**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ**

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ**

**«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук

Образовательная программа бакалавриата «Программная инженерия»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| СОГЛАСОВАНО  Научный руководитель,  старший преподаватель ДПИ ФКН  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ М.И. Фомичев  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2020 г. | |  | УТВЕРЖДАЮ  Академический руководитель образовательной программы «Программная инженерия»,  профессор ДПИ ФКН, канд. техн. наук  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.В. Шилов  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2020 г. | |
| |  |  | | --- | --- | | Подп. и дата |  | | Инв. № дубл. |  | | Взам. инв. № |  | | Подп. и дата |  | | Инв. № подл. |  | | **ПРОГРАММА ДЛЯ ПОСТРОЕНИЯ СЕЧЕНИЙ В ДОПОЛНЕННОЙ РЕАЛЬНОСТИ**  **Пояснительная записка**  **ЛИСТ УТВЕРЖДЕНИЯ**  **RU.17701729.04.01-01 81 01-1-ЛУ** | | | | | | |
|  | |  | | | | |
| Исполнитель:  студент группы БПИ191  \_\_\_\_\_\_\_\_\_\_\_\_\_\_/В.И. Беловицкий/  «\_\_» \_\_\_\_\_\_\_\_\_ 2020 г. | | | | |
|  | | | | | | |
|  | | | | |  | |

**Москва 2020**

|  |  |  |
| --- | --- | --- |
| УТВЕРЖДЕН  RU.17701729.04.01-01 81 01-1-ЛУ |  |  |
| |  |  | | --- | --- | | Подп. и дата |  | | Инв. № дубл. |  | | Взам. инв. № |  | | Подп. и дата |  | | Инв. № подл. |  | | **ПРОГРАММА ДЛЯ ПОСТРОЕНИЯ СЕЧЕНИЙ В ДОПОЛНЕННОЙ РЕАЛЬНОСТИ**  **Пояснительная записка**  **RU.17701729.04.01-01 81 01-1**  **Листов 28** | | |
|  | | |

**Москва 2020**

**СОДЕРЖАНИЕ**

[1. ВВЕДЕНИЕ 4](#_Toc40638075)

[1.1. Наименование программы 4](#_Toc40638076)

[1.2. Документы, на основании которых ведется разработка 4](#_Toc40638077)

[2. НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ 5](#_Toc40638078)

[2.1. Назначение программы 5](#_Toc40638079)

[2.1.1. Функциональное назначение 5](#_Toc40638080)

[2.1.2. Эксплуатационное назначение 5](#_Toc40638081)

[2.2. Краткая характеристика области применения 5](#_Toc40638082)

[3. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ 6](#_Toc40638083)

[3.1. Постановка задачи на разработку программы 6](#_Toc40638084)

[3.2. Описание используемого математического аппарата 6](#_Toc40638085)

[3.3. Описание алгоритма и функционирования программы 6](#_Toc40638086)

[3.3.1. Архитектура программы 6](#_Toc40638087)

[3.3.2. Алгоритм Raymarching 6](#_Toc40638088)

[3.3.3. Функционирование дополненной реальности в мобильном приложении 8](#_Toc40638089)

[3.4. Обоснование выбора алгоритма решения задачи 9](#_Toc40638090)

[3.4.1. Существующие аналоги рендеринга трехмерных объектов и преимущества Raymarching 9](#_Toc40638091)

[3.4.2. Обоснование выбора фреймворка Vuforia 9](#_Toc40638092)

[3.5. Описание и обоснование метода организации входных и выходных данных 10](#_Toc40638093)

[3.5.1. Отслеживаемые QR-коды 10](#_Toc40638094)

[3.5.2. Ввод параметров фигуры 10](#_Toc40638095)

[3.5.3. Измененное шейдером выходное изображение 10](#_Toc40638096)

[3.6. Описание и обоснование выбора состава технических и программных средств 10](#_Toc40638097)

[4. ОЖИДАЕМЫЕ ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ 12](#_Toc40638098)

[4.1. Предполагаемая потребность 12](#_Toc40638099)

[4.2. Экономические преимущества разработки по сравнению с отечественными и зарубежными аналогами. 12](#_Toc40638100)

[5. ПЕРСПЕКТИВЫ РАЗВИТИЯ 13](#_Toc40638101)

[СПИСОК ИСТОЧНИКОВ 14](#_Toc40638102)

[ПРИЛОЖЕНИЕ 1 15](#_Toc40638103)

[ПРИЛОЖЕНИЕ 2 16](#_Toc40638104)

[ПРИЛОЖЕНИЕ 3 18](#_Toc40638105)

[ПРИЛОЖЕНИЕ 4 26](#_Toc40638106)

[ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ 28](#_Toc40638107)

1. ВВЕДЕНИЕ
   1. Наименование программы

Наименование программы – RaymarchAR.

Наименование темы разработки – «Приложение для построения сечений в дополненной реальности».

* 1. Документы, на основании которых ведется разработка

Приказ декана факультета компьютерных наук И.В. Аржанцева «Об утверждении тем, руководителей курсовых работ студентов образовательной программы «Программная инженерия» факультета компьютерных наук» № 2.3-02/1112-04 от 11.12.2019 г.

1. НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ
   1. Назначение программы
      1. Функциональное назначение

Функциональным назначением мобильного Android-приложения RaymarchAR является построение сечений, вычитаний, смешиваний и объединений двух или трех геометрических объемных фигур с помощью технологий Raymarching [6] и дополненной реальности [3] с использованием отслеживаемых QR-кодов (Приложение 1).

* + 1. Эксплуатационное назначение

Мобильное Android-приложение RaymarchAR впервые (см. п. 4.2) предоставляет возможность использования технологий Raymarching и дополненной реальности для широкой аудитории пользователей. Ранее сочетание данных технологий было доступно только профессионалам на стационарных ПК, поскольку не было решения для мобильных устройств.

* 1. Краткая характеристика области применения

Приложение используется для задач визуализации пересечения трехмерных объектов, формы которых могут быть заданы в виде шара, куба, тора или призмы. Данные задачи актуальны в сферах технологий, науки, искусства, материаловедения, образования. Предлагаемое решение является дешевым экспресс-методом, не требующим мощных ресурсов, в случае практического применения в производственной практике. В образовательных целях реализуемые возможности визуализации способствуют развитию пространственного мышления, как в области точных наук, так и в сфере искусства.

1. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ
   1. Постановка задачи на разработку программы

Программа должна обеспечивать возможность выполнения перечисленных ниже функций:

* выбор типа фигуры для каждого QR-кода: сфера, куб, тор, призма;
* выбор операции для фигуры, связанной с определенным QR-кодом: объединение, вычитание, пересечение, смешивание;
* выбор цвета фигуры;
* выбор силы смешивания фигур;
* ввод вещественных чисел для определения размера фигуры;
* возможность перемещать фигуры в дополненной реальности посредством перемещения QR-кодов в реальном трехмерном пространстве;
* визуализация выбранных операций над фигурами в дополненной реальности;
* возможность удалять все фигуры с экрана посредством физического закрытия камеры устройства.
  1. Описание используемого математического аппарата

В качестве алгоритма, решающего основную задачу (построение сечений, вычитаний, смешиваний и объединений) был выбран алгоритм «марширующего луча» – Raymarching. Данный алгоритм позволяет эффективно визуализировать вышеперечисленные геометрические объекты. Подробнее алгоритм рассмотрен в п. 3.3.2, 3.4.1.

* 1. Описание алгоритма и функционирования программы
     1. Архитектура программы

Описания классов, структур, перечислений и шейдеров представлено в приложении 2 (табл. 1–4).

Описание полей, свойств и методов классов представлено в приложении 3 (табл. 5–30).

Описание основного вычислительного шейдера представлено в приложении 4 (табл. 31–33).

* + 1. Алгоритм Raymarching

Raymarching – алгоритм рендеринга, позволяющий визуализировать неявные геометрические объекты, для которых не существует конечной функции поверхности или ее вычисление является очень сложным. Raymarching основан на Signed Distance Functions (SDF) [4] – функциях, возвращающих минимальное расстояние от точки до объекта. Все объекты на сцене задаются своими SDF. Возвращаемое значение положительное, если точка находится за пределами объекта, или отрицательное, если точка находится внутри объекта. Большинство геометрических трехмерных объектов можно задать с помощью SDF или комбинации нескольких SDF примитивных объектов.

Алгоритм заключается в том, что из каждой точки (пикселя) изображения, получаемого камерой, выпускается луч, для которого запускается следующий цикл:

1. от точки начала луча или окончания последнего отложенного вектора (на второй итерации и далее) рассчитывается расстояние до сцены, которое определяется заданными операциями над фигурами и их SDF;
2. если расстояние больше заданного значения эпсилон, то на луче откладывается вектор, длина которого равна полученному расстоянию, и цикл повторяется с шага 1;
3. иначе, если расстояние до сцены меньше или равно эпсилон, считается, что луч пересек сцену и для данной точки (пикселя), из которой был выпущен луч, возвращается цвет объекта, в который попал луч.

Алгоритм отрабатывает для всех точек исходного изображения, формируя новое изображение на выходе.

Рассмотрим работу алгоритма на конкретном примере (рис. 1). В данном случае сцена состоит из пересечения синего куба и красной сферы. Для куба и сферы определены SDF (ниже приведен код на языке HLSL):

float SphereSDF(float3 p, float3 centre, float radius)

{

return distance(p, centre) - radius;

}

float CubeSDF(float3 p, float3 centre, float3 size)

{

float3 o = abs(p - centre) - size;

float ud = length(max(o, 0));

float n = max(max(min(o.x, 0), min(o.y, 0)), min(o.z, 0));

return ud + n;

}

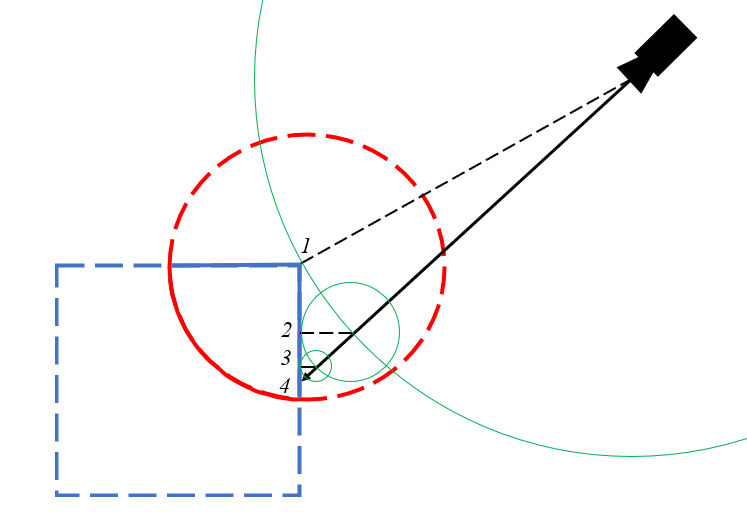


Рисунок 1 – Двумерная визуализация работы алгоритма Raymarching для сцены, состоящей из пересечения синего куба и красной сферы. Цифрами *1*–*4* отмечены шаги алгоритма

Так как куб и сфера образуют пересечение, на каждом шаге берется максимальное возвращенное двумя SDF значение [5]. На шаге *1* (см. рис. 1) таким максимальным значением становится расстояние до куба. На луче откладывается вектор длины, равной найденному на шаге *1* расстоянию, и поиск расстояний продолжается из точки окончания вектора.

На шаге *4* расстояние до куба становится меньше эпсилон и в точку экрана, из которой был выпущен луч, возвращается цвет куба. По этому принципу алгоритм отрабатывает для всех точек текущего изображения, и на выходе отображается часть куба, пересеченная сферой.

Можно заметить, что чем ближе луч проходит к границе геометрического объекта, тем больше шагов (итераций) необходимо совершить [6]. Для этого необходимо ограничить число шагов алгоритма и не возвращать новый цвет (т. е. оставлять исходный цвет изображения) в конкретную точку.

* + 1. Функционирование дополненной реальности в мобильном приложении

В качестве инструмента для работы с дополненной реальностью был выбран фреймворк Vuforia [8], основанный на отслеживании распечатанных изображений (QR-кодов) в реальном мире камерой устройства.

Каждый QR-код привязан к игровому объекту, хранящему параметры фигуры. Когда в поле зрения камеры мобильного устройства находится хотя бы один отслеживаемый QR-код, параметры фигуры передаются в вычислительный шейдер (табл. 31–33) и происходит преобразование изображения шейдером.

* 1. Обоснование выбора алгоритма решения задачи
     1. Существующие аналоги рендеринга трехмерных объектов и преимущества Raymarching

На данный момент наиболее распространенным алгоритмом рендеринга трехмерных объектов является представление объекта в виде полигональной сетки, т. е. множества вершин, ребер и граней (полигонов) [7].

Алгоритм рендеринга, использующий полигональную сетку, эффективен, когда объекты заданы явно, т. е. объект был заранее определен, и все изменения над сеткой явно описываются в программе. Для решения задач построения произвольного сечения, вычитания, объединения или смешивания объектов полигональная сетка не подходит.

Использованный алгоритм Raymarching не зависит от заданной поверхности конкретного объекта и не требует построения новой сетки для искомого объекта, поскольку он использует SDF для преобразования исходного изображение посредством изменения цвета пикселей.

* + 1. Обоснование выбора фреймворка Vuforia

Для реализации дополненной реальности с учетом требований об используемых языке программирования C# и среде разработки Unity существует два основных инструмента: AR Foundation [1] и Vuforia [8].

AR Foundation является нативным фреймворком для работы с дополненной реальностью в Unity. Для полноценной работы с AR Foundation необходима поддержка мобильным устройством платформы ARCore [2], но перечень поддерживающих ее устройств очень ограничен. Именно это послужило причиной отказа от использования AR Foundation, поскольку устройство (Xiaomi Redmi 4X), на котором велось тестирование приложения RaymarchAR, не входит в список поддерживаемых устройств ARCore.

Vuforia – фреймворк для работы с дополненной реальностью, официально интегрированный в Unity в 2019 г. Основывается на распознавании двумерных изображений (распечатанных на бумаге) или изображений, расположенных на определенном трехмерном объекте в реальном мире. Выбор был сделан в пользу Vuforia, поскольку данный фреймворк поддерживается на большинстве Android-устройств. Необходимость распечатывать QR-коды, которую можно было бы оценить как недостаток Vuforia, в данном случае не принципиальна.

* 1. Описание и обоснование метода организации входных и выходных данных
     1. Отслеживаемые QR-коды

В качестве входных данных для работы дополненной реальности выступают распечатанные на бумаге QR-коды, размер изображения 7 × 7 см. Пример QR-кода представлен на рис. 2.



Рисунок 2 – QR-код для первой фигуры

Каждый QR-код может отслеживаться скриптами Vuforia. В программе QR-код присвоен игровому объекту, определенному скриптами Vuforia, который хранит в дочернем объекте информацию о фигуре.

* + 1. Ввод параметров фигуры

Ввод параметров фигуры выполняется пользователем в интерфейсе программы. Входные данные (тип фигуры, операция над фигурой, цвет, сила смешивания и размер фигуры) выбираются пользователем, валидируются и устанавливаются в объект фигуры.

* + 1. Измененное шейдером выходное изображение

В качестве основного результата работы приложения пользователь получает изображение, измененное алгоритмом Raymarching. Исходным изображением служит изображение, полученное камерой мобильного устройства, в области видимости которой расположены QR-коды фигур.

* 1. Описание и обоснование выбора состава технических и программных средств

Минимальный состав технических и программных средств для надежной и бесперебойной работы программы следующий:

* мобильное устройство на базе операционной системы Android версии не менее 7.0 Nougat (API level 24);
* процессор Snapdragon 435 или выше;
* основная камера мобильного устройства с разрешением не менее 2,0 Мп;
* 1,0 гигабайт (ГБ) оперативной памяти;
* 0,3 гигабайта (ГБ) пространства на постоянном запоминающем устройстве;
* струйный или лазерный принтер;
* программа для работы с принтером.

Разработка программы проводилась с помощью следующих программных средств:

* операционная система Microsoft Windows 10;
* установленный Microsoft .NET Framework 4.0;
* Unity 2019.3.0a8 (64-bit) или выше;
* Unity Hub;
* Vuforia SDK.

1. ОЖИДАЕМЫЕ ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ
   1. Предполагаемая потребность

Разработанное Android-приложение RaymarchAR является первым программным решением для мобильных устройств, демонстрирующим использование технологии Raymarching в сочетании с дополненной реальностью.

До сих пор данная технология применялась для ресурсоемких программ, работающих на стационарных ПК, в сферах дизайна, анимации и разработки игр. Отсутствие бюджетных решений ограничивало круг начинающих программистов, способных заинтересоваться данной темой. Приложение RaymarchAR открывает возможности для демонстрации технологии Raymarching всем желающим, что, несомненно, поспособствует развитию этой технологии в перспективе.

Задачи визуализации пересечения трехмерных объектов могут быть востребованы в сферах технологий, науки, искусства, материаловедения, образования.

* 1. Экономические преимущества разработки по сравнению с отечественными и зарубежными аналогами.

В период разработки на рынке не было выявлено аналогичных продуктов.

1. ПЕРСПЕКТИВЫ РАЗВИТИЯ

Перспективы развития мобильного Android-приложение RaymarchAR связаны в первую очередь с расширением функционала: добавление новых типов фигур и операций над ними. Приложение сможет визуализировать фракталы, демонстрировать преобразование форм фигур с течением времени (движение), поворачивать фигуры на заданный угол, строить бесконечные повторения одинаковых объектов и др.

СПИСОК ИСТОЧНИКОВ

1. About AR Foundation [Электронный ресурс]. URL: https://docs.unity3d.com/Packages/com.unity.xr.arfoundation@3.1/manual/index.html (дата обращения: 14.05.2020).

2. ARCore. Unity Overview [Электронный ресурс]. URL: https://developers.google.com/ar/develop/unity (дата обращения: 14.05.2020).

3. Milgram P., Kishino F. A Taxonomy of Mixed Reality Visual Displays // IEICE Transactions on Information Systems. 1994, vol. E77D, № 12, pp. 1321–1329.

4. Osher S., Fedkiw R. Level Set Methods and Dynamic Implicit Surfaces. New York: Springer, 2003. 273 p.

5. Quilez I. Distance Functions. Primitive Combinations [Электронный ресурс]. URL: https://iquilezles.org/www/articles/distfunctions/distfunctions.htm (дата обращения: 14.05.2020).

6. Quilez I. Rendering Worlds with Two Triangleswith Raytracing on the GPUin 4096 Bytes [Электронный ресурс]. URL: https://iquilezles.org/www/material/nvscene2008/rwwtt.pdf (дата обращения: 14.05.2020).

7. Smith C. On Vertex-Vertex Systems and Their Use in Geometric and Biological Modelling. PhD (Comp. Sci.) Dissertation. Calgary: University of Calgary, 2006. 204 p.

8. Vuforia. Getting Started with Vuforia Engine in Unity [Электронный ресурс]. URL: https://library.vuforia.com/articles/Training/getting-started-with-vuforia-in-unity.html (дата обращения: 14.05.2020).

ПРИЛОЖЕНИЕ 1

ТЕРМИНОЛОГИЯ

Вычислительный шейдер – программа, запущенная на видеокарте за пределами территории обычного рендеринга.

Дополненная реальность – реальный мир, который дополняется виртуальными элементами и сенсорными данными.

Игровой объект – основная сущность в Unity (GameObject).

Скрипт – программа на языке C#, определяющая логику для игрового объекта.

Сцена – пространство для размещения игровых объектов в Unity.

Фреймворк – программная платформа, определяющая структуру программной системы; программное обеспечение, облегчающее разработку и объединение разных компонентов большого программного проекта.

QR-код (от *Quick Response*) – двухмерный штрихкод (бар-код), предоставляющий информацию для быстрого ее распознавания с помощью камеры на мобильном телефоне.

ПРИЛОЖЕНИЕ 2

ОПИСАНИЕ КЛАССОВ

Таблица 1 – Описание классов

|  |  |
| --- | --- |
| Класс | Назначение |
| ComputeRayScript | Основной скрипт для распознавания наблюдаемых QR-кодов, создания вычислительного буфера, передачи параметров в шейдер и преобразования изображения |
| RayShape | Скрипт, хранящий параметры фигуры |
| ShapePanelManager | Скрипт, наполняющий параметрами фигуры панель для управления параметрами фигуры |
| ShapePanelOpener | Скрипт для кнопки, отвечающий за открытие панели для управления параметрами фигуры |
| ShapeTypeDrop | Скрипт для работы с выпадающим списком типов фигуры |
| ShapeOperationDrop | Скрипт для работы с выпадающим списком операции для фигуры |
| ColorSlider | Скрипт, отвечающий за слайдер для настройки цвета фигуры |
| BlendSlider | Скрипт, отвечающий за слайдер для настройки силы смешивания фигуры |
| SizePanelManager | Скрипт, наполняющий значениями размера фигуры поля для ввода под управлением SizeInput |
| SizeInput | Скрипт, отвечающий за корректный ввод размера фигуры |
| InfoPanelOpener | Скрипт для кнопки, отвечающий за открытие панели с информацией |
| LinkOpener | Скрипт для кнопки, отвечающий за открытие ссылки в браузере |

Таблица 2 – Описание структур

|  |  |
| --- | --- |
| Структура | Назначение |
| ShapeData | Структура, хранящая параметры фигуры для передачи данных в шейдер |

Таблица 3 – Описание перечислений

|  |  |
| --- | --- |
| Перечислений | Назначение |
| ShapeType | Типы фигур |
| Operation | Операции над фигурами |

Таблица 4 – Описание шейдеров

|  |  |
| --- | --- |
| Шейдер | Назначение |
| ComputeRayShader | Вычислительный шейдер для графического чипа, реализующий алгоритм Raymarching |

ПРИЛОЖЕНИЕ 3

ОПИСАНИЕ ПОЛЕЙ, СВОЙСТВ И МЕТОДОВ КЛАССОВ

Таблица 5 – Поля класса ComputeRayScript

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Модификатор доступа | Тип | Назначение |
| raymarchShader | public | ComputeShader | Основной  вычислительный шейдер |
| threadDelimeter | private | float | Число для подсчета групп потоков для шейдера |
| target | private | RenderTexture | Текстура, получаемая на выходе работы шейдера |
| camera | public | Camera | Основная камера |
| lightSource | public | Light | Источник света |
| disposeBuffer | private | ComputeBuffer | Ссылка для хранения и высвобождения вычислительного буфера |
| shapes | private | List<RayShape> | Список фигур, отслеживаемых в текущем кадре |

Таблица 6 – Методы класса ComputeRayScript

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название | Модификатор доступа | Тип | Аргументы | Назначение |
| Awake | private | void | – | Устанавливает разрешение экрана при загрузке скрипта |
| DetectTrackables | private | void | – | Определяет отслеживаемые в текущем кадре QR-коды и наполняет shapes |
| OnRenderImage | private | void | RenderTexture source, RenderTexture destination | Если были найдены отслеживаемые QR-коды, инициализирует шейдер |
| FillBufferShapes | private | void | – | Создает экземпляры ShapeData и наполняет ими вычислительный буфер |
| SetCameraAndLightParameters | private | void | – | Устанавливает параметры камеры и света в шейдер |
| InitTargetTexture | private | void | – | Инициализирует текстуру, изменяемую шейдером |

Таблица 7 – Поля класса RayShape

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Модификатор доступа | Тип | Назначение |
| \_shapeType | public | ShapeType | Тип фигуры. По умолчанию сфера |
| operation | public | Operation | Операция для фигуры |
| color | public | Color | Цвет фигуры |
| blendStrength | public | float | Сила смешивания фигуры |
| scaleWasChangedManually | private | bool | Определяет, был ли изменен размер вручную |
| scale | public | Vector3 | Размер фигуры |

Таблица 8 – Свойства класса RayShape

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Модификатор доступа | Тип | Назначение |
| shapeType | public | ShapeType | Тип фигуры |
| Position | public | Vector3 | Координаты фигуры |
| Scale | public | Vector3 | Размер фигуры. Зависит от того, изменялся ли scale вручную |

Таблица 9 – Методы класса RayShape

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название | Модификатор доступа | Тип | Аргументы | Назначение |
| GetScale | private | Vector3 | – | Возвращает размер по умолчанию в зависимости от типа фигуры |

Таблица 10 – Поля класса ShapePanelManager

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Модификатор доступа | Тип | Назначение |
| Panel | private | GameObject | Панель с параметрами фигуры |
| shapes | private | List<RayShape> | Список всех фигур |
| typeDrop | private | GameObject | Выпадающий список типов фигуры |
| operationDrop | private | GameObject | Выпадающий список операций для фигуры |
| colorSlider | private | GameObject | Слайдер для выбора цвета фигуры |
| blendSlider | private | GameObject | Слайдер для выбора силы смешивания фигуры |
| sizePanel | private | GameObject | Панель с полями для ввода размера фигуры |

Таблица 11 – Свойства класса ShapePanelManager

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Модификатор доступа | Тип | Назначение |
| Index | public | int | Индекс текущей выбранной фигуры |

Таблица 12 – Методы класса ShapePanelManager

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название | Модификатор доступа | Тип | Аргументы | Назначение |
| InitElements | private | void | – | Инициализирует поля – элементы интерфейса |
| GetAllTrackables | public | void | – | Определяет все отслеживаемые QR-коды и наполняет shapes |
| SetData | public | void | int index | Наполняет панель параметрами фигуры с заданным индексом |

Таблица 13 – Поля класса ShapePanelOpener

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Модификатор доступа | Тип | Назначение |
| ShapePanel | public | GameObject | Панель с параметрами фигуры |
| shapeIndex | public | int | Индекс фигуры и данной кнопки |

Таблица 14 – Методы класса ShapePanelOpener

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название | Модификатор доступа | Тип | Аргументы | Назначение |
| TogglePanel | public | void | – | Открывает или закрывает панель с данным index |

Таблица 15 – Поля класса ShapeTypeDrop

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Модификатор доступа | Тип | Назначение |
| dropdown | private | Dropdown | Выпадающий список |
| currentShape | private | RayShape | Текущая фигура |
| sizePanel | private | SizePanelManager | Панель с полями ввода размера фигуры |

Таблица 16 – Методы класса ShapeTypeDrop

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название | Модификатор доступа | Тип | Аргументы | Назначение |
| Init | private | void | – | Инициализирует выпадающий список |
| OnClickHandler | public | void | Dropdown drop | Обработчик нажатия на элемент списка |
| SetOptions | public | void | RayShape shape | Устанавливает тип фигуры в список |
| SetSizePanel | public | void | SizePanelManager spm | Присваивает ссылку полю sizePanel |

Таблица 17 – Поля класса ShapeOperationDrop

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Модификатор доступа | Тип | Назначение |
| dropdown | private | Dropdown | Выпадающий список |
| currentShape | private | RayShape | Текущая фигура |

Таблица 18 – Методы класса ShapeOperationDrop

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название | Модификатор доступа | Тип | Аргументы | Назначение |
| Init | private | void | – | Инициализирует выпадающий список |
| OnClickHandler | public | void | Dropdown drop | Обработчик нажатия на элемент списка |
| SetOptions | public | void | RayShape shape | Устанавливает операцию для фигуры в список |

Таблица 19 – Поля класса ColorSlider

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Модификатор доступа | Тип | Назначение |
| slider | public | Slider | Слайдер |
| sliderHandle | public | Image | Изображение ручки слайдера |
| sliderBackground | public | Image | Изображение фона слайдера |
| colorTexture | private | Texture2D | Цветная текстура для фона слайдера |
| currentShape | private | RayShape | Текущая фигура |

Таблица 20 – Методы класса ColorSlider

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название | Модификатор доступа | Тип | Аргументы | Назначение |
| Awake | private | void | – | Инициализирует внешний вид слайдера |
| SetData | public | void | RayShape shape | Устанавливает значение слайдера равным цвету фигуры |
| OnValueChanged | private | void | float value | Обработчик изменения значения слайдера |
| CreateColorTexture | private | Texture2D | int density | Создает цветную текстуру в диапазоне цветов Hue |

Таблица 21 – Поля класса BlendSlider

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Модификатор доступа | Тип | Назначение |
| slider | public | Slider | Слайдер |
| currentShape | private | RayShape | Текущая фигура |

Таблица 22 – Методы класса BlendSlider

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название | Модификатор доступа | Тип | Аргументы | Назначение |
| Awake | private | void | – | Добавляет обработчик изменения значения |
| SetData | public | void | RayShape shape | Устанавливает значение слайдера равным силе смешивания фигуры |
| OnValueChanged | private | void | float value | Обработчик изменения значения слайдера |

Таблица 23 – Поля класса SizePanelManager

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Модификатор доступа | Тип | Назначение |
| Panel | private | Panel | Панель с тремя полями для ввода SizeInput |
| currentShape | private | RayShape | Текущая фигура |

Таблица 24 – Методы класса SizePanelManager

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название | Модификатор доступа | Тип | Аргументы | Назначение |
| Init | private | void | – | Инициализирует панель |
| SetData | public | void | RayShape shape | Находит все объекты типа SizeInput и устанавливает значение равное размеру фигуры |

Таблица 25 – Поля класса SizeInput

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Модификатор доступа | Тип | Назначение |
| Index | public | int | Индекс части размера |
| input | private | InputField | Поле для ввода вещественного числа |
| currentShape | private | RayShape | Текущая фигура |

Таблица 26 – Методы класса SizeInput

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название | Модификатор доступа | Тип | Аргументы | Назначение |
| Init | private | void | – | Инициализирует поле для ввода |
| SetValue | public | void | RayShape shape | Устанавливает значение части размера фигуры |
| ChangeScale | private | void | String value | Валидирует введенное значение и изменяет размер фигуры |

Таблица 27 – Поля класса InfoPanelOpener

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Модификатор доступа | Тип | Назначение |
| InfoPanel | public | GameObject | Информационная панель |

Таблица 28 – Методы класса InfoPanelOpener

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название | Модификатор доступа | Тип | Аргументы | Назначение |
| TogglePanel | public | void | – | Открывает или закрывает информационную панель |

Таблица 29 – Поля класса LinkOpener

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Модификатор доступа | Тип | Назначение |
| link | public | string | ссылка на сайт |

Таблица 30 – Методы класса LinkOpener

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название | Модификатор доступа | Тип | Аргументы | Назначение |
| OpenLink | public | void | – | Открывает ссылку в Интернет-браузере по умолчанию |

ПРИЛОЖЕНИЕ 4

ОПИСАНИЕ СТРУКТУР, ПОЛЕЙ И МЕТОДОВ ШЕЙДЕРОВ

Таблица 31 – Структуры шейдера ComputeRayShader

|  |  |
| --- | --- |
| Название | Назначение |
| Shape | Структура, хранящая параметры фигуры |
| Ray | Структура, хранящая параметры луча |

Таблица 32 – Поля шейдера ComputeRayShader

|  |  |  |
| --- | --- | --- |
| Название | Тип | Назначение |
| Source | Texture2D<float4> | Исходная текстура |
| Destination | RWTexture2D<float4> | Выходная текстура |
| \_CamToWorld | float4x4 | Матрица, преобразующая пространство камеры в пространство мира |
| \_CamInverseProjection | float4x4 | Матрица проекции камеры |
| \_Light | float3 | Вектор (точка) света |
| maxDst | float | Максимальная дистанция для пути луча |
| maxSteps | float | Максимальное количество шагов для луча |
| epsilon | float | Маленькое число для сравнения двух вещественных чисел |
| shadowBias | float | Сдвиг тени |
| shapes | StructuredBuffer<Shape> | Буфер, хранящий список переданных фигур |
| shapesNumber | int | Количество переданных фигур |

Таблица 33 – Методы шейдера ComputeRayShader

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Тип | Аргументы | Назначение |
| SphereSDF | float | float3 p, float3 center, float radius | SDF сферы |
| CubeSDF | float | float3 p, float3 center, float3 size | SDF куба |
| TorusSDF | float | float3, float3 center, float r1, float r2 | SDF тора |
| PrismSDF | float | float3 p, float center, float h | SDF призмы |
| CreateRay | Ray | float3 origin, float3 direction | Создает луч из заданной точки в заданном направлении |
| CreateCameraRay | Ray | float2 uv | Создает луч из текущей точки изображения |
| Blend | float4 | float a, float b, float3 colorA, float3 colorB, float k | Смешивает две фигуры на основе расстояния до них, учитывая цвет |
| Combine | Float4 | float dstA, float dstB, float3 colorA, float3 colorB, int operation, float blendStrength | Применяет операцию к двум фигурам и возвращает результат – цвет и расстояние |
| GetShapeDst | float | Shape shape, float3 p | Вычисляет расстояние от точки до фигуры в зависимости от ее типа |
| SceneDst | float4 | float3 p | Вычисляет расстояние от точки до всей сцены |
| CalculateNomal | float3 | float3 p | Вычисляет нормаль |
| CalculateShadow | float | Ray ray, float dstToShadePoint | Вычисляет тень |
| RayMarching | void | uint id, uint width, uint height | Запускает алгоритм Raymarching для каждой точки исходного изображения |
| CSMain | void | uint id | Основной метод вычислительного шейдера |

ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Изм. | Номера листов (страниц) | | | | Всего листов (страниц) в документе | № документа | Входящий № сопрово-дитель-ного документа и дата | Подпись | Дата |
| измененных | замененных | новых | аннулированных |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |