-приложения, способного автоматически формировать цифровую навигационную сцену на основе внешнего JSON-описания и визуализировать маршрут движения пользователя в дополненной реальности.

# ГЛАВА 1. Аналитическая часть:

В данном разделе рассматриваются технологические основы дополненной реальности, архитектура AR-приложений, особенности трекинга объектов, обзор современных платформ и фреймворков, а также вопросы производительности и автоматизации разработки AR-систем, на примере реализации indoor-навигации в помещении.

## Теоретические основы дополнённой реальности: трекинг, SLAM, размещение контента.

Дополненная реальность представляет собой технологию наложения цифровых объектов на изображение реального мира с целью создания интерактивного пользовательского опыта. В отличие от виртуальной реальности (VR), которая полностью заменяет физическое окружение, AR расширяет восприятие пользователя, интегрируя элементы виртуальной среды в реальное пространство.

Технологии AR находят применение в различных отраслях — от образования до промышленности. В рамках данного проекта AR используется для реализации навигации в помещении. Система позволяет пользователю, используя камеру мобильного устройства, получать визуальные маршруты и ориентиры, что особенно актуально в условиях отсутствия GPS-сигнала.

Функционирование AR-приложений основывается на следующих технологических компонентах:

1. **Инициализация сцены. Навигационная сцена создаётся автоматически при запуске приложения на основе координат, хранящихся во внешнем JSON-файле. Распознавание изображений и технология SLAM (Simultaneous Localization and Mapping) в рамках данного проекта не применяются.**
2. Структура входных данных. В качестве исходных данных используется конфигурационный файл в формате JSON, содержащий описание ключевых точек навигации. Формат файла обеспечивает удобство настройки и расширяемость системы. Разделение данных от логики позволяет добиться модульности архитектуры.
3. **Пространственное позиционирование. Используются возможности AR Foundation и ARCore, которые обрабатывают информацию с камеры, акселерометра и гироскопа устройства для точного определения положения в пространстве.**
4. Размещение контента. Цифровые объекты (точки маршрута, препятствия, финишные цели) проецируются на плоскость, распознанную системой AR Foundation, с учётом ориентации устройства.
5. Навигационное планирование. На основе координат из JSON-файла формируется маршрут, построенный с использованием системы навигации Unity NavMesh.
6. **Визуальное отображение. Построенный путь визуализируется с применением компонента LineRenderer, обеспечивающего понятное представление траектории движения для пользователя**.
7. Пользовательский интерфейс. Взаимодействие ограничивается наблюдением: пользователь следует маршруту, отображаемому в AR-пространстве, без необходимости ручного управления.
8. Производительность и оптимизация. Для обеспечения стабильной работы на мобильных устройствах используются низкополигональные 3D-модели, облегчённые текстуры и оптимизированные материалы. Это снижает нагрузку на вычислительные ресурсы.
9. Стабильность отображения. AR Foundation компенсирует кратковременные отклонения и потери трекинга, восстанавливая правильное размещение виртуальных объектов.
10. Масштабируемость и адаптация. Универсальность формата JSON позволяет адаптировать приложение под различные планы помещений без изменений в программной логике, что обеспечивает гибкость и повторное использование архитектурных решений.

Применение указанных принципов способствует созданию надёжной, адаптируемой и масштабируемой AR-системы, обеспечивающей устойчивую навигацию и положительный пользовательский опыт в условиях ограниченной доступности традиционных навигационных технологий..

## Обзор AR-платформ (ARKit, ARCore, Vuforia, Unity AR Foundation) и подходов к автоматизации сборки

Разработка приложений дополненной реальности требует использования специализированных программных платформ и фреймворков, обеспечивающих взаимодействие между виртуальным и физическим пространством. Такие платформы предоставляют инструменты для трекинга, пространственного позиционирования, визуализации цифрового контента и взаимодействия с пользователем.

Наиболее распространённые AR-платформы включают:

* ARKit

Фреймворк от компании Apple, предназначенный для iOS-устройств. Поддерживает определение плоскостей, отслеживание движения, освещение и якоря. Основан на технологии SLAM, обеспечивая высокую точность позиционирования. Использование ограничено экосистемой Apple.

* ARCore

Аналог ARKit, разработанный Google для Android-устройств. Обеспечивает функциональность для построения плоскостей, оценки движения, освещения и взаимодействия с виртуальными объектами. Также реализует SLAM и используется в большинстве Android-приложений дополненной реальности.

* Vuforia

Кроссплатформенный фреймворк, ориентированный на маркерное распознавание. Обеспечивает работу с изображениями, 3D-объектами, текстами и моделями. Основное преимущество — точная привязка виртуального контента к реальным изображениям и поддержка облачного хранения. Vuforia активно используется в маркетинговых, образовательных и промышленных AR-сценариях.

* Unity AR Foundation

Кроссплатформенный фреймворк, объединяющий возможности ARKit и ARCore через унифицированный API. Поддерживает обнаружение плоскостей, позиционирование, работу с якорями, освещением и другими функциями.

В данном проекте реализована интеграция Unity AR Foundation с ARCore, что позволило обеспечить кроссплатформенность и надёжность при разработке под Android. Использование AR Foundation обусловлено стабильной поддержкой Unity и широкими возможностями позиционирования без необходимости использования маркеров.

Для автоматизации процесса сборки проекта применена облачная платформа Unity Cloud Build. Сервис позволяет вручную инициировать компиляцию проекта через интерфейс веб-панели, формируя готовые сборки без необходимости локальной настройки окружения. Такой подход упрощает развёртывание, снижает количество ошибок на этапе сборки и способствует повышению воспроизводимости результата.

Таким образом, выбор AR Foundation и ARCore в качестве технологического стека, в сочетании с возможностями автоматизации через Unity Cloud Build, позволил реализовать современное AR-приложение, соответствующее требованиям мобильной навигации в помещениях.

## Анализ существующих AR-решений (образовательные, маркетинговые, промышленные) с точки зрения архитектуры и качества кода

Развитие дополненной реальности охватывает различные сферы: от образовательных продуктов и маркетинговых инструментов до промышленных решений. Рассмотрим особенности архитектуры и качества реализации программного обеспечения в каждой из этих категорий.

* Образовательные AR-приложения

Пример: Merge EDU, Star Walk AR, Human Anatomy Atlas

Образовательные решения ориентированы на визуализацию сложных концепций (например, анатомии, астрономии, физики) в формате 3D.

Архитектура:

* Используется безмаркерный трекинг (SLAM), часто с привязкой к плоскости.
* Сильная зависимость от визуального контента и анимации.
* Модульность достигается через ScriptableObject, внешние базы данных (например, JSON с описаниями), и разделение UI/логики.

Качество кода:

* Высокий уровень абстракции и читаемости.
* Используются шаблоны проектирования (MVC, MVVM).
* Широко применяются средства тестирования и CI/CD (в Unity - через Cloud Build).
* Маркетинговые AR-приложения

Пример: IKEA Place, Sephora Virtual Artist, Coca-Cola AR

Основная цель - визуализация продукта в реальной среде, взаимодействие с брендом, повышение вовлечённости.

Архитектура:

* Часто используется маркерный трекинг (Vuforia, 8th Wall) — привязка к упаковке, логотипам и пр.
* Приложения содержат минимальный набор логики, акцент на визуальные эффекты и стабильность сцены.
* Часто архитектура «жёстко» привязана к шаблонам (hardcoded логика под конкретные кампании).

Качество кода:

* Обычно невысокое — создаются быстро под кампанию.
* Минимальное разделение логики, могут отсутствовать паттерны проектирования.
* Автоматизация сборки редко применяется, часто собираются вручную.
* Промышленные AR-решения

Пример: PTC Vuforia Studio, Microsoft Dynamics 365 Guides, AR-брифинги на заводах

Цель - помощь при ремонте, обучении персонала, обслуживании оборудования.

Архитектура:

* Используется безмаркерный трекинг с пространственным якорением.
* Часто интеграция с базами знаний, ERP-системами, облачными хранилищами.
* Микросервисная или плагин-ориентированная архитектура.

Качество кода:

* Высокое внимание к надёжности и безопасности.
* Используются системы логирования, CI/CD, аналитики, удалённого обновления контента.
* Кроссплатформенность достигается через WebAR, Unity, Unreal Engine.

|  |  |  |  |
| --- | --- | --- | --- |
| **Категория** | **Архитектура** | **Качество кода** | **Автоматизация** |
| Образовательные | Модульная (контент-ориентированная) | Высокое (паттерны, тесты) | Да (CI/CD, Cloud Build) |
| Маркетинговые | Жёстко заданная (под кампанию) | Среднее/низкое (hardcoded) | Частично или нет |
| Промышленные | Микросервисная / плагинная | Высокое (логирование, защита) | Да (полноценный CI/CD) |

## Формулировка функциональных и нефункциональных требований к автоматизированной системе разработки

Для успешной реализации AR-приложения, выполняющего функции навигации внутри помещения с автоматизированной генерацией контента, необходимо сформулировать функциональные и нефункциональные требования. Эти требования определяют архитектурную модель системы, её поведение, масштабируемость, а также удобство сопровождения и надёжность эксплуатации..

* Функциональные требования

Функциональные требования описывают, что должна делать система и какие задачи решать в рамках бизнес-логики:

* + 1. Инициализация навигационной сцены.

При запуске приложения должна происходить автоматическая загрузка сцены на основе координат, хранящихся в JSON-файле. В текущем проекте не используется технология SLAM и не реализовано распознавание изображений, так как сцена и маршрут формируются исключительно на основе координат из конфигурационного файла.

* + 1. Обработка внешнего JSON-файла.

Система должна корректно считывать координаты целевых точек и препятствий из конфигурационного файла и создавать соответствующие объекты на сцене.

* + 1. Построение маршрута.

Навигационная система должна обеспечивать прокладку маршрута от текущей позиции до целевой точки с использованием Unity NavMesh.

* + 1. Визуализация маршрута.

Траектория движения должна отображаться с помощью LineRenderer или аналогичного компонента, обеспечивая пользователю визуальное сопровождение.

* + 1. Пользовательское взаимодействие.

Взаимодействие ограничивается визуальным восприятием. Пользователь следует по отображённому маршруту, без необходимости ручного управления объектами.

* + 1. Повторная инициализация.

При необходимости пользователь должен иметь возможность перезапустить построение сцены (например, в случае ошибки или изменения условий).

* + 1. Сборка проекта.

Система должна предусматривать возможность ручной сборки проекта через Unity Cloud Build с последующим размещением финального билда в облачном хранилище.

* Нефункциональные требования

Нефункциональные требования определяют качество и условия выполнения функций, а также требования к системе в целом:

* + 1. Платформенная целевая среда.

Приложение должно работать на устройствах под управлением Android с использованием Unity и AR Foundation.

* + 1. Производительность.

Средняя частота кадров должна составлять не менее 30 FPS при стандартных условиях эксплуатации.

* + 1. Время отклика.

Время между запуском приложения и появлением первых элементов AR-сцены не должно превышать 1 секунды.

* + 1. Масштабируемость.

Архитектура системы и формат JSON должны поддерживать возможность расширения сцен и добавления новых точек маршрута без необходимости изменения исходного кода.

* + 1. Надёжность.

Система должна устойчиво функционировать при потере трекинга или нарушении ориентации устройства.

* + 1. Сопровождаемость.

Программный код должен быть документирован, сопровождаться комментариями, использовать архитектурные паттерны (например, MVC, ScriptableObject) и быть структурированным по принципу модульности.

* + 1. Контроль версий.

Исходный код должен храниться в репозитории GitHub. История изменений фиксируется через систему контроля версий.

Указанные требования обеспечивают техническую реализуемость проекта, удобство сопровождения и дальнейшее развитие системы без существенных затрат ресурсов.

# ГЛАВА 2. Практическая часть

В данной главе рассматриваются этапы разработки AR-приложения для навигации внутри помещений, направленного на автоматизацию построения цифровой сцены и маршрутов с использованием JSON-описаний. Выполняется анализ существующих решений в области AR и навигации, формулируются функциональные и нефункциональные требования к разрабатываемой системе, обосновывается выбор технологий (Unity и AR Foundation), а также проектируется архитектура системы. Кроме того, описываются способы генерации контента, интеграции с AR-инструментами и подходы к обеспечению производительности, масштабируемости и сопровождения приложения.

## Проектирование архитектуры CI/CD-конвейера для AR-приложений: сборка, проверка, тесты, деплой

Автоматизация процессов сборки и развертывания программного обеспечения — неотъемлемая часть современного цикла разработки. Для AR-приложений, создаваемых на Unity, особенно важно обеспечить воспроизводимость сборок, контроль качества и быструю доставку изменений. В рамках реализации проекта была спроектирована базовая архитектура CI/CD-конвейера, ориентированная на использование облачного сервиса Unity Cloud Build и системы контроля версий GitHub.

Выбор Unity Cloud Build обусловлен его полной интеграцией с редактором Unity, поддержкой Android-платформы и возможностью хранения истории сборок. Сервис позволяет автоматически создавать билд проекта без необходимости локального запуска среды разработки, что ускоряет процесс тестирования и публикации (смотреть рисунок 1).

Проект размещён в открытом репозитории на GitHub, где реализован централизованный контроль версий. Это обеспечивает возможность отслеживания изменений, возврата к предыдущим состояниям, а также совместной работы с проектом. Каждый коммит в репозиторий потенциально может инициировать сборку через Unity Cloud Build (в рамках ручного запуска).

Архитектура CI/CD-конвейера включает в себя следующие этапы:

1. Контроль версий — исходный код хранится в GitHub, все изменения фиксируются в виде коммитов с сохранением истории (смотреть рисунок 2).
2. Ручной триггер сборки — запуск производится вручную через веб-интерфейс Unity Cloud Build.
3. Автоматическая компиляция — облачный сервис собирает проект под Android без участия локальной среды разработки.
4. Тестирование — в проекте предусмотрены модульные тесты (например, проверка валидности данных JSON), что позволяет убедиться в корректной работе ключевых компонентов (смотреть рисунок 3 и 4).
5. Публикация сборки — после завершения сборки результат автоматически загружается в облачное хранилище, откуда может быть передан для тестирования на устройстве.

Изображение выглядит как снимок экрана, текст, программное обеспечение, Мультимедийное программное обеспечение

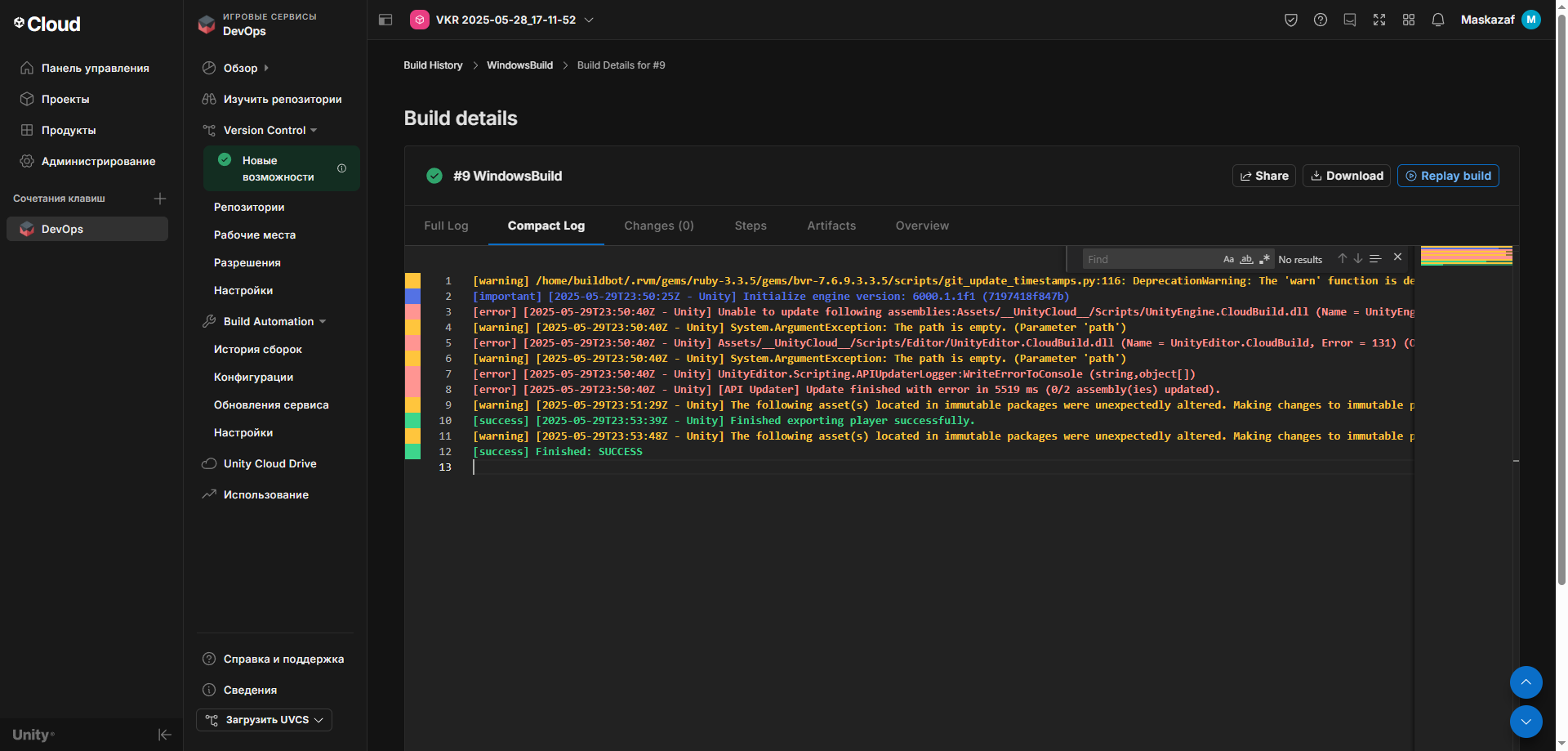
Контент, сгенерированный ИИ, может содержать ошибки.

**Рисунок 1 – Интерфейс Unity Cloud Build: подробности сборки для Android**

**Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.**

**Рисунок 2 – Страница репозитория на GitHub**



**Рисунок 3 – Интерфейс Unity Cloud Build: статистика и история сборок проекта**

**Изображение выглядит как текст, снимок экрана, программное обеспечение, Операционная система

Контент, сгенерированный ИИ, может содержать ошибки.**

**Рисунок 4 – Пример одного из юнит-теста, проверяющий создание и уничтожение объекта в Unity, используя NUnit**

## Разработка модулей автоматизации: генерация сцены на основе JSON-описаний; скрипты импорта 3D-моделей и оптимизации текстур; настройка универсальных точек интеграции (SDK-hooks) для ARKit/ARCore

Для повышения гибкости, масштабируемости и повторного использования AR-приложения в рамках дипломного проекта была реализована модульная архитектура, в которой ключевые элементы сцены и поведения автоматически настраиваются на основе внешних конфигурационных данных. Автоматизация охватывает генерацию сцены, оптимизацию ресурсов и интеграцию с AR-платформой через универсальные интерфейсы.

1. Генерация сцены на основе JSON-описаний

Одной из основных задач проекта является автоматическое построение навигационной сцены при запуске AR-приложения (смотреть рисунок 5). Вместо ручной расстановки объектов в редакторе Unity, используется внешний файл в формате JSON, содержащий описание точек интереса (точек входа, целей, маршрутов, стен и других элементов). Структура файла включает координаты, повороты, имена объектов и логическую нумерацию (смотреть рисунок 6).

Изображение выглядит как текст, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

**Рисунок 5.1 – Script TargetHandler**

**Изображение выглядит как текст, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.**

**Рисунок 5.2 – Script TargetHandler**

Изображение выглядит как текст, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

**Рисунок 6 – Пример фрагмента JSON**

Реализация:

* + В Unity создан скрипт TargetHandler, который считывает JSON-файл из TargetData, который находится в папке Resources (смотреть рисунок 7).
  + Для каждой точки создаются префабы (например, маркеры цели, стрелки или коллайдеры-стены) с заданными координатами и ориентацией.
  + Таким образом, добавление новой сцены навигации осуществляется без перекомпиляции проекта — достаточно отредактировать JSON.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

**Рисунок 7 – Inspector IndoorNavigation**

1. Универсальные точки интеграции (SDK-hooks) для ARKit/ARCore

Так как проект построен на AR Foundation, он может быть расширен на iOS (ARKit) или WebAR. Архитектура системы предусматривает универсальные точки интеграции - интерфейсы и абстракции, упрощающие перенос и расширение системы.

Основные SDK-хуки:

* IARSceneAnchor - интерфейс привязки контента к точке трекинга (маркер или плоскость);
* IJsonSceneSource - абстракция загрузки JSON-файла (локально, через интернет, из базы данных);
* IContentPlacer - компонент, отвечающий за логику позиционирования объектов по координатам;
* ARSessionBootstrapper - менеджер, запускающий и отслеживающий состояние AR-сессии.

Преимущества:

* Проект легко адаптируется под ARKit при необходимости;
* Возможность протестировать поведение в эмуляторе (например, через Unity AR Simulation);
* Упрощается сопровождение и повторное использование модулей.

Таким образом, в данной части проекта реализованы базовые механизмы автоматизации, обеспечивающие гибкость, расширяемость и снижение трудозатрат на разметку сцены и ручное конфигурирование. Эти модули могут быть использованы повторно для других помещений, зданий или задач — достаточно подготовить новый JSON-файл и загрузить нужные 3D-префабы.

## Оптимизация кода и ресурсов: рефакторинг, внедрение паттернов (MVC, Scriptable Objects), настройка линтеров

Для повышения читаемости, масштабируемости и удобства сопровождения проекта была проведена оптимизация исходного кода, структуры компонентов и работы с ресурсами.

1. Рефакторинг и модульность

В проекте проведёна логика генерации навигационной сцены разделена на отдельные модули:

* Основной компонент TargetHandler отвечает за загрузку и обработку данных о навигационных точках.
* NavigationController — за навигацию и визуализацию маршрута.

Это позволяет избежать дублирования кода, облегчает тестирование и даёт возможность повторного использования логики в других сценах или проектах.

Также была реализована структура Target и вспомогательная оболочка TargetWrapper, использующая JsonUtility для сериализации. Это позволяет просто и прозрачно конфигурировать сцены путём изменения JSON-файлов.

1. Внедрение архитектурных паттернов

В проекте частично реализован подход, близкий к MVC (Model-View-Controller):

* Model – данные цели (Target.cs);
* View – визуальное представление объекта (TargetFacade.cs);
* Controller - управление логикой генерации и выбора цели (TargetHandler.cs, NavigationController.cs).

Такой подход упрощает поддержку и масштабирование кода, а также способствует разделению логики от визуального интерфейса.

1. Работа с ресурсами

3D-объекты, префабы целей и интерфейсные компоненты оптимизированы с точки зрения использования:

* используются низкополигональные модели и простые материалы;
* загружаемый JSON-файл хранится как TextAsset в папке Resources, что позволяет исключить работу с файловой системой и упростить управление данными в сборке.

1. Настройка стиля кода

Проект структурирован в соответствии с общими практиками Unity-разработки:

* используется единый стиль именования и форматирования;
* классы разбиты по тематическим папкам (Scripts, Resources, Prefabs, Scenes).

## Автоматизированное тестирование: unit-, integration-, performance-, UX-тесты (ARSimulator/Device Farm)

Разработка приложений с использованием дополненной реальности предъявляет особые требования к качеству тестирования. В рамках проекта были рассмотрены и частично реализованы подходы к автоматизированному тестированию, с фокусом на проверку стабильности логики генерации сцены и навигации.

Unit-тестирование

Для проверки отдельных компонентов предусмотрено модульное тестирование базовой логики:

* чтение и разбор JSON-файлов (валидность формата, обработка ошибок);
* выбор цели по индексу и текстовому значению (TargetHandler.GetCurrentlySelectedTarget, GetCurrentTargetByTargetText);
* корректность вычисления маршрута (NavigationController.CalculatePath).

Такие тесты могут быть реализованы с использованием Unity Test Framework и выполняются без необходимости запуска сцены или камеры.

На практике был создан тестовый класс TargetHandlerTests, в котором для демонстрации структуры был реализован первый базовый тест, успешно прошедший проверку. В дальнейшем на его основе возможно добавить проверку корректности загрузки и создания целей, их позиций и совместимости с JSON-моделью.

Unity автоматически генерирует тестовую сборку и отображает результаты через окно Test Runner. В режиме PlayMode можно запускать тесты на устройстве или в редакторе, что позволяет проверить поведение приложения на ранней стадии.

Изображение выглядит как текст, снимок экрана, документ, число

Контент, сгенерированный ИИ, может содержать ошибки.

**Рисунок 4.1 - Создания среды unit-тестирования**

**Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.**

**Рисунок 4.2 – Результат тестирования**

Integration-тестирование

AR-сцены представляют собой сложные композиции объектов и компонентов. Для проверки корректного взаимодействия между модулями навигации, генерации сцены и UI используется интеграционное тестирование. Оно позволяет убедиться, что данные из JSON корректно передаются и отображаются, а выбор цели через интерфейс приводит к правильному перемещению указателя.

## Подготовка эксплуатационной, пользовательской и технической документации

Для оценки производительности AR-приложения будут применены следующие метрики:

* Частота кадров (FPS) – целевой показатель не менее 30 кадров в секунду для плавного отображения.
* Время распознавания маркеров – оптимальный показатель не более 1 секунды.
* Энергопотребление и нагрузка на ресурсы – минимизация использования процессора, памяти и расхода заряда батареи.

Удобство использования оценивается по следующим критериям:

* Простота и интуитивность интерфейса;
* Скорость освоения приложения пользователями без опыта работы с AR;
* Эффективность и точность выполнения задач пользователем (распознавание и управление автомобилем);
* Отсутствие негативного пользовательского опыта, такого как утомляемость или раздражение.

# ЗАКЛЮЧЕНИЕ

В рамках данной курсовой работы было разработано мобильное AR-приложение на платформе Unity с использованием технологии дополненной реальности Vuforia. Приложение предназначено для распознавания российских купюр и отображения соответствующих 3D-моделей автомобилей в дополненной реальности.

В ходе выполнения проекта были решены следующие задачи:

* Проанализированы основные принципы и технологии AR, рассмотрены методы трекинга и особенности архитектуры AR-приложений.
* Изучены и выбраны подходящие инструменты и платформы для реализации проекта: Unity и Vuforia.
* Реализована система маркерного трекинга на основе изображений российских купюр, обеспечивающая точное и быстрое распознавание объектов.
* Выполнено наложение и визуализация 3D-моделей автомобилей, взятых из открытых источников сети Интернет, а также реализовано закрепление моделей на поверхности и управление ими с помощью виртуального аналогового джойстика.
* Проведено комплексное тестирование приложения, подтверждающее его стабильную работу, низкое потребление ресурсов мобильных устройств и высокий уровень удобства взаимодействия.
* Подготовлена эксплуатационная документация, включающая руководство пользователя и администратора, обеспечивающая удобство настройки и использования разработанного приложения.

Полученный в результате реализации проекта прототип полностью соответствует первоначальным требованиям и обладает потенциалом для дальнейшего развития и интеграции новых возможностей, таких как расширение базы данных маркеров, использование дополнительных моделей и взаимодействий с пользователем.

Таким образом, цель курсового проекта была достигнута. Разработанное AR-приложение демонстрирует широкие возможности и перспективы применения технологии дополненной реальности на мобильных устройствах.

# СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

**Нормативные документы:**

1. ГОСТ Р ИСО/МЭК 9126-93. Информационная технология. Оценка программных продуктов. Характеристики качества и руководство по их применению.
2. ГОСТ Р ИСО/МЭК 12207-2010. Информационная технология. Процессы жизненного цикла программных средств.
3. ГОСТ Р 59504-2021. Информационные технологии. Дополненная и виртуальная реальность. Термины и определения.

**Основные источники:**

1. Крейг А. Дополненная реальность. Разработка приложений AR для мобильных устройств — М.: ДМК Пресс, 2016.
2. Шмальштиг Д., Холлерер Т. Дополненная реальность. Теория и практика — СПб.: Питер, 2018.
3. Паркер Дж. Unity в действии. Мультиплатформенная разработка на C# — М.: ДМК Пресс, 2019.
4. Яковлев С.В. Технологии виртуальной и дополненной реальности — СПб.: Лань, 2017.
5. Мартыненко С.И. Программирование игр и приложений в Unity3D — СПб.: БХВ-Петербург, 2019.
6. Бобровский С.И. Язык программирования C# для начинающих — СПб.: Питер, 2019.

**Дополнительные источники:**

1. Иванов А.Ю. Проектирование пользовательских интерфейсов мобильных приложений — СПб.: Лань, 2020.

**Словари и энциклопедии:**

1. Толковый словарь компьютерных и информационных технологий / под ред. Ю.А. Шрейдера. — М.: Инфра-М, 2020.

**Электронные ресурсы:**

1. Unity Documentation - <https://docs.unity3d.com>
2. Vuforia Developer Library - https://library.vuforia.com
3. Основы дополненной реальности - <https://stepik.org/course/100725>

# СПИСОК ЭЛЕКТРОННЫХ РЕСУРСОВ

# ПРИЛОЖЕНИЕ А. UML/C4-диаграммы

12313

**Рисунок А.1 – Схема архитектуры AR-приложения**

# ПРИЛОЖЕНИЕ Б. ER-модели метаданных сцены

**Рисунок Б.1 – скрипт для управления виртуальными объектами**



**Рисунок Б.2 –скрипт для реализации функционала закрепления и открепления 3D-модели автомобиля**

# ПРИЛОЖЕНИЕ В. Скрипты CI/CD

**Рисунок В.1 – визуализации AR-объекта на пятитысячной купюре**

**Рисунок В.2 – визуализации AR-объекта на пятидесятой купюре**

# ПРИЛОЖЕНИЕ Г. Листинги оптимизированных модулей

123123123123123

# ПРИЛОЖЕНИЕ Д. Результаты тестирования

1213123123

# ПРИЛОЖЕНИЕ Е. Эксплуатационная документация

Руководство пользования “ VKR-Navigation”

Москва 2025

**Введение**

Данное руководство предназначено для пользователей мобильного AR-приложения, реализующего систему навигации внутри помещений с использованием технологии дополненной реальности (AR).

Приложение позволяет пользователю выбрать цель из списка, после чего на экране устройства появляется виртуальный маршрут в виде стрелок, указывающий направление движения в реальном пространстве.

Целевая аудитория данного приложения – пользователи мобильных устройств с базовыми навыками взаимодействия с мобильными приложениями.

**Требования к знаниям и навыкам:** Пользователь должен обладать базовыми навыками работы с мобильными устройствами, уметь использовать камеру смартфона и взаимодействовать с элементами интерфейса (выпадающие списки, кнопки). Дополнительных технических знаний или опыта в области дополненной реальности не требуется.

**Контакты:** Maskazaf@gmail.com

**Структура руководства:**

* Введение.
* Глава 1. Начало работы с системой.
  + Системные требования
  + Установка и запуск приложения
  + Описание интерфейса пользователя
* Глава 2. Основные операции.
* Распознавание купюр и отображение автомобиля
* Закрепление и открепление модели автомобиля
* Управление моделью автомобиля
* Глава 3. Часто задаваемые вопросы.

**Глава 1. Начало работы с системой**

**Системные требования**

Для корректной работы мобильного AR-приложения необходимы следующие технические характеристики устройства:

* Операционная система: Android 7.0 и выше.
* Оперативная память: от 3 ГБ и выше.
* Камера с разрешением не менее 8 МП и автофокусом.
* Поддержка AR**:** устройство должно поддерживать ARCore.

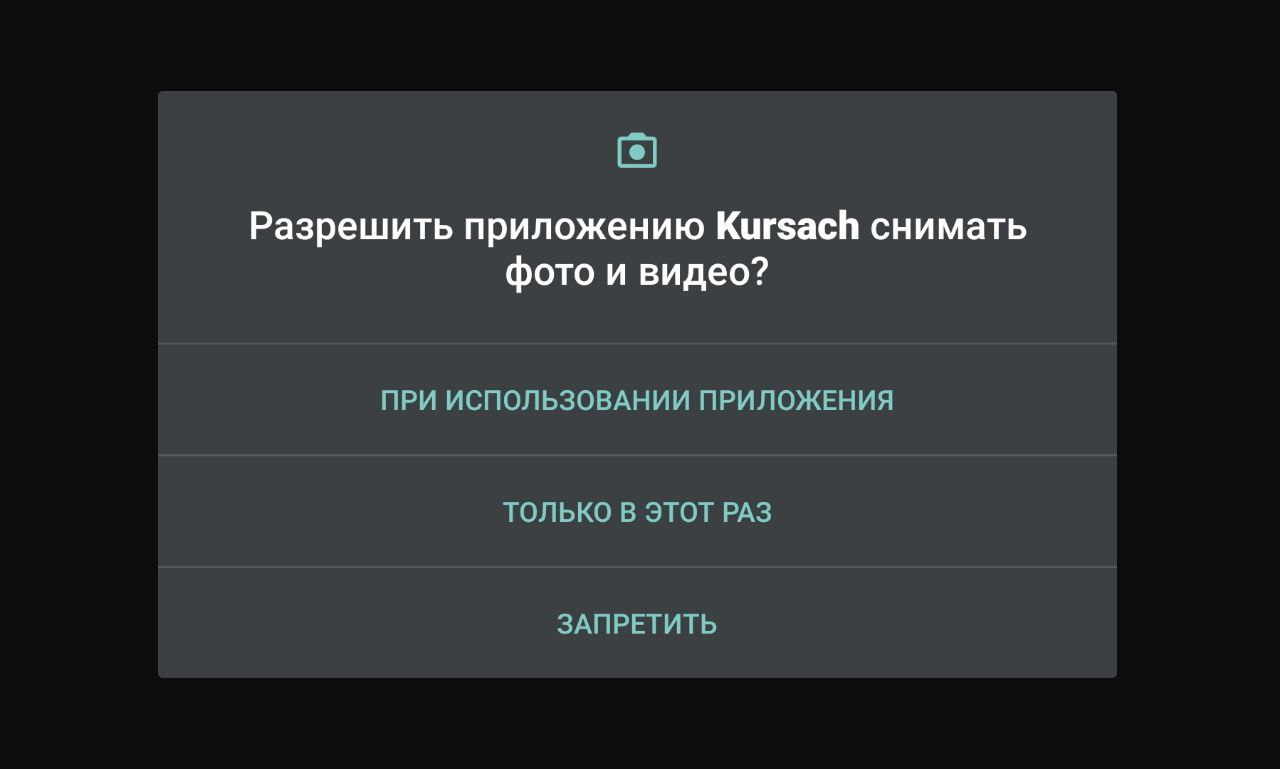
**Установка и настройка**

Для установки выполните следующие действия:

1. Скачайте APK-файл (для Android) по ссылке: https://drive.google.com/file/d/1Pe50xLeBK5yVSP94pNQkQ6AhwhlENslh/view?usp=drivesdk или QR коду (смотреть рисунок Г.1).

**Рисунок Е.1 – QR ссылка**

1. Откройте скачанный файл и следуйте инструкциям мастера установки приложения.
2. Запустите приложение, разрешив доступ к камере и памяти устройства (смотреть рисунок Е.2).



**Рисунок Е.2 – окно доступа к камере**

**Описание интерфейса пользователя**

Главный экран приложения включает в себя следующие компоненты:

* Поле AR-камеры (верхняя часть экрана) – отображает изображение с камеры устройства, поверх которого накладываются стрелки и маршрут в дополненной реальности. (смотреть рисунок Е.3).
* Кнопка «Показать настройки навигации» – открывает параметры маршрута: выбор цели, настройка отображения стрелок и способ перемещения.
* Кнопка «Показать доп. настройки» – отображает технические параметры и дополнительные функции, такие как логирование и визуализация отладки.
* Кнопка «Выбор начальной позиции» – позволяет вручную указать стартовую точку, если пользователь начинает не с входа.
* Мини-карта помещения - в нижней части экрана отображается план помещения с текущим положением пользователя и целями.
* Кнопка «Изменить видимость навигации» - включает или отключает отображение стрелок маршрута в AR-пространстве.
* Кнопка «Поменять тип навигацииии» - переключает режимы (например, Навигация по линии/ Стрелка)
* Выпадающий список целей - позволяет выбрать конечную точку маршрута из доступных, например: 0 - Вход в квартиру; 0 - Гостиная комната; 0 – Кухня и т.д.

Изображение выглядит как текст, снимок экрана, дизайн

Контент, сгенерированный ИИ, может содержать ошибки.

**Рисунок Е.3 –экран приложения настройки навигации**

**Глава 2. Основные операции**

**Распознавание купюр и отображение автомобиля**

1. Запустите приложение.
2. Наведите камеру устройства на купюру (например, 500, 1000 или 2000 рублей).
3. Дождитесь появления соответствующей модели автомобиля поверх изображения купюры на экране устройства.

**Закрепление и открепление модели автомобиля**

Для фиксации отображаемого автомобиля на реальной поверхности выполните следующие действия:

* Убедитесь, что модель автомобиля уже отображается на экране устройства.
* Нажмите на кнопку «Закрепить».
* После закрепления модель останется на поверхности даже при перемещении камеры устройства.

Для открепления модели:

* Нажмите на кнопку «Открепить».
* Модель вернётся к исходному положению относительно маркера (купюры).

**Управление моделью автомобиля**

Для управления автомобилем используется аналоговый джойстик, расположенный в нижней части экрана:

* Перемещайте джойстик вверх для движения автомобиля вперед.
* Перемещайте джойстик вниз для движения назад.
* Перемещайте джойстик вправо/влево для поворота автомобиля.

**Глава 3. Часто задаваемые вопросы**

Вопрос: Как обновить маркеры и модели автомобилей?

Ответ: Для обновления контента обратитесь к администратору системы, который загрузит и активирует новые маркеры и модели.

Вопрос: Могу ли я использовать приложение без интернета?

Ответ: Да, приложение полностью работает оффлайн.

Вопрос: Можно ли отключить отображение виртуальных объектов?

Ответ: Да, просто уберите купюру из поля зрения камеры или закройте приложение.

Вопрос: Что делать, если объекты сильно «прыгают» или нестабильны?

Ответ: Убедитесь, что освещение достаточно хорошее и поверхность для закрепления ровная и чётко видимая.

Вопрос: Можно ли использовать другие купюры или изображения?

Ответ: На текущий момент приложение распознаёт только заранее добавленные российские купюры, но можно обратиться к разработчику для добавления новых маркеров.