МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное
учреждение высшего образования
«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «КубГУ»)

Факультет компьютерных технологий и прикладной математики
Кафедра информационных технологий

КУРСОВАЯ РАБОТА

РАЗРАБОТКА МОБИЛЬНОГО ПРИЛОЖЕНИЯ ДОПОЛНЕННОЙ РЕАЛЬНОСТИ ДЛЯ ОС IOS С ИСПОЛЬЗОВАНИЕМ ПЛАТФОРМЫ ARKIT

Работу выполнил _____К.Р. Сеидов
(подпись)

Направление подготовки 01.03.02 Прикладная математика и информатика

Направленность (профиль) Системное программирование и компьютерные технологии (Математическое и программное обеспечение вычислительных машин)

Научный руководитель
канд. техн. наук, доц. _____С.Г. Синица
(подпись)

Нормоконтролёр_____А.В.Харченко
(подпись)

Краснодар 2018

РЕФЕРАТ

Курсовая работа содержит 33 страницы, 10 рисунков, 8 источников.

ДОПОЛНЕННАЯ РЕАЛЬНОСТЬ, AR, AUGMENTET REALITY, ARKIT, UNITY, GLTF, РАЗРАБОТКА, ТЕХНОЛОГИИ, 3D-МОДЕЛИ, IOS.

Объектом исследования является технология дополненной реальности, а также платформа ARKit, игровой движок Unity, формат 3D-моделей GLTF.

Целью курсовой работы является изучение технологии дополненной реальности, принципов разработки приложений для iOS и принципов работы с графическими движками на примере Unity.

В результате была создана программная реализация AR-приложения, функциями которой являются:

– построение виртуальной плоскости по множеству точек;

– загрузка сохраненной в память устройства 3D-модели формата GLTF;

– размещение 3D-модели на виртуальной плоскости и создание эффекта дополненной реальности.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

На сегодняшний день технология дополненной реальности (англ. *augmented reality, AR*) является одной из самых перспективных и быстроразвивающихся в сфере информационных технологий в XXI веке. Спектр её применения достаточно широк: начиная с развлекательных приложений и заканчивая программным обеспечением для интерфейса кабины военных истребителей. В последние несколько лет технология дополненной реальности нашла широкое применение в сфере мобильных развлечений. Наиболее ярким примером мобильного приложения с функцией дополненной реальности является «Pokémon Go», число скачиваний которой в первые 19 дней после выхода в цифровой магазин составило порядка 75 миллионов. Данный пример продемонстрировал, насколько перспективным является рынок мобильных AR-приложений.

На сегодняшний день технические характеристики современных смартфонов позволяют с помощью таких платформ, как ARKit для ОС iOS создавать программы с все более и более широким функционалом. Процент AR-приложений в цифровых магазинах AppStore и GooglePlay продолжает возрастать, и многие аналитики прогнозируют увеличение в несколько раз объёма рынка виртуальной и дополненной реальности к 2020 году.

В данной курсовой работе будет рассмотрено создание AR-приложения для ОС iOS с функцией realtime-загрузки произвольной 3D-модели формата GLTF. Данный подход позволяет в будущем загружать модели и анимации к ним с сервера.

Целью курсовой работы является разработка мобильного приложения дополненной реальности для ОС IOS с использованием платформы ARKIT.

Исходя из названной цели, в работе поставлен ряд задач:

– изучить технологии дополненной реальности;

– провести обзор возможностей существующих методов отображения дополненной реальности;

– обосновать выбор среды разработки;

– изучить принципы работы в графических движках на примере Unity;

– разработать и представить на рассмотрение программный продукт.

Общую теоретико-методологическую базу курсового проекта составили труды зарубежных исследователей, опубликованные на различных иностранных сайтах.

1 Технология дополненной реальности, 3D-модели. Основные понятия

1.1 Технология AR

1.1.1 Общая информация

Дополненная реальность (Augmented reality, AR) — это область исследований, ориентированная на использование компьютеров для совмещения реального мира и данных, сгенерированных компьютером.

На сегодняшний день большинство исследований в области AR сконцентрировано на использовании живого видео, подвергнутого цифровой обработке и «дополненного» компьютерной графикой.

Изначально термин AR был введён в противовес виртуальной реальности: вместо погружения пользователя в синтезированное, полностью информационное окружение, задачей AR является дополнение реального мира возможностями по обработке дополнительной информации. Другие же исследователи понимают виртуальную реальность как специальный случай дополненной реальности.

Первым исследователем дополненной реальности можно считать Айвэна Сазерленда, который построил работающий прототип системы в 1967 году. Он использовал стереоочки Sword of Damocles для показа трехмерной графики.

Современный этап исследований начался в 1990 году, когда исследователи фирмы Boeing решили использовать наголовные стереодисплеи при сборке и обслуживании самолётов, накладывая интерактивную графику на изображения реального мира. Одним из наиболее известных исследователей в этой области сегодня является Рональд Азума из HRL Laboratories. В 1997 г. он опубликовал большую обзорную статью «A Survey of Augmented Reality» [6], где впервые были ясно очерчены проблемы и возможности, связанные с внедрением этой технологической концепции. Он определяет AR как систему, которая:

1) совмещает виртуальное и реальное;
2) взаимодействует в реальном времени;
3) работает в 3D.

Герман Бенес пишет: «Дополненная реальность определяется в соответствии с контекстом и при этом рассматривается не как абстрактная запись, а так, словно, объекты дополненной реальности существуют в природе и жизни. Дополненная реальность — это инструмент, который позволпяет одному или многим наблюдателям расширить своё поле зрение при помощи виртуальных элементов, обычно созданных компьютером» [1].

Он определил следующие правила, которые могут быть определены как необходимые для того, чтобы дополненная реальность была принята бизнесом, образованием и обществом:

1) полная интерактивность в реальном времени;
2) точное и сверхбыстрое отслеживание;
3) стереоскопия;
4) сверхпортативность и беспроводность;
5) ощущение «полного погружения».

Большинство людей, которые интересуются AR, считают одной из самых важных характеристик способ, которым осуществляется трансформация места, где происходит взаимодействие. В интерактивной системе важно не просто точно определить местоположение, но и распознать окружение.

Три основные составляющие персональной системы дополненной реальности:

1) носимый компьютер;
2) средства отображения;
3) средства позиционирования.

И если первые два пункта не представляют особой сложности в реализации в связи с появлением графических процессоров, достаточной для обработки высокодетализированной 3D-графики, мощности, а также LED-

дисплеями высокого разрешения, то вот проблему позиционирования виртуальных объектов относительно пользователя решить удалось лишь недавно.

1.1.2 Решение проблемы позиционирования

С момента зарождения концепции AR существовала проблема позиционирования в пространстве виртуальных объектов относительно пользователя.

Использование «классической» навигационной системы GPS не решало проблему. Ведь идея дополнения реальности заключается в получении правильной информации в правильном месте. «Классический» датчик GPS дает минимальную погрешность от 3 до 30 м. Учитывая то, что подписи на дисплее должны совмещаться с изображением реальных объектов с точностью до сантиметров, такая ошибка делает затею бессмысленной.

Поскольку основной задачей дополненной реальности является синтез реальных и виртуальных объектов в пространстве, то возникает необходимость в предварительной обработке данных об окружающем пространстве. Регистрирование геометрических характеристик небольших помещений сегодня уже стало нормой жизни для широкого круга специалистов. Все оказывается гораздо сложнее, когда речь заходит об открытых пространствах: как взаимно расположены виртуальные и реальные предметы, какой из них находится на первом плане? Работа здесь ведется по двум направлениям: съемка «карты глубин» (depth sensing) в реальном масштабе времени и предварительный сбор информации о местности.

Комбинация гироскопа и компаса дает неплохие результаты, а если добавить к ним распознавание изображений заранее известных элементов ландшафта — точность возрастает до пиксельного уровня. Расхождение измеряется пикселями исключительно из-за особенностей человеческого

мозга и зрения, способных выявить малейший промах при размещении виртуальных предметов в реальном пространстве.

Пользу здесь приносят даже разработки создателей спецэффектов в кино, которые занимаются восстановлением траектории движения камеры с помощью отслеживания перемещений в кадре объектов-маркеров.

В 2018 году компания Apple презентовала обновлённую линейку своих смартфонов. Одним из наиболее обсуждаемых нововведений стало распознавание лица. И хоть на первый взгляд связи с AR не так много, но именно возможность камеры строить математическую модель лица, или сказав по-другому – распознавать поверхность, дало технологиям AR новый толчок.

Рассмотрим технологию на примере Spectra системы от Qualcomm.

Частью этой системы является расположенный рядом камерой модуль, который генерирует инфракрасное излучение. Он испускает лазерные лучи в инфракрасном диапазоне, которые падают на различные точки поверхности, а в свою очередь датчик воспринимает обратную связь и соответствующим образом реагирует. Например, оценивает положение поверхности относительно камеры и то, под каким углом она расположена. Совместное использование этой технологии с акселерометром, гироскопом и возможностями машинного зрения позволяет построить наиболее точную на сегодняшний день сцену дополненной реальности.

1.1.3 Рынок дополненной реальности

Дополненная реальность переживала несколько витков развития и сейчас мы видим новый виток, при котором впервые дополненная реальность имеет массовый эффект.

Революция, которую мы сейчас наблюдаем, — это следствие появления так называемой «четвертой платформы» (рис. 1).

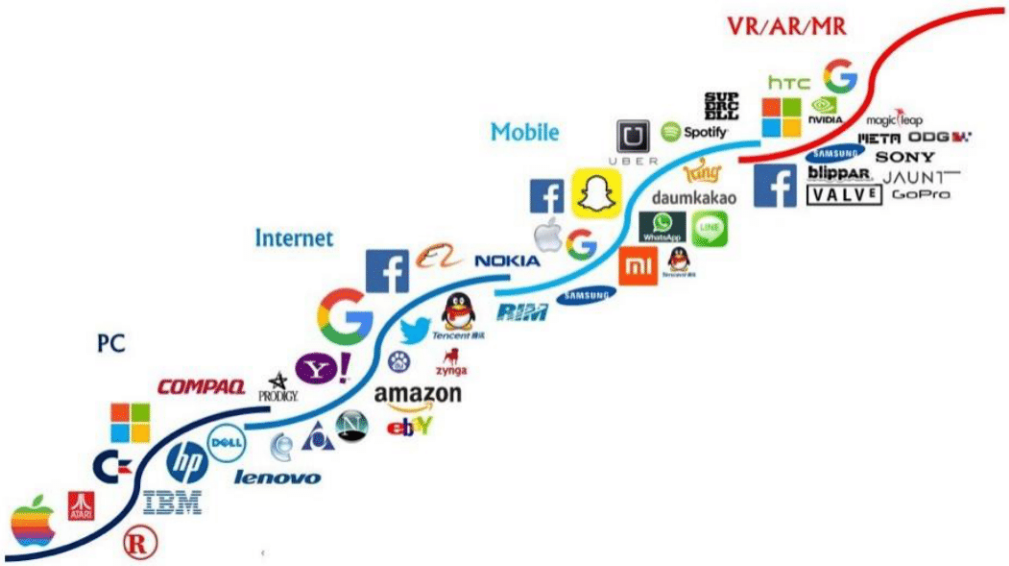

Рисунок 1 – Виток развития платформ

Первая платформа – персональные компьютеры, которые появились в конце 80-х – начале 90-х, потом пришел интернет, следующим этапом стали мобильные технологии. Сейчас мобильное потребление уже больше, чем на персональных компьютерах. Виртуальная и дополненная реальность – следующая платформа, для которой будут создаваться новые рынки, предложения и бизнес. Сейчас тот самый момент, когда надо инвестировать в VR/AR и развивать технологии в этой сфере.

По оценке аналитиков, сейчас объём рынка виртуальной и дополненной реальности в выручке от продаж контента и устройств составляет несколько миллиардов долларов, но уже к 2020 году будет составлять более $150 млрд (рис. 2).

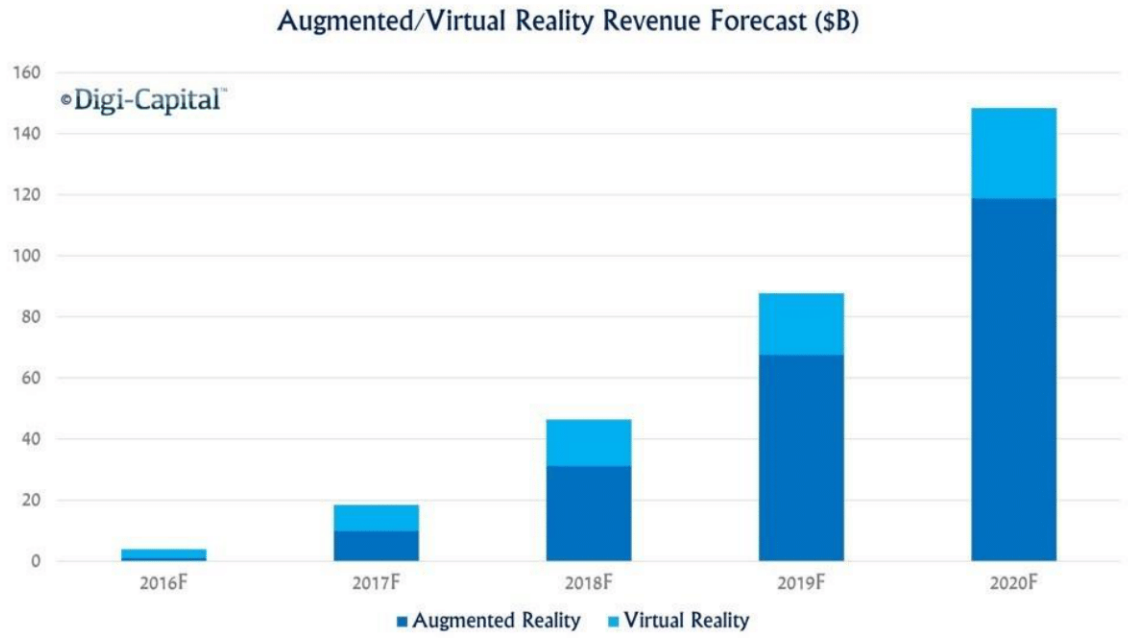

Рисунок 2 – Прогноз AR/VR рынка, произведенный порталом Digi-Capital в 2016 году

Это огромная возможность для инвесторов. Сейчас основная выручка создается шлемами виртуальной реальности и приложениями, который для них создается. Но картина будет меняться – большая ставка будет сделана на дополненную реальность.

Синим на диаграмме отображена выручка от сервисов, приложений и очков дополненной реальности. Как мы видим, что к 2020 году доля рынка AR-технологии превысит VR в несколько раз.

Прогнозируемые гигантские размеры рынка порождают огромные возможности для создания собственного бизнеса.

1.2 Понятие игрового движка.

Игровой движок – это среда разработки программного обеспечения, предназначенная для людей, создающих видеоигры. Разработчики используют игровые движки для создания игр для консолей, мобильных устройств и персональных компьютеров. Основные функциональные возможности, обычно предоставляемые игровым движком, включают в себя механизм рендеринга («рендерер» или генерация/обработка графического изображения) для 2D или 3D-графики, физический движок или обнаружение столкновений (и реагирование на столкновения), звук, сценарии, анимацию, искусственный интеллект, сетевое взаимодействие, потоковую передачу, память управление, многопоточность, поддержка локализации, графические сцены и может включать в себя поддержку видео для кинематографии.

Во многих случаях игровые движки предоставляют набор инструментов визуальной разработки в дополнение к повторно используемым программным компонентам. Эти инструменты обычно предоставляются в интегрированной среде разработки, что позволяет упростить и ускорить разработку игр на основе данных. Большинство игровых движков предоставляют средства, облегчающие разработку, такие как графика, звук, физика и функции искусственного интеллекта. Эти игровые движки иногда называют «промежуточным программным обеспечением», потому что, как и в деловом смысле этого слова, они предоставляют гибкую и многократно используемую программную платформу, которая предоставляет все основные функциональные возможности, необходимые для разработки игрового приложения, при одновременном снижении затрат.

Как и другие типы промежуточного программного обеспечения, игровые движки обычно предоставляют абстракцию платформы, позволяя запускать одну и ту же игру на различных платформах, включая игровые приставки и персональные компьютеры, с небольшими изменениями в

исходном коде игры, если таковые имеются. Зачастую игровые движки разрабатываются на основе компонентной архитектуры, которая позволяет заменять или расширять конкретные системы в движке более специализированными (и зачастую более дорогими) компонентами игрового промежуточного программного обеспечения. Некоторые игровые движки спроектированы как набор слабо связанных компонентов промежуточного программного обеспечения, которые можно выборочно комбинировать для создания собственного движка вместо более распространенного подхода к расширению или настройке гибкого интегрированного продукта. Несмотря на то, что расширяемость достигается, она остается высоким приоритетом для игровых движков из-за большого разнообразия областей применения, для которых они применяются. Несмотря на специфику названия, игровые движки часто используются для других видов интерактивных приложений с графическими потребностями в реальном времени, такими как маркетинговые демонстрации, архитектурные визуализации, симуляции обучения и среды моделирования.

Некоторые игровые движки предоставляют только возможности 3D-рендеринга в реальном времени вместо широкого спектра функций, необходимых для игр. Эти движки полагаются на разработчика игр, чтобы реализовать остальную часть этой функциональности или собрать ее из других компонентов промежуточного программного обеспечения игры. Эти типы движков обычно упоминаются как «графический движок», «движок рендеринга» или «трехмерный движок» вместо более всеобъемлющего термина «игровой движок». Эта терминология используется непоследовательно, поскольку многие полнофункциональные игровые движки называются просто «трехмерными движками». Современные игровые или графические движки обычно предоставляют графические сцены, который является объектно-ориентированным представлением игрового мира 3D, который часто упрощает игровой дизайн и может использоваться для более эффективного рендеринга обширных виртуальных миров.

1.3 Формат 3D-моделей GLTF.

1.3.1 Общая информация

GLTF (GL Transmission Format) - это формат файлов для хранения 3D-сцен и моделей, использующий стандарт JSON, разработанный командой Khronos Group и анонсированный на HTML5DevConf 2016. Создатели описывают его как «JPEG 3D». Предполагается, что GLTF будет эффективным, совместимым форматом, который сжимает размер 3D-сцен и минимизирует их время обработки для приложений, использующие WebGL и другие API.

3D-сцена, сохраненная в формате gltf, представляет собой структуру данных, схематично изображенную на рисунке 3, состоящую из несколько файлов:

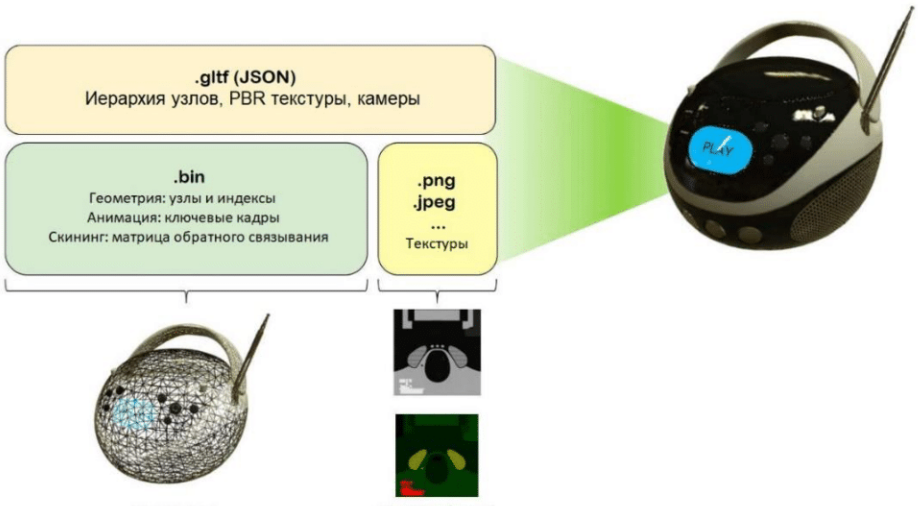

Рисунок 3 – Схема структуры данных GLTF-модели

1) Gltf – файл, в котором хранится сама структура модели, иерархия ее частей (так называемых узлов), текстуры материалов (англ. Physically Based Rendering, PBR – физически обоснованный рендеринг, при котором

некоторые физические свойства объектов, такие как отражение, принимаются в расчет) и информация о камере в этой сцене;

2) Bin – файл, который хранит геометрию узлов и их индексацию, информацию о ключевых кадрах анимации, а также матрицу обратного связывания, необходимую для связи «скелета» модели (набора джоинтов) и ее «плоти» (3D-меша);

3) файлы изображения, хранящие текстуры модели.

1.3.2 Преимущества и недостатки формата

GLTF 2.0 даёт разработчикам шанс стандартизировать рабочий процесс между программным обеспечением для 3D-моделирования и игровыми движками. GLTF – это формат, основанный на JSON, текстовый формат обмена данными, основанный на JavaScript. Это значительно снижает необходимость полагаться на сложные сторонние библиотеки для его чтения. Многие языки (например, Javascript, Python) поддерживают этот формат изначально.

Первым преимуществом является отдельный двоичный блоб (BLOB – с англ. Binary Large OBject двоичный большой объект – массив двоичных данных для хранения изображений, аудио и видео и тд). GLTF позволяет указывать внешний файл для данных в формате двоичного кода. Он также поддерживает многие распространенные типы данных графического процессора, поэтому на практике этот файл можно перемещать частями непосредственно в память графического процессора. GLTF является первым из таких форматов, который можно эффективно использоваться в качестве внутренних данных в графическом движке. Импорт и экспорт GLTF-моделей происходит крайне быстро в сравнении с другими форматами, и это делает возможным внесения изменение в файл с помощью ПО для 3D-моделирования и его последующее обновления в движке, происходящее почти мгновенно.

Вторым и не менее важным является отсутствие у данного формата возможности разночтения. Файловая структура GLTF не содержит никаких дополнительных или избыточных данных. Существует только один способ понять определение сцены. Это существенно упрощает процесс импорта, гарантируя, что все существующие сцены будут работать с первой попытки внедрения.

Также нет поддержки различных систем координат или единиц. GLTF использует правую систему координат, то есть, перекрестное произведение X и Y приводит к Z. GLTF определяет ось Y как вверх. Единицами измерения для всех линейных расстояний являются метры. Все углы в радианах. Положительное вращение против часовой стрелки.

Кроме того, GLTF поддерживает многие современные функции и особенности своих конкурентов. GLTF 2.0 полностью поддерживает скелеты и морфы цели, которые могут быть проанализированы легко и однозначно. Он также поддерживает материалы на основе PBR с металлическим, шероховатым, нормальным, эмиссионным непрозрачным покрытием и обрабатывает двусторонние и прозрачные материалы, включая частично-прозрачные покрытия.

Поддержка анимации также хорошо сделана. GLTF 2.0 поддерживает несколько анимаций на файл, что идеально для экспорта действий. Анимации хранятся в двоичном файле, что делает возможным её быструю загрузку.

Не смотря на ряд своих преимуществ в плане организации данных, формат GLTF был раскритикован некоторыми разработчиком 3D-движков. Так Эрик Ленгель [2] указывает на то, что GLTF не хватает многих возможностей, когда речь идет о структуре сцены, применении материалов, создания обложек и освещения, а также невозможности указать физические единицы или глобальный вектор направления вверх. Он также утверждает, что поддержка анимации в GLTF недостаточна для точного воспроизведения анимаций, созданных в большинстве популярных программ моделирования (включая Blender, Autodesk 3ds Max и Autodesk Maya).

Ведущий разработчик Godot Engine Хуан Линецкий[1] резко критиковал GLTF 1.0. Однако он выступил за то, чтобы GLTF 2.0 был стандартным форматом обмена активами для игровых движков. Он утверждает, что во многих случаях сокращение возможностей на самом деле является преимуществом, устраняя неопределенность и делая импорт файла проще в реализации

На текущий момент формат непрерывно развивается обретает все большую популярность. Команда The Khronos Group Inc непрерывно совершенствует свое детище, добавляя новые функции и совершенствуя уже имеющиеся.

2 Мультиплатформенная среда разработки Unity. Платформа ARKit

2.1 Мультиплатформенная среда разработки Unity

2.1.1 Общие сведения о Unity

Unity – это кроссплатформенный игровой движок, разработанный Unity Technologies, впервые анонсированный и выпущенный в июне 2005 года на Всемирной конференции разработчиков Apple Inc. в качестве эксклюзивного игрового движка для OS X. На момент 2018 года движок был расширен для поддержки 27 платформ.

Unity дает пользователям возможность создавать игры как в 2D, так и в 3D, движок предлагает основной API для написания скриптов на C# как для редактора Unity в виде плагинов, так и для самих игр.

2.1.1 Возможности

Редактор Unity имеет простой Drag&Drop интерфейс, который легко настраивать, состоящий из различных окон, благодаря чему можно производить отладку игры прямо в редакторе. Движок поддерживает C# в качестве скриптового языка. Расчёты физики производит физический движок PhysX от NVIDIA.

Проект в Unity делится на уровни, которые представлены отдельными файлами сцен, содержащие свои игровые миры со своим набором объектов, игровых сценариев, так называемые скрипты, и настроек. Сцены могут содержать в себе как объекты, содержащие модели, так и пустые игровые объекты. Они в свою очередь содержат наборы компонентов, с которыми и взаимодействуют скрипты. Также у объектов есть название, им возможно присвоить метку, так называемый тег, и слой, на котором он должен

отображаться. Так, у любого объекта на сцене обязательно присутствует компонент Transform, который хранит в себе координаты местоположения, поворота и размеров объекта по всем трём осям. У объектов с видимой геометрией также по умолчанию присутствует компонент Mesh Renderer, делающий модель объекта видимой. К объектам можно добавлять такие компоненты, как коллайдеры (англ. collider), которые отвечают за параметры сталкивания объектов.

Также Unity поддерживает физику твёрдых тел и ткани, а также физику типа Ragdoll (тряпичная кукла). В редакторе имеется система наследования объектов; дочерние объекты будут повторять все изменения позиции, поворота и масштаба родительского объекта. Скрипты в редакторе прикрепляются к объектам в виде отдельных компонентов.

Редактор Unity поддерживает написание и редактирование шейдеров. Редактор Unity имеет компонент для создания анимации, но также анимацию можно создать предварительно в 3D-редакторе и импортировать вместе с моделью, а затем разбить на файлы.

Unity 3D поддерживает систему Level Of Detail (LOD), суть которой заключается в том, что на дальнем расстоянии от игрока высокодетализированные модели заменяются на менее детализированные, и наоборот, а также систему Occlusion culling, суть которой в том, что у объектов, не попадающих в поле зрения камеры, не визуализируется геометрия и коллизия, что снижает нагрузку на центральный и графический процессоры и позволяет оптимизировать проект. При компиляции проекта создается исполняемый файл (*.exe) игры, а в отдельной папке – данные игры, включая все игровые сцены и динамически подключаемые библиотеки.

Движок поддерживает множество популярных форматов. Модели, звуки, текстуры, материалы, скрипты можно запаковывать в формат *.unityassets и передавать другим разработчикам или выкладывать в свободный доступ. Этот же формат используется во внутреннем магазине Unity Asset Store, в котором разработчики могут бесплатно или за деньги

выкладывать в общий доступ различные элементы, нужные при создании игр.

2.1.3 Основные преимущества и недостатки

Как правило, игровой движок предоставляет множество функциональных возможностей, позволяющих их задействовать в различных играх, в которые входят моделирование физических сред, карты нормалей, динамические тени и многое другое. В отличие от многих игровых движков, у Unity имеется два основных преимущества: наличие визуальной среды разработки и межплатформенная поддержка. Первый фактор включает не только инструментарий визуального моделирования, но и интегрированную среду, цепочку сборки, что направлено на повышение производительности разработчиков, в частности, этапов создания прототипов и тестирования. Под межплатформенной поддержкой предоставляется не только места развертывания, но и наличие инструментария разработки.

Еще одним преимуществом является модульная система компонентов Unity, с помощью которой происходит конструирование игровых объектов, когда последние представляют собой комбинируемые пакеты функциональных элементов. В отличие от механизмов наследования, объекты в Unity создаются посредством объединения функциональных блоков, а не помещения в узлы дерева наследования. Такой подход облегчает создание прототипов, что актуально при разработке игр.

В качестве недостатков приводятся ограничение визуального редактора при работе с многокомпонентными схемами, когда в сложных сценах визуальная работа затрудняется. Вторым недостатком является отсутствие поддержки в Unity ссылок на внешние библиотеки, работу с которыми программистам приходится настраивать самостоятельно; это также затрудняет командную работу. Ещё один недостаток связан с использованием шаблонов экземпляров (англ. prefabs). Также, WebGL-версия

движка, в силу специфики своей архитектуры (трансляция кода из C# в C++ и далее в JavaScript), имеет ряд нерешённых проблем с производительностью, потреблением памяти и работоспособностью на мобильных устройствах.

2.2 Платформа ARKit

2.2.1 Основная информация

ARKit — это SDK (software development kit) для работы с дополненной реальностью, анонсированный компанией Apple на WWDC 2017 и полноценно представленный в возможностях iOS 11. Благодаря ему, порог вхождения в эту технологию AR стал значительно ниже.

Основным требованием к любому AR-опыту и определяющим признаком ARKit является возможность создавать и отслеживать соответствие между реальным пространством, в котором живет пользователь, и виртуальным пространством, в котором вы можете моделировать визуальный контент. Когда ваше приложение отображает виртуальные объекты вместе с изображением с камеры в реальном времени, пользователь испытывает дополненную реальность: иллюзию того, что ваш виртуальный контент является частью реального мира. ARKit сочетает в себе отслеживание движения устройства, захват сцены с помощью камеры и расширенную обработку этой сцены, чтобы упростить задачу создания AR-приложений.

ARKit использует системы координат мира и камеры, в которой в качестве базиса используется правая тройка векторов. Так ось Y направлена вверх, ось Z направлена на зрителя, а ось X – вправо.

Конфигурации сеансов могут изменить происхождение и ориентацию системы координат относительно реального мира. Каждый якорь (англ. anchor) в сеансе AR определяет свою собственную локальную систему координат, также основанную на правой тройке векторов.

2.2.1 Принцип отслеживания мира

Чтобы создать соответствие между реальным и виртуальным пространством, ARKit использует технику, называемую визуально-инерциальную одометрию (англ. visual-inertial odometry). Этот процесс комбинирует информацию, полученную информацию с датчиков обнаружения движения устройства iOS, с анализом по средствам компьютерного зрения сцены, распознаваемой камерой устройства (рис. 4). ARKit распознает заметные особенности в изображении сцены, отслеживает различия в расположении этих объектов в видеокадрах и сравнивает эту информацию с данными о движении. В результате получается высокоточная модель положения и движения устройства в пространстве. Иными словами, происходит анализ полученного изображения под разными углами, когда устройство перемещается и переориентируется в физическом пространстве (требуется перемещение; вращение не дает достаточно информации). Изображения, полученные в этом процессе, используются вместе, чтобы понять глубину; больше это похоже на то, как люди воспринимают глубину двумя глазами

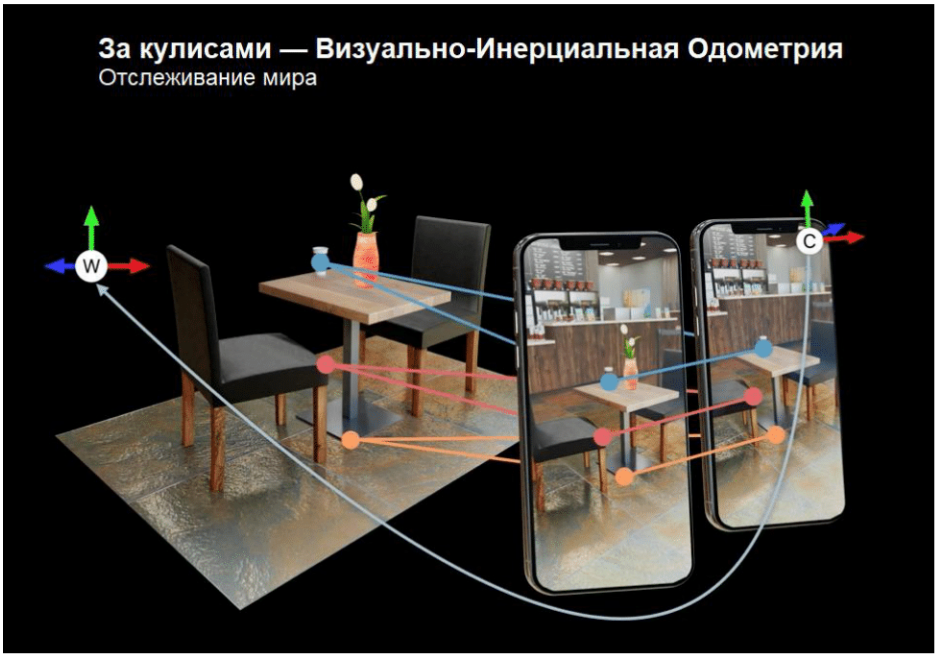

Рисунок 4 – Принцип отслеживания мира

Это генерирует то, что Apple называет картой мира, которую можно использовать для позиционирования и ориентации объектов, применения к ним освещения и теней и многого другого. Чем больше пользователь перемещается и переориентируется, тем больше информации отслеживается и тем точнее и реалистичнее могут стать объекты дополненной реальности. Когда ARKit строит карту мира, он сопоставляет ее с виртуальным координатным пространством, в которое могут быть помещены объекты.

Устройство нуждается в непрерывных данных датчика, и этот процесс лучше всего работает в хорошо освещенных средах, которые имеют текстурированные поверхности и очень четкие характеристики.

ARKit отслеживает качество карты мира под капотом и указывает на одно из трех состояний, о которых разработчикам рекомендуется по-своему сообщать пользователям:

1) недоступно: карта мира еще не построена;
2) ограниченный: какой-то фактор помешал построить адекватную карту мира, поэтому функциональность и точность могут быть ограничены;
3) обычный: карта мира достаточно надежна, чтобы можно было ожидать хорошего дополнения.

Отслеживание мира также анализирует и понимает содержание сцены. Используйте методы проверки попадания, чтобы найти реальные поверхности, соответствующие точке на изображении с камеры. Если вы включите настройку Plane Detection в своей конфигурации сеанса, ARKit обнаружит плоские поверхности на изображении с камеры и сообщит об их положении и размерах. Вы можете использовать результаты теста на удар или обнаруженные плоскости для размещения или взаимодействия с виртуальным контентом в вашей сцене.

2.2.2 Основные возможности ARKit

В данном разделе речь пойдет об основных возможностях ARKit, добавленных и функционирующих на момент декабря 2018 года.

Основной функции рассматриваемого SDK, доступной на релизе, была возможность распознавания горизонтальных поверхностей (англ. Plane Detection). Функция обнаружения плоскостей использует карту мира для обнаружения поверхностей, на которых могут быть размещены объекты дополненной реальности. Когда ARKit был запущен, устройство могло распознавать и использовать только горизонтальные плоскости, а такие особенности поверхности, как выпуклости и изгибы, могли легко помешать попыткам точно разместить 3D-объект. На текущей момент существует возможность делать то же самое с вертикальными поверхностями и (в некоторой степени) с поверхностями произвольной формы, которые не являются полностью плоскими.

Так же в ARKit 1.5 добавлено базовое отслеживание 2D-изображений. Это означает, что приложения ARKit могут распознавать что-то вроде страницы в книге, постера фильма или рисунка на стене. Разработчики могут легко заставить свои приложения вводить объекты в окружающую среду, как только устройство распознает эти 2D-изображения.

В новой версии ARKit это функция также распространяется на трехмерные объекты (рис. 5). По сути, способ, которым приложение читает реальный трехмерный объект, похож на способ, которым он строит карты мира. Как и в случае с отслеживанием 2D-изображений, разработчики должны включить в приложение эталонный объект для сравнения с реальным объектом. Потенциальные применения этой функции многочисленны. ARKit может идентифицировать конкретную марку и модель автомобиля в реальном мире и разместить представление названия автомобиля и технические характеристики рядом с его местоположением в поле зрения пользователя.

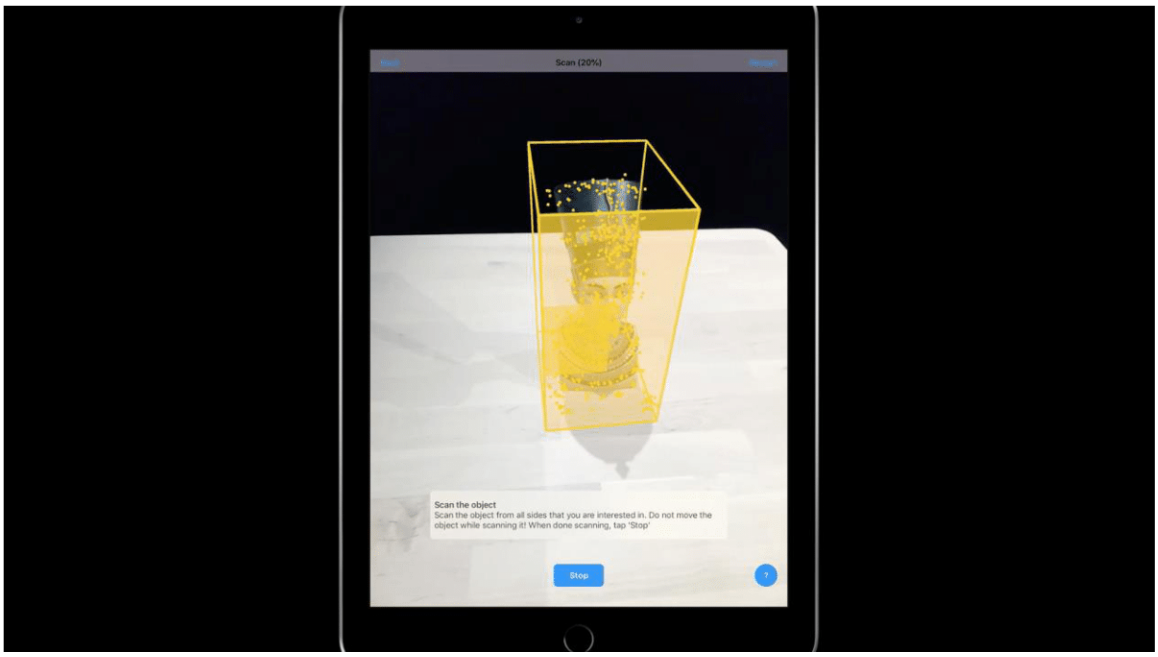

Рисунок 5 – Сканирование объекта

Начиная с версии 2.0 появилась возможность сохранять карту окружения вместе с расставленными объектами дополненной реальности. Имея карту, можно инициализировать с помощью неё AR-сессию, после чего ранее расставленные объекты появятся в нужных местах. Сохранённую карту также можно передать на сервер и использовать на других устройствах.

Механизм сохранения карты окружения позволил синхронизировать систему координат между несколькими устройствами. Зная положение каждого из устройств относительно карты окружения, можно построить многопользовательские сценарии.

Но многопользовательские игры - не единственный возможный вариант использования. Помимо прочего, сохранение и загрузка карт может позволить разработчикам приложений создавать постоянные объекты в определенном месте, например, виртуальную статую на городской площади, которую все пользователи будут видеть в одном и том же месте при каждом посещении. Пользователи могут даже добавлять свои собственные объекты в мир, чтобы их могли найти другие пользователи.

Наконец, ARKit 2 поддерживает расширенное текстурирование среды. Это означает несколько вещей. Во-первых, Apple утверждает, что она

обучила нейронную сеть тысячам сред, что позволяет ARKit по существу галлюцинировать содержимое пробелов в сцене и карте мира с некоторой степенью точности.

Текстурирование среды также позволяет более реалистично отображать объекты в контексте сцены. ARKit отслеживает окружающий свет в окружающей среде и генерирует тени от виртуальных объектов в поле зрения пользователя. Одно это закрывает большой пробел в том, как реальные пользователи могут воспринимать объект; Оказывается, объект, который не отбрасывает тень, портит наши головы.

Он также применяет отражения окружающей среды к объектам, которые должны иметь их.Это еще один пробел для убедительного дополнения.

3 Программная реализация

3.1 Постановка задачи

Программной реализацией будет являться мобильной приложение для операционной системы iOS, способное распознать плоскую поверхность и загрузить на ее виртуальный аналог произвольную 3D-модель, предварительно сохраненных на смартфон посредствам функции iTunes «Общие файлы». Текущая версия приложения загружает модель, сохраненную по пути «/model1/scene.gltf».

3.2 Выбор платформы и языка программирования

Платформой для разработки был выбран игровой движок Unity, благодаря наличию плагина для поддержки формата GLTF, возможности создания AR-приложений, его мультиплатформенности, а также наличию бесплатной версии с достаточным для разработки функционалом и набором инструментов.

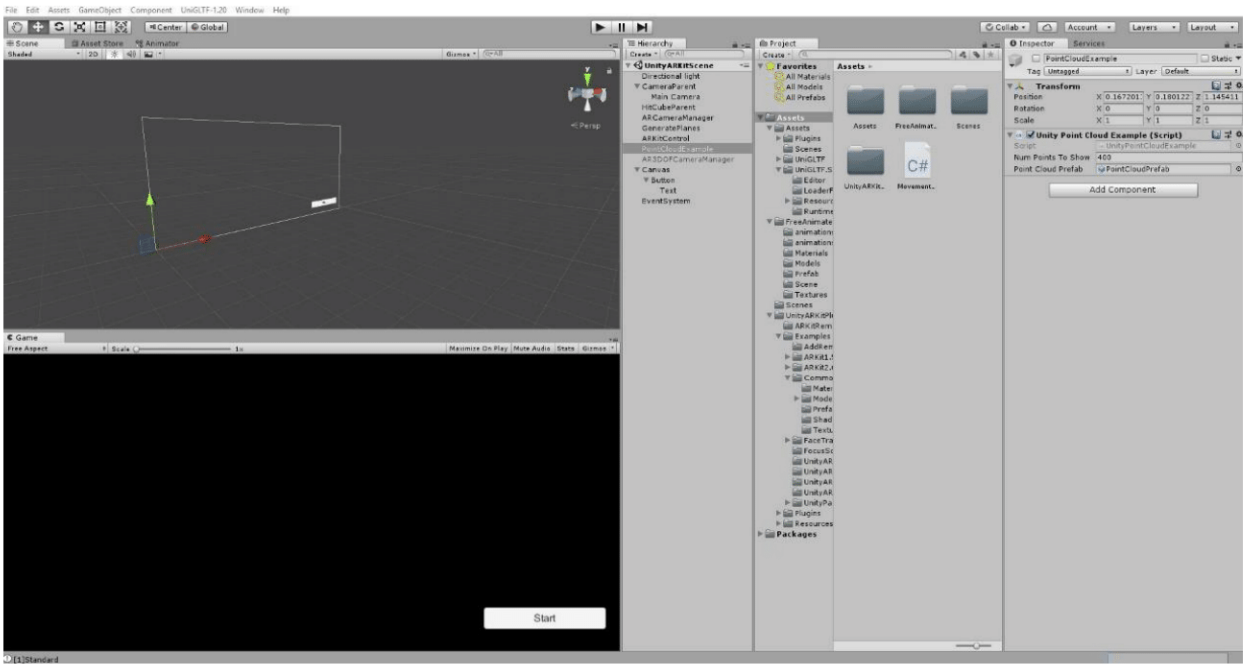

Рисунок 6 – Интерфейс среды Unity

В качестве языка программирования был выбран язык C# как основной скриптовый язык Unity.

3.3 Настройка среды Unity. Установка плагина

Плагин UnityGLTF разделен на две части: решение Visual Studio GLTFSerializer и проект-пример Unity.

Для корректной работы необходимо собрать эту библиотеку в папке Plugins основного проекта Unity, в котором идет разработка. Для этого нужно открыть GLTFSerialization.sln в Microsoft Visual Studio и скомпилировать его с настройкой компилятора Release и параметром командной строки -unsafe. Это поместит двоичные файлы в папку по пути <название основного проекта>\Assets\UnityGLTF\Plugins. После этого плагин будет внедрен в проект и его можно будет использовать.

3.4 Построение внутреигровой сцены

Рассмотрим структуру внутреигровой сцены. На рисунке 7 справа изображена иерархия игровых объектов (англ. game object). Объект Direktional light отвечает за параметры освещения и содержит вектор направления и положение источника света. Набор объектов CameraParent и Main Camera составляют виртуальную внутреигровую камеру. ARCameraManager и AR3DOFCameraManager обеспечивают корректное взаимодействие между аппаратным обеспечением смартфона, таким как цифровая камера и акселерометр, с виртуальной камерой. Объект PointCloud определяет множество точек реальной поверхности, а объект GeneratePlanes строит по ним виртуальную поверхность. Объект Canvas представляет собой холст экрана и содержит объекты интерфейса. EventSystem является стандартным объектом реализует систему событий, таких как нажатие на клавишу пользовательского интерфейса. HitCubeParent является на момент

запуска программы пустым объектом, но после нажатия на кнопку «Start» ему присваивается обьект-потомок Scene, содержащий загруженную 3D-модель.

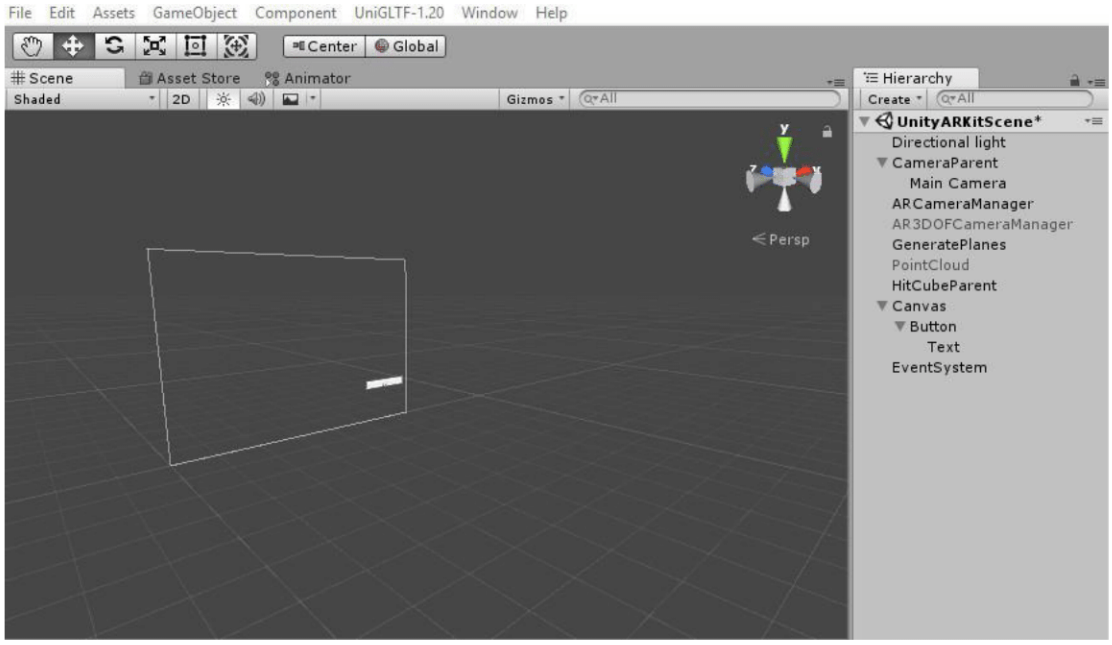

Рисунок 7 – Окно внутреигровой сцены

3.5 Результаты работы

После запуска программа методом визуально-инерциальной одометрии определяет реальную поверхность, строя множество точек (рис. 9). Далее происходит построение виртуальной поверхности (рис. 10). после нажатия на кнопку «Start» происходит загрузка 3D-модели из памяти устройства и создается объект Scene, являющийся потомком HitCubeParent (рис. 8). При этом 3D-модель масштабируется и размещается на созданной виртуальной поверхности, создавая эффект дополненной реальности.

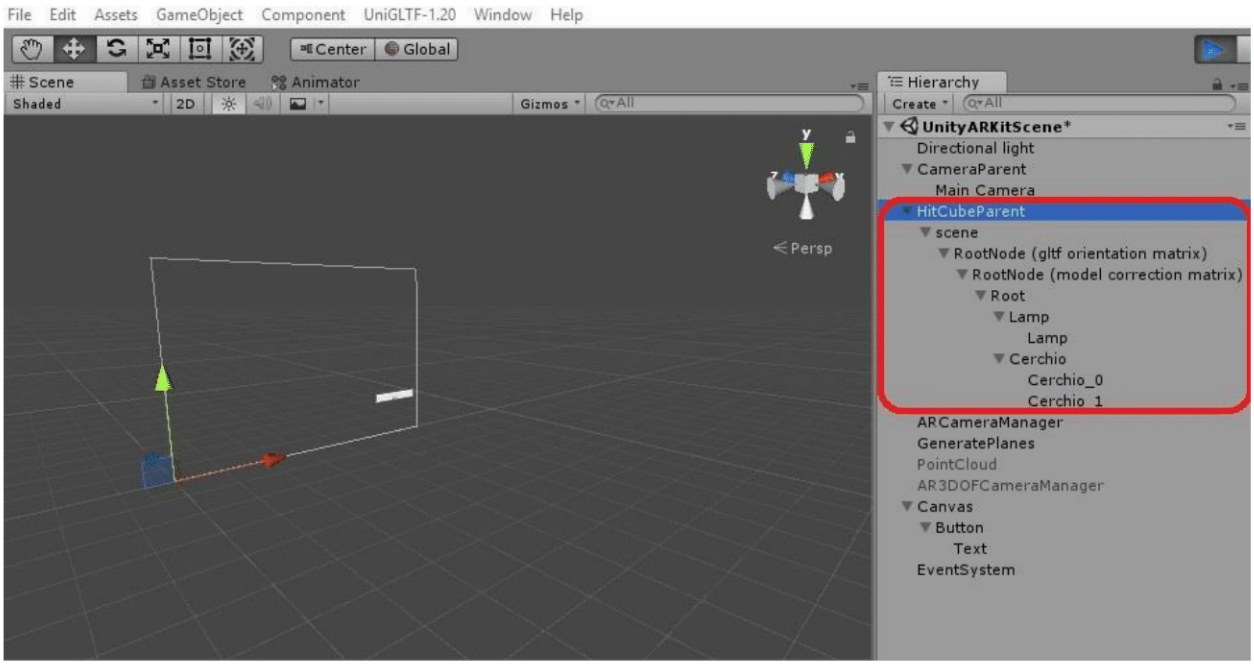

Рисунок 8 – Загрузка модели

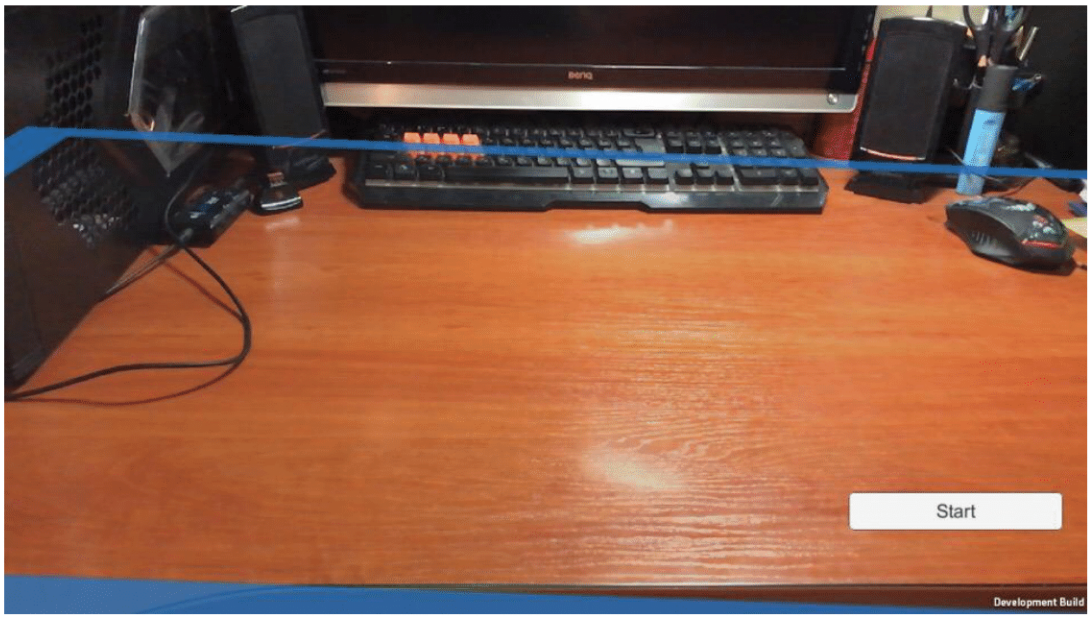

Рисунок 9 – Виртуальная поверхность

Рисунок 10 – Загрузка 3D-модели

ЗАКЛЮЧЕНИЕ

В результате курсовой работы было проведено исследование технологии дополненной реальности и возможностей ее реализации с помощью ARKit, что позволило оценить то, насколько перспективным является это направление. Также был выполнен обзор возможностей игрового движка Unity, его преимуществ и недостатков; рассмотрен формат 3D-моделей GLTF и его особенности.

Основной трудностью, встреченной при сборе информации по теме, была нехватка либо полное отсутствие сведений на русском языке, так как данная область является передовой и быстро развивающейся. Основными источниками были тематические англоязычные порталы, статьи разработчиков и документация ПО.

Результатом работы является разработка мобильного приложения для операционной системы iOS, способное распознать плоскую поверхность и загрузить на нее произвольную 3D-модель, предварительно сохраненных на смартфон посредствам функции iTunes «Общие файлы».

Одним из пунктов плана развития проекта на будущее является разработка подпрограммы с функцией локального файлового менеджера, которая позволит пользователю выбирать из нескольких 3D-моделей, сохраненных им в памяти устройства. На данный момент в приложении пока нет такой возможности, что с тем, что iOS имеет закрытую файловую систему и для реализации подобной опции необходима разработка отдельной подпрограммы. Также в дальнейшем планируется добавление возможности воспроизведения анимации для 3D-модели, изменение размера модели и ее поворот вдоль оси во время работы программы, загрузка файла модели из облака и внедрение синхронизации мира для нескольких пользователей.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Benda P., Ulman M. Augmented Reality As a Working Aid for Intellectually Disabled Persons For Work in Horticulture. URL – https://www.researchgate.net/publication/298033889_Augmented_Reality_As_a_Working_Aid_for_Intellectually_Disabled_Persons_For_Work_in_Horticulture
2. Linietsky Juan. A Small Defense Of GLTF 2.0 On Its Comparison Against Opengex. URL – https://godotengine.org/article/small-defense-gltf
3. Brainville Arthur. Why glTF 2.0 is awesome! URL – https://dev.to/ybalrid/why-gltf-20-is-awesome-2khp
4. GLTF Overview. URL – https://www.khronos.org/gltf/
5. Understanding World Tracking in ARKitURL. – https://developer.apple.com/documentation/arkit/understanding_world_tracking_in_arkit
6. Axon Samuel. How ARKit 2 works, and why Apple is so focused on AR. URL – https://arstechnica.com/gadgets/2018/06/arkit-2-why-apple-keeps-pushing-ar-and-how-it-works-in-ios-12/
7. Ronald T. Azuma. A Survey of Augmented Reality. URL – https://www.cs.unc.edu/~azuma/ARpresence.pdf
8. Linietsky Juan. Why We Should All Support GLTF 2.0 As The Standard Asset Exchange Format For Game Engines. URL – https://godotengine.org/article/we-should-all-use-gltf-20-export-3d-assets-game-engines

ПРИЛОЖЕНИЕ А

Скрипт обработки кнопки старт

```
using System;
using System.IO;
using System.Linq;
using System.Text;
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.XR.iOS;

namespace UniGLTF
{
    public class GuiManager : MonoBehaviour
    {
        [SerializeField]
        Button m_importButton;

        GameObject m_root;

        void Start()
        {
            m_importButton.onClick.AddListener(OnClick);
        }

        public void OnClick()
        {
           #if UNITY_EDITOR
            var path = "C:/Users/Kirse/Documents/Unity3D/bottiglia/scene.gltf";
            #else
            var path = Application.persistentDataPath + "/model1/scene.gltf";
            #endif
            Debug.Log("path");
            if (string.IsNullOrEmpty(path))
            {
                return;
            }

            if (m_root != null)
            {
                GameObject.Destroy(m_root);
            }

            m_root = Load(path);
            m_root.transform.parent = GameObject.Find("HitCubeParent").transform;
            m_root.AddComponent<UnityARHitTestExample>();
            m_root.transform.localPosition = Vector3.zero;
            m_root.transform.localScale = new Vector3(0.05F, 0.05f, 0.05f);
        }

        GameObject Load(string path)
        {
            var bytes = File.ReadAllBytes(path);

            Debug.LogFormat("[OnClick] {0}", path);
            var context = new ImporterContext();

            var ext = Path.GetExtension(path).ToLower();
            switch(ext)
            {
                case ".gltf":
```

```
                    context.ParseJson(Encoding.UTF8.GetString(bytes), new
FileSystemStorage(Path.GetDirectoryName(path)));
                    break;

                case ".zip":
                    {
                        var zipArchive=Zip.ZipArchiveStorage.Parse(bytes);
                        var gltf = zipArchive.Entries.FirstOrDefault(x =>
x.FileName.ToLower().EndsWith(".gltf"));
                        if (gltf == null)
                        {
                            throw new Exception("no gltf in archive");
                        }
                        var jsonBytes = zipArchive.Extract(gltf);
                        var json = Encoding.UTF8.GetString(jsonBytes);
                        context.ParseJson(json, zipArchive);
                    }
                    break;

                case ".glb":
                    context.ParseGlb(bytes);
                    break;

                default:
                    throw new NotImplementedException();
            }

            gltfImporter.Load(context);
            context.Root.name = Path.GetFileNameWithoutExtension(path);
            context.ShowMeshes();

            return context.Root;
        }
    }
}
```

Рисунок А.1 – Листинг фрагмента программы