Санкт-Петербургский государственный университет

Кафедра системного программирования

Группа 22.Б07-мм

Создание 3D игры в жанре survival horror на Unity Engine

Лодыгин Леонид Александрович

Отчёт по учебной практике
в форме «Производственное задание»

Научный руководитель:
старший преподаватель кафедры системного программирования, к. т. н., Ю. В. Литвинов

Санкт-Петербург
2023

Оглавление

Введение **3**

1. Постановка задачи **4**

2. Обзор **5**

2.1. Терминология 5

2.2. Обзор аналогов 6

2.2.1. Resident Evil 7: Biohazard 6

2.2.2. Call of Cthulhu: Dark Corners of the Earth 6

2.3. Существующие технологии 7

2.3.1. Unity Engine 7

2.3.2. Unreal Engine 8

2.4. Использованные технологии 8

3. Детали реализации **10**

3.1. Концепт игры 10

3.2. Модель главного героя 10

3.3. Передвижение игрока 11

3.4. Оружие 13

3.5. Система психического состояния 13

3.6. Система способностей 14

Заключение **16**

Список литературы **17**

Введение

Игровая индустрия, зародившаяся в 70-х годах прошлого века, за несколько десятилетий выросла в огромный самостоятельный пласт в разработке программного обеспечения. Миллионы пользователей по всему миру, десятки тысяч программистов, работающих в этой отрасли, как в больших компаниях, так и самостоятельно. Разработка игр является востребованным направлением для будущих специалистов.

Survival horror является одним из популярных жанров для видеоигр. Как правило, игрок в таких играх исследует локацию, собирая попутно необходимые для выживания ресурсы. Открытые бои с противниками, по возможности, избегаются.

Небольшие игровые компании и самостоятельные разработчики для создания игр, как правило, используют готовые игровые движки, обеспечивающие игру соблюдением законов физики, графической визуализацией, взаимодействием между объектами и многим другим. Одним из таких движков является *Unity Engine*.

Изучение возможностей движка путём создания видеоигры позволяет получить большой практический опыт, лучше понять все этапы этого процесса, начиная от проектирования, заканчивая тестированием и оптимизацией, столкнуться с массой различных аспектов движка, например, работой с графикой, анимацией, звуком.

1. Постановка задачи

Целью данной работы является изучение возможностей игрового движка Unity путем командной разработки 3D игры в жанре survival horror. Для её выполнения были поставлены следующие задачи:

1. разработать концепт игры;
2. провести обзор игр аналогичного жанра;
3. изучить документацию Unity Engine;
4. разработать следующие игровые механики:
    - Передвижение игрока;
    - Оружие и его анимации;
    - Система психического состояния игрока;
    - Система способностей игрока.
5. провести апробацию реализованных механик;

Концепт игры разрабатывался с участием Плоскарева В.В, затем он приступил к отдельной задаче по реализации системы искуственного интеллекта для противников.

2. Обзор

2.1. Терминология

Игровой движок — программное обеспечение, предназначенное для разработки и создания видеоигр. Он предоставляет набор инструментов и функций, которые облегчают процесс разработки игр, включая управление графикой, звуком, физикой, искусственным интеллектом, вводом пользователя и многими другими аспектами игрового процесса.

Ассет — любой ресурс, который можно использовать в проекте. Это может включать в себя изображения, звуки, 3D-модели, текстуры, анимации, скрипты, материалы, шейдеры и многое другое. В общем смысле, ассеты представляют собой файлы и ресурсы, которые нужны для создания и развития игры или приложения.

Префаб — шаблон объекта, который можно создать, настроить и затем использовать многократно в разных частях игры без необходимости создавать его снова.

Компонент — модульный блок функциональности, который можно добавить к игровым объектам для определения их поведения или характеристик.

Рейкаст (`Raycast`) — техника, которая используется для определения того, на что направлен луч в трехмерном или двухмерном пространстве. Рейкаст создает луч и проверяет, пересекается ли он с каким-либо объектом на сцене. Эта техника часто применяется для решения различных задач, таких как обработка пользовательских взаимодействий, определение столкновений, определение видимости объектов и т.д.

Скелет (`Rig`) — система костей и контрольных объектов, которые привязываются к 3D-модели, чтобы управлять её движением и анимацией.

Коллайдер — компонента, которая используется для определения границ и обнаружения столкновений между объектами в трехмерном или двухмерном пространстве. Коллайдеры играют важную роль в физическом взаимодействии объектов.

2.2. Обзор аналогов

В качестве аналогов выбирались игры данного жанра со схожим концептом. Обзор таких игр помог выявить интересные механики и удачные решения в визуальной составляющей.

2.2.1. Resident Evil 7: Biohazard

Компьютерная игра в жанре `survival horror`, разработанная и изданная японской компанией `Capcom`. Особенности игры включают в себя:

1. Перспективу от первого лица. В отличие от предыдущих частей, где камера была установлена в третьем лице, `Resident Evil 7` [10] предлагает перспективу от первого лица, что придает игроку более сильное пугающее ощущение;

2. Красивую и продуманную визуальную составляющую, например, покачивание оружия в руке главного героя;

3. Фокус на выживании — игроку приходится исследовать поместье, решать головоломки и бороться за выживание.

2.2.2. Call of Cthulhu: Dark Corners of the Earth

Игра в стиле `Survival horror` [8] от `Headfirst Productions`, изданная `Bethesda Softworks`. Основанная на произведениях Г.Ф. Лавкрафта, особенно на его романе «Тень над Иннсмутом» (`The Shadow over Innsmouth`), игра предлагает игрокам погрузиться в мрачный и атмосферный мир Лавкрафтовского ужаса. Основные черты игры:

1. смешение стилей игры. В игре присутствуют элементы шутера от первого лица, детективной работы и выживания. Игрокам предстоит исследовать окружающий мир, решать головоломки и бороться за выживание в мире, наполненном тайнами и ужасами;

2. система уровня страха. В игре присутствует уникальная система уровня страха, которая влияет на поведение персонажа в зависимости от того, насколько страшным было прошлое событие;

3. сюжет и атмосфера. Игра рассказывает историю детектива Джека Уолтерса, который расследует загадочные события в небольшом городе Иннсмут. Сюжет тесно связан с мифологией Ктулху и других творений Лавкрафта, предоставляя игрокам уникальную возможность погрузиться в мир космического ужаса.

2.3. Существующие технологии

Для того, чтобы каждый раз не воссоздавать физическое моделирование, визуализацию, взаимодействие между объектами и многие другие аспекты, присущие всем играм, были разработаны игровые движки, позволяющие сосредоточить свое внимание на реализации уже конкретных механик игры. Самыми популярными среди движков, поддерживающих разработку 3D игр, являются `Unity Engine` [11] и `Unreal Engine` [2].

2.3.1. Unity Engine

`Unity` — кроссплатформенный игровой движок, который разрабатывается и поддерживается компанией `Unity Technologies`. Этот движок используется для создания различных видеоигр, виртуальной реальности, анимаций, симуляций и другого интерактивного контента. Имеет активное сообщество и обширную библиотеку ассетов, а также учебные материалы и форум. Основным языком для написания скриптов является C#. Концепт движка состоит в том, что каждый объект на сцене является игровым объектом, содержащим различные компоненты, определяющие его поведение и внешний вид. Основным классом является `MonoBehaviour`, предоставляющий основные методы для управления поведением объектов в игре, такие как `Start()`, `Update()`. Метод `Start()` предназначен для инициализации объекта до начала выполне-

ния каких-либо методов, вызывается один раз при активации объекта на сцене. Метод `Update()` вызывается на каждом кадре игры, что позволяет обновлять логику объекта в реальном времени. Еще одним важным классом является `ScriptableObject`, предназначенный для создания и хранения данных вне сцен. Часто используется для хранения настроек, ресурсов и информации, не привязанной к конкретным объектом на сцене.

2.3.2. Unreal Engine

`Unreal Engine` — мощный и широко используемый игровой движок, разработанный компанией Epic Games. Он предоставляет разработчикам среду для создания высококачественных 2D и 3D видеоигр, виртуальной реальности, архитектурных визуализаций, тренировочных симуляторов и многого другого. Поддерживает программирование на языке C++ и систему программирования под названием `Blueprints`, предоставляющую визуальный интерфейс для создания логики игры без необходимости программирования на языке кода.

2.4. Использованные технологии

Для разработки данной игры был сделан выбор в пользу `Unity Engine`, так как имеется опыт разработки на `.NET`. Также большое количество обучающего материала облегчает процесс изучения возможностей данного движка.

Для создания 3D модели персонажа использовалось приложение `Fuse` [3] от компании `Adobe`, предоставляющее широкий набор гуманоидных моделей. Анимирование модели происходило с помощью технологии `Mixamo`, позволяющей для созданной модели подобрать необходимые костевые анимации (`rig animation`). Аналоги данной технологии не рассматривались, так как `Mixamo` идет в связке с приложением `Fuse` и предоставляет разнообразную базу высококачественных анимаций. Для обрезания рук модели игрока была использована программа `Blender` [1].

Для считывания ввода игрока задействован пакет `Input System` [6], предоставляющий единый интерфейс для работы с различными устройствами ввода.

Для построения окружения был задействован пакет `ProBuilder` [4], позволяющий без глубоких знаний в моделировании объемных объектов создавать и гибко настраивать 3D модели.

Для реализации визуальных эффектов были использованы пакеты `Post Processing` [7] и `Animation Rigging` [5]. `Post Processing` позволяет с помощью компонент на основной камере накладывать необходимые эффекты и фильтры на изображение. `Animation Rigging` предоставляет интерфейс для создания сложных и интересных анимаций персонажей, в том числе с использованием инверсной кинематики.

3. Детали реализации

В данном разделе предлагается рассмотреть основные детали реализации игровых моделей и механик. Разработка осуществлена на языке C#.

3.1. Концепт игры

Действие игры происходит в недалёком прошлом, главный герой это детектив, получающий письмо с просьбой о помощи в расследовании странного дела. Игра представляет из себя последовательность локаций с возможностью их альтернативного прохождения. Игрок может пройти локацию, уничтожив всех противников в открытую, используя при этом определённый набор способностей, однако такой стиль повлияет на психическое состояние главного героя, что отразится на концовке и некоторых выборах в диалогах. С другой стороны, локацию возможно пройти скрытным путем, сохранив психическое состояние главного героя. Помимо общего психического состояния у главного героя существует динамическое состояние, на которое могут повлиять внешние факторы, например, какое-либо страшное событие, попавшее в его область видимости.

3.2. Модель главного героя

Модель главного героя создана с помощью программы `Fuse` от компании `Adobe`, оснащена скелетом для реализации анимирования с помощью `Mixamo` и обрезана до состояния рук с помощью программы `Blender`. Такая реализация упрощает взаимодействие с камерой, так как нет необходимости задумываться о том, что в камеру попадет внутренность головы главного героя или же предмет из рук. Для того, чтобы персонаж мог держать в руке оружие, был подключен пакет `Animation Rigging`, позволяющий на основе существующего скелета рук накладывать анимацию поверх существующих. С помощью компоненты `Two Bone IK` рука персонажа привязывается к необходимой точке, в данном

случае к оружию, соблюдая при этом все костные соединения. Такая реализация позволяет в будущем иметь возможность персонажу держать не только оружие, но и любой предмет, поменяв лишь расположение костей для кисти, что упрощает взаимодействие с объектами. Из готовой модели был создан префаб для упрощения использования в различных сценах.

Модель оснащена составным коллайдером, что позволит в будущем релизовать механику ранений для разных частей тела. На данный момент имеются коллайдеры для ног, туловища, двух рук и головы.

3.3. Передвижение игрока

Передвижение игрока реализовано с использованием пакета `Input System`. Для считывания поступаемых данных от игрока был написан класс `InputManager`, наследуемый от главного класса `MonoBehavior`, что позволяет навесить его как компоненту на игровой объект. Далее, считываемая информация, хранящаяся в свойствах и полях данного класса, передается реализованному классу `PlayerController`, хранящему в себе реализации функций для передвижения игрока (рисунок 1).

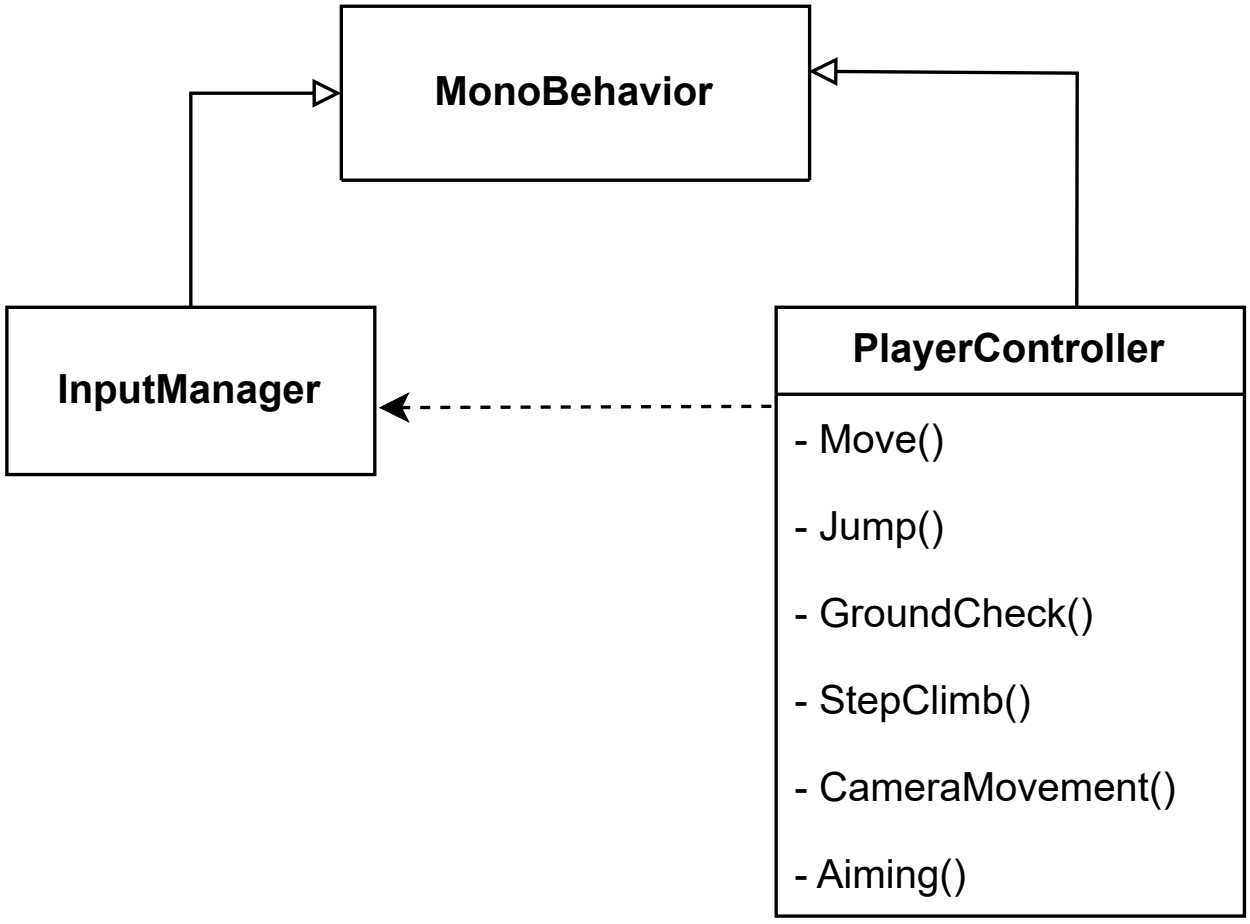

Рис. 1: Диаграмма классов для управления персонажем

Перемещение игрока реализовано с помощью применения к нему физической силы, а не изменением координат. Это позволяет иметь более плавные движения и избегать проблем с прохождением физических тел сквозь игрока, например пуль врагов.

Для прыжка необходимо находиться на поверхности, для проверки данного состояния написана функция `GroundCheck`, которая пускает луч из центра массы тела вниз и проверяет столкновение с коллайдером поверхности.

Персонаж имеет возможность переступать через невысокие препятствия на пути, для этого написана функция `StepClimb`. Из ног и коленей персонажа пускается два луча в направлении взгляда, в случае столкновения только нижнего луча с коллайдером поверхности, персонаж перемещается прямо перед собой на высоту препятствия. Для обзора реализована функция `CameraMovement`, принимающая значения

от `InputManager` и поворачивающая камеру и физическое тело игрока на нужный угол с помощью метода `MoveRotation`.

3.4. Оружие

В качестве оружия реализован пистолет, на данный момент возможность стрельбы отсутствует. Модель оружия была взята из публичной библиотеки 3D моделей с соблюдением лицензии. Для управления оружием был реализован класс `WeaponController`, наследуемый от `MonoBehavior` (рисунок 2). В данном классе реализованы функции для покачивания оружия в режиме «простоя» главного героя, для следования оружия за взглядом игрока и для тряски во время бега или ходьбы. Покачивание оружия осуществляется по траектории Лиссажу [9].

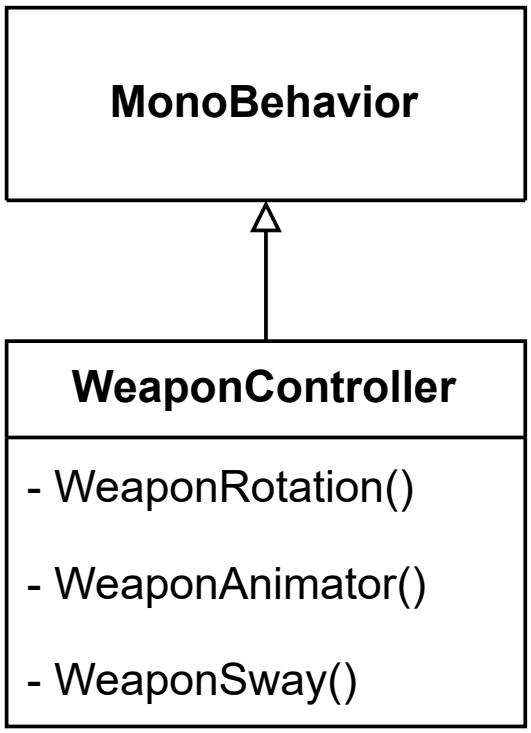

Рис. 2: Диаграмма классов для оружия

3.5. Система психического состояния

Для реализации динамического психического состояния главного героя был подключен пакет `PostProcessing` и реализован специальный

класс `TriggerDeadBody`, наследуемый от `MonoBehavior`. Данный скрипт предназначен для ухудшения психического состояния героя при попадании в область зрения мертвого тела. При столкновении коллайдера игрока со специальным коллайдером на мертвом теле начинает осуществляться проверка нахождения мертвого тела в области видимости камеры игрока. При выполнении всех условий включается постобработка изображения игрока с постепенным усилением эффекта с течением времени.

3.6. Система способностей

Для реализации системы способностей создан класс `AbilityManager`, наследуемый от `MonoBehavior`. Данный класс получает информацию от класса `InputManager` и обрабатывает состояния способностей игрока. Реализован класс `Ability`, наследуемый от класса `ScriptableObject`, являющийся абстракцией над всеми способностями игрока (рисунок 3). Каждая способность обладает названием, активным временем и временем перезарядки, что определяет три состояния для каждой способности — готовое, активное и перезарядка. На данный момент реализованы две способности:

1. ярость, изменяющая скорость игрока и накладывающая эффекты постобработки на изображение;

2. притягивание врагов. При активации способности пускается рейкаст из центра камеры игрока, в случае столкновения с коллайдером врага, враг будет притянут к игроку путем приложения к нему силы.

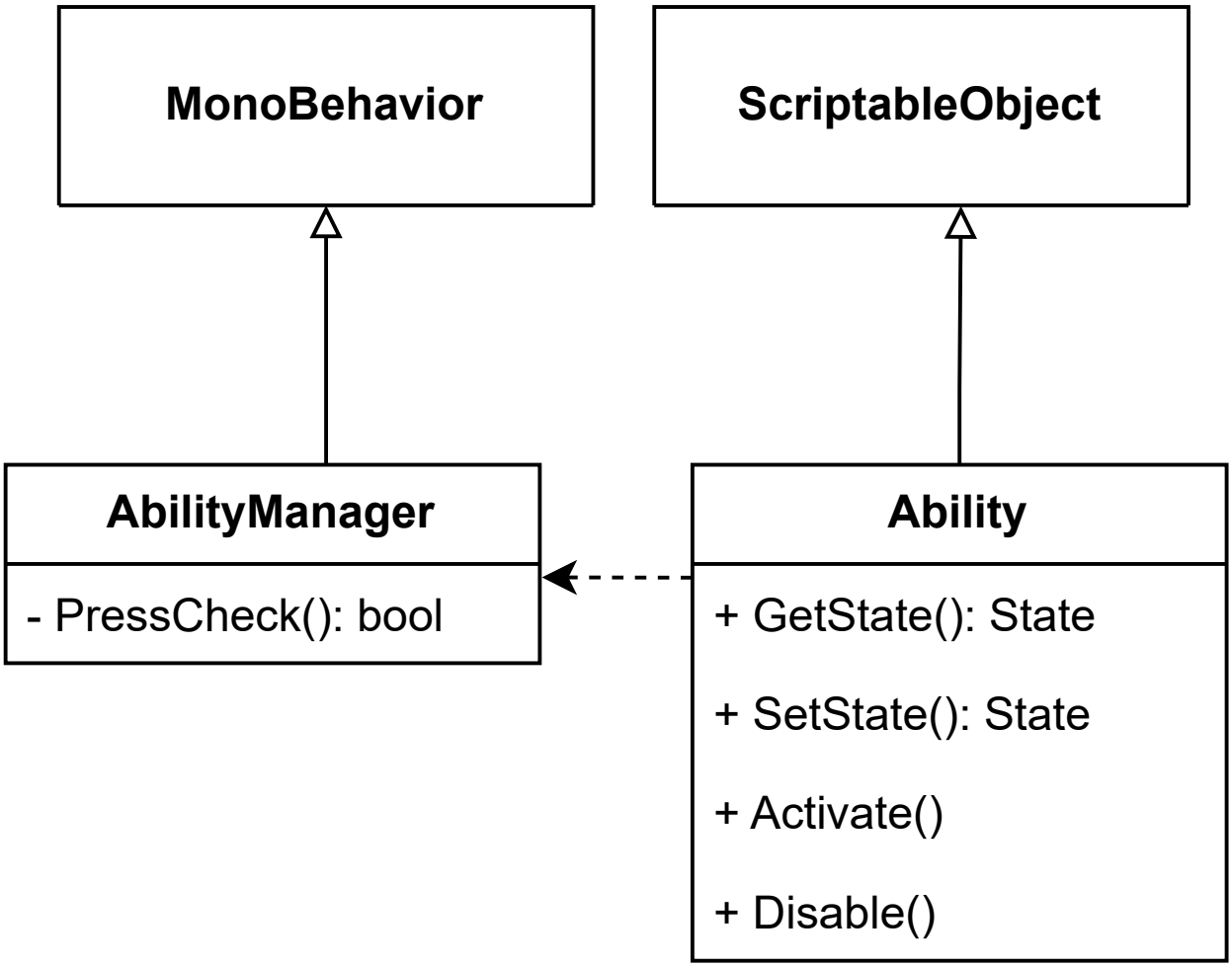

Рис. 3: Диаграмма классов для способностей

Заключение

В рамках проведения работы были получены следующие результаты.

- Разработан концепт игры.
- Проведен обзор существующих аналогов.
- Изучена документация используемого движка.
- Реализована модель главного героя и некоторые игровые механики, такие как:
    1. передвижение игрока;
    2. оружие и его анимации;
    3. система психического состояния игрока;
    4. система способностей игрока.

В качестве будущих задач можно выделить следующие пункты.

- Расширение системы способностей игрока.
- Реализация стрельбы и системы здоровья персонажа.
- Система инвентаря.
- Интерактивное окружение.
- Расширение системы психического состояния персонажа.

Полный код реализации доступен в публичном репозитории[1].

[1]Репозиторий с реализацией алгоритма: https://github.com/LeonidLodygin/EdgeOfMadnesss (дата доступа: 7 декабря 2023 г.).

Список литературы

[1] Community Blender Online. — Blender - a 3D modelling and rendering package. — Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. — Дата обращения: 07.12.2023. URL: http://www.blender.org.

[2] Epic Games. Unreal Engine. — Дата обращения: 07.12.2023. URL: https://www.unrealengine.com.

[3] Fuse. — Дата обращения: 07.12.2023. URL: https://www.adobe.com/ie/products/fuse.htmlMarmoset.

[4] ProBuilder. — Дата обращения: 07.12.2023. URL: https://unity.com/features/probuilder.

[5] Unity Documentation.AnimationRigging. — Дата обращения: 07.12.2023. URL: https://docs.unity3d.com/Packages/com.unity.animation.rigging@1.0/manual/index.html.

[6] Unity Documentation.Input System. — Дата обращения: 07.12.2023. URL: https://docs.unity3d.com/Packages/com.unity.inputsystem@1.7/manual/index.html.

[7] Unity Documentation.PostProcessing. — Дата обращения: 07.12.2023. URL: https://docs.unity3d.com/Packages/com.unity.postprocessing@3.0/manual/index.html.

[8] Wikipedia contributors. Call of Cthulhu: Dark Corners of the Earth — Wikipedia, The Free Encyclopedia. — 2023. — Дата обращения: 07.12.2023. URL: https://en.wikipedia.org/w/index.php?title=Call_of_Cthulhu:_Dark_Corners_of_the_Earth&oldid=1182890909.

[9] Wikipedia contributors. Lissajous curve — Wikipedia, The Free Encyclopedia. — 2023. — Дата обращения: 07.12.2023.

URL: https://en.wikipedia.org/w/index.php?title=Lissajous_curve&oldid=1187167880.

[10] Wikipedia contributors. Resident Evil 7: Biohazard — Wikipedia, The Free Encyclopedia. — 2023. — Дата обращения: 07.12.2023. URL: https://en.wikipedia.org/w/index.php?title=Resident_Evil_7:_Biohazard&oldid=1188461634.

[11] Wikipedia contributors. Unity (game engine) — Wikipedia, The Free Encyclopedia. — 2023. — Дата обращения: 07.12.2023. URL: https://en.wikipedia.org/w/index.php?title=Unity_(game_engine)&oldid=1186824491.