Минобрнауки России

Юго-Западный государственный университет

Кафедра программной инженерии

ОТЧЕТ

о преддипломной (производственной) практике

наименование вида и типа практики

на (в) ООО «МЦОБ. Онлайн-сервисы»

наименование предприятия, организации, учреждения

Студента 4курса, группы ПО-02б

курса, группы

Снатенкова Николая Ивановича

фамилия, имя, отчество

Руководитель практики от предприятия, организации, учреждения

директор

должность, звание, степень

Оценка _____

_____ _____

фамилия и. о. подпись, дата

Руководитель практики от университета

к.т.н. доцент

должность, звание, степень

Оценка _____

Чаплыгин А. А. _____

фамилия и. о. подпись, дата

Члены комиссии

_____ _____

подпись, дата фамилия и. о.

_____ _____

подпись, дата фамилия и. о.

_____ _____

подпись, дата фамилия и. о.

Курск 2024 г.

СОДЕРЖАНИЕ

1 Анализ предметной области 5
1.1 История создания трёхмерной компьютерной графики 5
1.2 Основные принципы работы трёхмерной графики 5
1.3 Работа с OpenGL 6
2 Техническое задание 8
2.1 Основание для разработки 8
2.2 Цель и назначение разработки 8
2.3 Требования к программной системе 8
2.3.1 Требования к данным программно-информационной системы 8
2.3.2 Функциональные требования к программной системе 9
2.3.3 Требования к графическому интерфейсу программы 10
2.4 Моделирование вариантов использования 11
2.5 Нефункциональные требования к программной системе 12
2.6 Требования к оформлению документации 12
3 Технический проект 13
3.1 Общая характеристика организации решения задачи 13
3.2 Обоснование выбора технологии проектирования 13
3.3 Описание используемых технологий и языков программирования 14
3.3.1 Язык программирования C# 14
3.3.1.1 Достоинства языка C# 14
3.3.1.2 Недостатки языка C# 14
3.3.2 Спецификация OpenGL 4.6 14
3.3.2.1 Достоинства спецификации OpenGL 15
3.3.2.2 Недостатки спецификации OpenGL 15
3.4 Диаграмма классов и компоненты программы 16
3.5 Описание работы парсера 18
3.6 Проектирование пользовательского интерфейса 20
3.6.1 Главное окно программы 20
3.6.2 Окно инспектора 21

3.6.3 Вкладки окон инспектора 21
3.6.4 Активное меню вкладки инспектора 22
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 22

ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

ИС – информационная система.

ИТ – информационные технологии.

КТС – комплекс технических средств.

ОМТС – отдел материально-технического снабжения.

ПО – программное обеспечение.

РП – рабочий проект.

ТЗ – техническое задание.

ТП – технический проект. ё UML (Unified Modelling Language) – язык графического описания для объектного моделирования в области разработки программного обеспечения.

OBJ – формат файлов описания геометрии, разработанный в Wavefront Technologies для их анимационного пакета Advanced Visualizer. Формат файла является открытым и был принят другими разработчиками приложений 3D-графики.

3D (от англ. 3-dimensional) графика – раздел компьютерной графики, посвящённый методам создания изображений или видео путём моделирования объектов в трёх измерениях.

GL (Graphic Library) – это программная библиотека, предназначенная для оказания помощи в рендеринге компьютерной графики на мониторе.

1 Анализ предметной области

1.1 История создания трёхмерной компьютерной графики

Впервые трёхмерная компьютерная графика была реализована в 1960-х годах, когда Айван Сазерленд и Дэвид Эванс основали первую в мире кафедру компьютерной графики в университете Юты, США. Основой трёхмерной компьютерной графики стала Евклидова геометрия, а также все математические открытия, которые были сделаны до XX века. Первые трёхмерные изображения состояли из множества точек и кривых, определяемых математическим уравнением. Именно в то время появился прародитель для всех современных 3D-редакторов - программа Sketchpad.

Развитие трёхмерной графики шло быстрым ходом, и уже совсем скоро вместо векторной графики появилась возможность создавать и трёхмерные растровые изображения, а также использовать различные текстуры для объектов, обрабатывать их тени, и наконец - проводить компиляцию всего кода в готовое изображение.

1.2 Основные принципы работы трёхмерной графики

Главным основоположником систематизации математических аксиом и теорем в области геометрии был Евклид. Именно его труды способствовали разработке и технологическому прорыву трёхмерной графики в XX веке. Но за её развитием стоит не только сам Евклид, а также и другие великие умы и открытия, как например, формулы Виета для нахождения корней квадратичного уравнения, благодаря чему в символьном анализе алгебры неизвестные переменные обозначаются как x, y и z, а коэффициенты - a, b и c. А основой отсчёта пространства стала система трёхмерных координат Декарта.

Также незаменимый влкад в разработку трёхмерной графики и геометрии внесли Российские учёные XX века - Борис Делоне и Георгий Вороной. Борис Делоне предложил метод «Триангуляции Делоне», которая стала основой формирования граней трёхмерных моделей, а Георгий Вороной создал

«Диаграмму Вороного», которая используется до сих пор в картографических софтах, дизайне и трёхмерной графике.ё

Любая трёхмерная модель объекта математически находится в трёх измерениях. И если, чтобы отрисовать изображение объекта в двух измерениях компьютерной мощи требовалось не так много, то с добавлением третьего измерения - оси Z, требовательность к производительности компьютерного железа резко возрасла.

Трёхмерные объекты математически представляются в виде множества точек - вершин, и в тоже время они, соединенные отрезками, образуют рёбра фигиры, а множество рёбер на одной плоскости образуют поверхности - грани фигуры. Данный массив точек трансформируется с учётом матрицы перспективы камеры, далее - накладываются текстуры на поверхности граней фигуры - изображения также растягивают и трансформируют с учётом перспективы наклона грани, затем - обрабатывают свет, падающий на модель, путём изменения яркости текстуры граней, в зависимости от источников света, и наконец - добавляют тень, которая отбрасывает данная трёхмерная модель.

1.3 Работа с OpenGL

”OpenGL”(Open Graphics Library) - это кроссплатформенная спецификация, независимая от языка программирования, которая определяет программный интерфейс для написания приложений для двумерной и трёхмерной компьютерной графики. По своей сути, OpenGL - это низкоуровневый API, который позволяет напрямую работать с коммандами и буфферами видеокарты, так что для его использования необходимо иметь хотя бы основные понимания работы трёхмерной графики и линейной алгебры. Для простого начала знакомства и использования спецификации OpenGL существует множество официальных готовых эффективных реализаций для Windows, Unix-платформ и MacOS.

Спецификация OpenGL была создана в эпоху активного распространения компьютерных 3D игр и приложений, когда взникла сложность с сов-

местимостью устройств во время реализации компьютерного кода при использовании различных аппаратных устройств - процессоров, видеоадаптеров, устройств хранения информации и материнской платы. Это вызывало сильные затраты, трудности и замедляло разработку программного обеспечения. Поэтому, компания Silicon Graphics - лидирующая в то время в сфере производства оборудования для обработки трёхмерной графики разработала программный интерфейс, целью которого было систематизировать доступ и обработку трёхмерных моделей на аппаратном уровне. Плодами их трудов стала спецификация OpenGL, который стандартизировал обработку различных функций 3D систем, которые выполняли единые команды из списка доступных, согласно установленной программой спецификации, что позволило создавать программное обеспечение, которое одинаково корректно работает на всех видах графического оборудования.

Принцип работы OpenGL заключается в получении геометрических векторных примитивов и построении растровой картинки в памяти видеокарты и выводе её на экран. Из-за своей низкоуровневости, данная спецификация требует диктовать программе точный порядок действий от программиста и использовать императивный подход в разработке приложений. Но несмотря на эти сложности, это даёт большой простор, гибкость и свободу в создании программ.

Последняя вышедшая версия OpenGL 4.6, в данный момент уже считается устаревшей, и на смену ей пришёл современный и оптимизированный для новых видеокарт API Vulcan, который стал прямым преемником OpenGL.

2 Техническое задание

2.1 Основание для разработки

Основанием для разработки является задание на производственную преддипломную практическую работу в компании «ООО МЦОБ. Онлайн Сервисы».

2.2 Цель и назначение разработки

Основной задачей производственной преддипломной практической работы является разработка программной системы для визуализации трёхмерных данных, использующая библиотеку OpenGL.

Используя библиотеку OpenGL последней версии, планируется разработать приложение на языке C#, способное импортировать массив трёхмерных данных и визуализировать их в виде трёхмерных моделей.

Задачами данной разработки являются:

- создание программы, реализующей графику, на основе спецификации OpenGL;
- реализация программы парсера, способного преобразовывать и загружать в программу трёхмерные данные;
- разработка интерфейса для взаимодействия с программой;
- реализация функции хранения и загрузки трёхмерных данных внутри программы.

2.3 Требования к программной системе

2.3.1 Требования к данным программно-информационной системы

Входными данными для программной системы являются файлы с расширением *.obj, внутри которых содержатся информация в виде массивов трёхмерных данных; файлы текстур - изображения с расширением *.png и

*.jpg; данные ввода с клавиатуры и мыши, которые служат для управления перемещением и вращением камеры.

Выходными данными для программной системы является выводимое на экран изображение двумерной проекции трёхмерных объектов на виртуальной сцене.

На рисунке 2.1 представлена диаграмма описания потоков данных в программе.

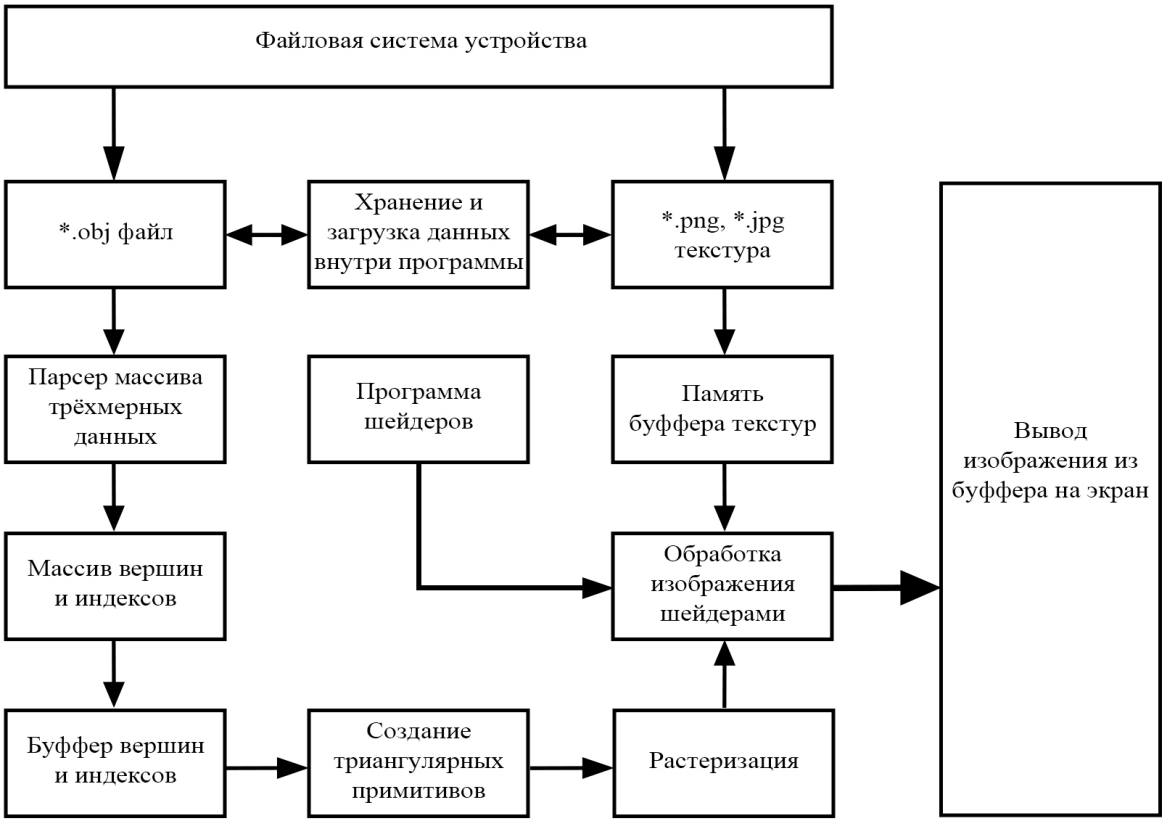

Рисунок 2.1 – Диаграмма потоков данных в программе

2.3.2 Функциональные требования к программной системе

Программа должна реализовывать следующие функции:

- позволять пользователю импортировать любые массивы трёхмерных данных (включая текстуры для трёхмерных моделей);

- выполнять отрисовку трёхмерной сцены в реальном времени (каждый кадр);

- пользволять пользователю свободно управлять перспективой виртуальной камеры;

- предоставлять возможность преобразовывать массив трёхмерных данных (производить аффинные преобразования) в рамках: сдвига, вращения и растяжения (сжатия);

- импортировать, хранить и загружать массив трёхмерных данных внутри самой программы.

Виды преобразований массива трёхмерных данных, предоставляемых программой, представлены на рисунке 2.2.

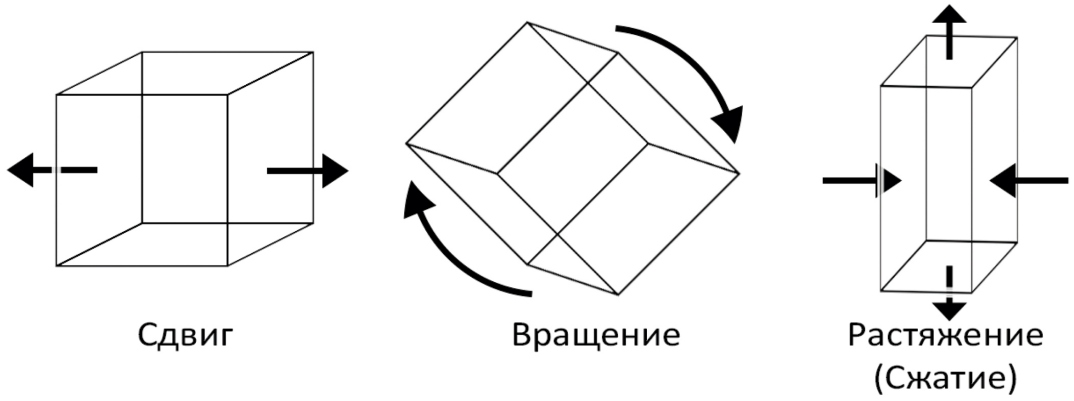

Рисунок 2.2 – Виды преобразований массива трёхмерных данных

2.3.3 Требования к графическому интерфейсу программы

Программное обеспечение должно иметь минимальный, но содержательный и интуитивный интерфейс. Большую часть основного окна приложения должно занимать само окно вывода растрового изображения проекции виртуальной сцены и всех трёхмерных данных на ней. Сбоку на основном окне приложения должно находиться специально выведенное отдельное окно «инспектора», с помощью которого пользователь сможет управлять загрузкой и сохранением массивов трёхмерных данных, а также взаимодействовать с уже существующими данными на виртуальной сцене.

Макет пользовательского интерфейса, составленный по данным требованиям представлен на рисунке 2.3

Рисунок 2.3 – Макет пользовательского интерфейса

2.4 Моделирование вариантов использования

Для разрабатываемого программного обеспечения была реализована модель, которая демонстрирует наглядное представление вариантов использования программы.

На основании анализа предметной области в программе должны быть реализованы следующие прецеденты:

1. Импортирование массивов трёхмерных данных из файловой системы пользователя.
2. Просмотр визуализированного массива трёхмерных данных в виде отрисованных объектов на экране.
3. Осуществление трансформаций над массивом трёхмерных данных.
4. Хранение и загрузка массивов трёхмерных данных.

На рисунке 2.4 представлена диаграмма вариантов использования

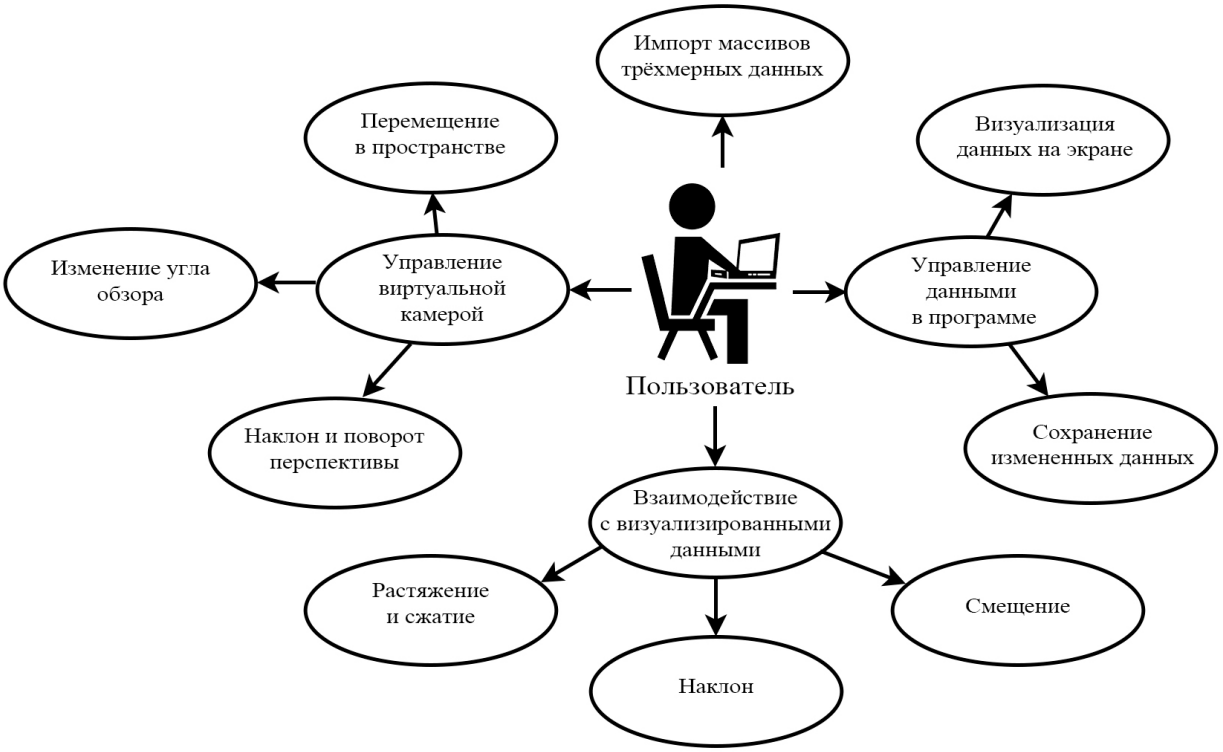

Рисунок 2.4 – Диаграмма вариантов использования

2.5 Нефункциональные требования к программной системе

Требования к аппаратной совместимости:

- видеоадаптер с поддержкой OpenGL версии не ниже 4.6;
- минимальное разрешение экрана - 800x600 пикселей.

Требования к программной совместимости:

- операционная система x86 Windows 7 и выше;
- система, с установленными компонентами Microsoft Visual C++ версии 2015 года и выше.

2.6 Требования к оформлению документации

Разработка программной документации и программного изделия должна производиться согласно ГОСТ 19.102-77 и ГОСТ 34.601-90. Единая система программной документации.

3 Технический проект

3.1 Общая характеристика организации решения задачи

Поставлена цель разработать программу, работающую на спецификации OpenGL, способную загрузить массив трёхмерных данных и графически осуществить их визуализацию на экране для пользователя.

Трёхмерный графический движок на высоком уровне представляет из себя структуру основных классов - примитивов для построения основной минимальной трёхмерной графики: массив координат точек, массив нумерации граней, матрица проекциий, объект виртуальной камеры, набор текстур и программ шейдеров, которые все вместе осуществляют отрисовку объекта на трёхмерной сцене.

На более низком уровне трёхмерный движок представляет из себя набор комманд и программ для работы с видеоадаптерами и буфферами памяти системы. Из данного набора команд и программ была создана основа разрабатываемой программы, работающей под спецификацией OpenGL.

В движке используется разработанная программа парсера. Парсер считывает структуру файлов трёхмерных объектов. Массив трёхмерных данных считывается последовательно и обрабатываются синтаксическим анализатором. На основе данных считанных объектов формируются массивы координат вершин, индексов и координат текстур.

3.2 Обоснование выбора технологии проектирования

Обоснованием выбора технологии проектирования послужило задание на разработку, целью которой являлось создание программы, работающей на спецификации OpenGL. Соответственно, была выбрана последняя выпущенная версия OpenGL 4.6. А выбором среды и языка проектирования интерфейса и создания программы послужило удобство и простота использования языка C# в среде Visual Studio.

3.3 Описание используемых технологий и языков программирования

В процессе разработки программного обеспечения было использовано программное средство IDE Visual Studio, а также использованы языки программирования C# - при создании интерфейса программы и C - при работе со спецификацией OpenGL.

3.3.1 Язык программирования C#

3.3.1.1 Достоинства языка C#

C# - многоцелевой объектно-ориентированный язык программирования. Относится к семье с C-подобным синтаксисом, наиболее идентичен C++ и Java. Имеет более расширенный спект функций, чем у предшественника - C++, чем является намного проще в использовании, но также из-за своих удобств является более высокоуровневым, что делает его ориентированным в основном на разработку десктопных приложений.

3.3.1.2 Недостатки языка C#

Язык C# является немного более высокоуровневым, чем C++, поэтому менее тесно взаимодействует с аппаратной частью вычислительных систем, что сильно ограничивает его функционал в случаях разработки низкоуровневых приложений, где работа с памятью системы и прямое взаимодействие с аппаратной частью необходимо.

3.3.2 Спецификация OpenGL 4.6

OpenGL ориентирован на две задачи:

- Скрыть сложности адаптации различных 3D-ускорителей, предоставляя разработчику единый API;

- Скрыть различия в возможностях аппаратных платформ, требуя реализации недостающей функциональности с помощью программной эмуляции.

3.3.2.1 Достоинства спецификации OpenGL

Главным достоинством OpenGL является его гибкость, открытый код и низкие требования к ресурсам устройства.

- Возможности данной спецификации позволяют разработчику создавать полностью уникальные и подстроенные под особую специфику задач приложения;

- устройство данной графической библиотеки позволяет запускать и поддерживать приложения с максимально возможной производительностью;

- это низкоуровная библиотека, которая позволяет напрямую работать с аппаратным железом - управлять памятью системы и буффером видеоадаптера;

- самодостаточность спецификации даёт возможность не использовать дополнительные плагины и библиотеки в реализации визуализации графики;

- полная мультиплатформенность - OpenGL работает на всех платформах, языках и устройствах.

3.3.2.2 Недостатки спецификации OpenGL

- Низкоуровневость библиотеки - для неопытного, или же начинающего разработчика это может стать главной проблемой в работе с OpenGL, потому как для создания программного обеспечения, использующего спецификацию OpenGL, необходимы глубокие знания об основах трёхмерной графики и линейной алгебры, а также иметь минимальное представление об обмене данными в видеоадаптерах на аппаратном уровне;

- данная технология в настоящее время считается уже устаревшей, и была заменена более современным API Vulcan;

- с выходом новых видеокарт, их спектр возможностей, как и реализуемых функций расширился, так что новые функции, такие как DLSS и RTX не вошли в стандарт спецификации OpenGL.

3.4 Диаграмма классов и компоненты программы

Диаграмма классов описывает виды классов программы и различного рода связи, которые существуют между элементами. На диаграммах изображаются также атрибуты классов, их функции, принимаемые значения, а также ограничения описывающие взаимодействие между классами. Вид и представление диаграммы классов напрямую зависит от уровня абстракции: классы могут быть показаны в качестве сущностей предметной области или же как элементы частей программного обеспечения. В нашем случае, на рисунке 3.1 показана диаграмма классов, которые являются элементами программной среды.

Атрибуты в диаграмме описывают свойства объектов класса. Элементы класса обладают своей индивидуальностью из-за различий в их параметрах, а также связях с другими объектами.

В большинстве случаев, в данном программном обеспечении большинство классов существуют в единственном экземпляре, из-за их уникальной сущности. Как пример, можно привести класс, определяющий виртуальную камеру, которая должна существовать в единственном экземпляре, ведь на один экран должно выводиться одновременно лишь одно изображение.

Но несмотря на это, структура классов в данном программном обеспечении всё равно является объектно ориентированной, благодаря чему внутри неё был реализован механизм управления объектными моделями, текстурами и другими элементами, которые все связаны между собой наследуемостью классов, а также существует возможность дальнейшего развития и расширения общей системы классов, путём добавлением в неё новых элементов.

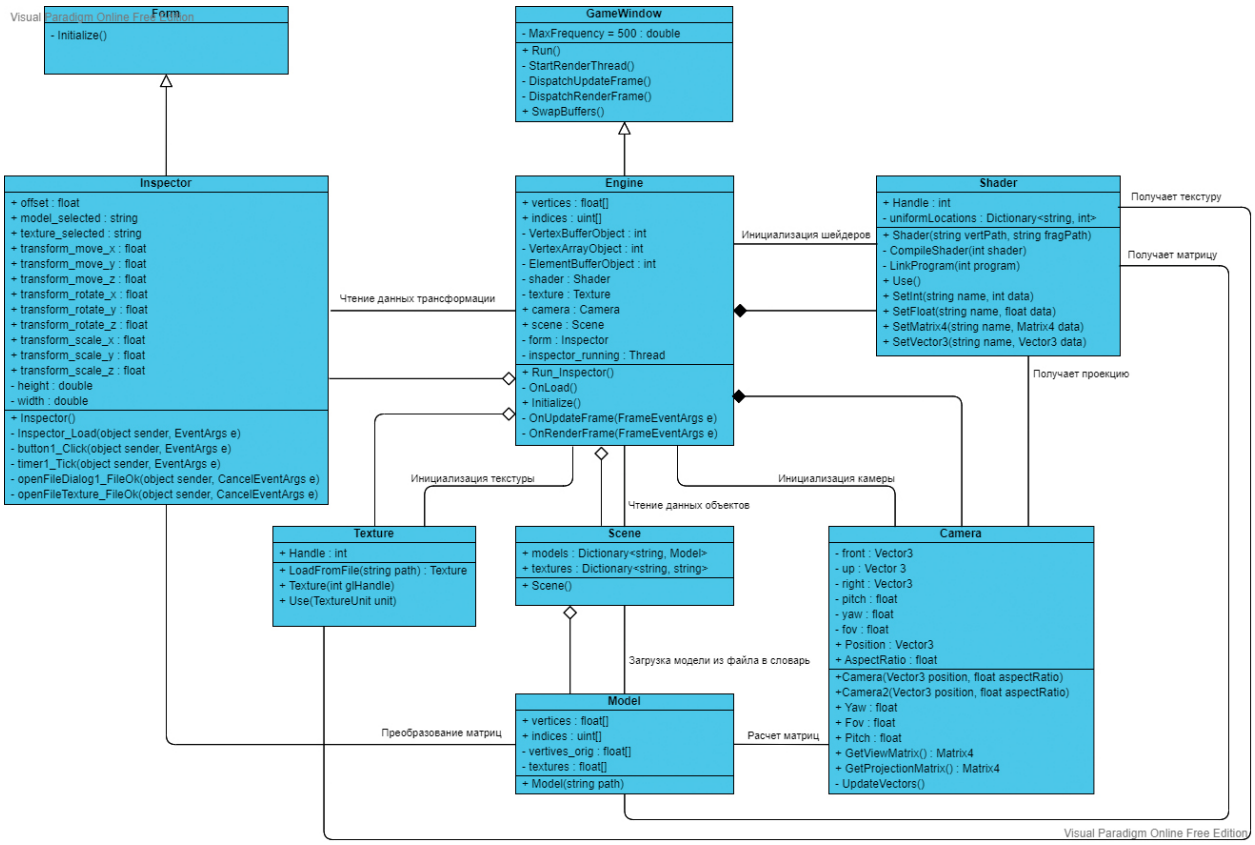

Рисунок 3.1 – Диаграмма классов программы

Все элементы программы имеют общий родительский элемент в виде главного класса - обработчика событий, который является главным элементом и центром всей программы.

- Класс Model является каркасом трёхмерной модели. Выполняет роль приёма, хранения и передачи массива трёхмерных данных для обработчика триангулярных примитивов. При создании выполняет чтение и преобразование в массивы вершин и индексов, загруженного файла с данными трёхмерного объекта;

- класс Texture при инициализации выделяет место в буффере памяти видеоадаптера для загруженной в программу текстуры, а также обрабатывает настройки параметров её отображения;

- класс Scene выполняет роль хранилища моделей и изображений, а также выступает виртуальной сценой, на которой находятся все трёхмерные объекты и остальные элементы, учавствующие в визуализации. При инициализации выполняет загрузку данных массивов моделей и привязанных к ним файлов текстур, которые уже были раннее загружены в проект;

- класс Shader является программой настройки конечной визуализации, и при инициализации создаёт подпрограмму в общей графике, для преобразования растеризированного изображения внутри буффера памяти видеоадаптера, для отображения текстур и освещения;

- класс Camera является матрицей преобразования, для корректного отображения перспективы виртуального трёхмерного вида сцены, и для того, чтобы конечный вид проекции сцены был перспективным, а не ортографическим;

- класс Engine является основой проекта, графическим движком, который инициализирует всю графику, принимает события ввода пользователя, отвечает за обновление кадров и управление всеми настройками, а также связывает все классы между собой;

- класс Inspector является частью пользовательского интерфейса, и представляет из себя окно взаимодействия между пользователем и программой. Служит для того, чтобы пользователь осуществлял загрузку собственных массивов трёхмерных данных моделей и текстур в проект, а также осуществлял указанные аффинные преобразования над загруженными моделями.

Также отдельно стоит указать класс GameWindow - это родительский класс для класса Engine. По своей сути он является измененным элементом класса Form от WindowsForms и не принимает прямого участия в работе проекта, но содержит формальные данные, необходимые для корректной инициализации главного окна проекта.

3.5 Описание работы парсера

В результате тесной работы со множеством различных файлов трёхмерных данных с расширением *.obj, возникла необходимость автоматизировать загрузку и интерпретацию этих данных для программы. Для решения данной задачи был разработан специальный алгоритм программы парсера.

Парсер, или же синтаксический анализатор - часть программы, которая преобразовывает текстовые входные данные в структурированный формат.

Главный принцип работы разрабатываемого приложения завязан на использовании файлов трёхмерных данных формата *.obj. Это формат файлов описания геометрии, разработанный в Wavefront Technologies. Он содержит только 3D геометрию, а именно: позицию каждой вершины, связь координат текстуры с вершиной, направления нормалей, а также параметры, которые создают триангулярные примитивы. Исходя из этого, необходимо, чтобы парсер мог разделить все эти данные на отдельные массивы, которые будут использовать программа. В заключение вышесказанного, можно построить схему, по которой должен будет работать парсер, как показано на рис. 3.2, где изображён общий принцип работы подпрограммы парсера.

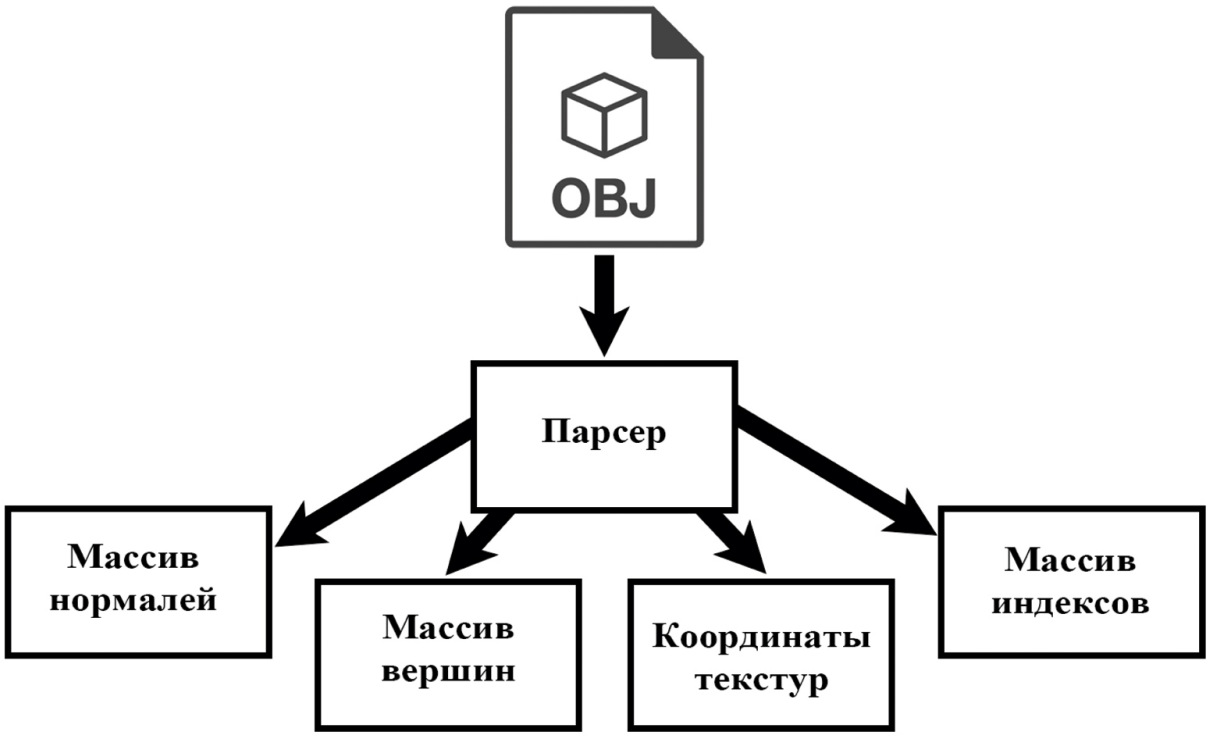

Рисунок 3.2 – Принцип работы программы парсера

Так как парсер - это не отдельный класс, то его подпрограмма будет находиться в классе Model, чтобы в будущем каждый элемент этого класса мог сразу автоматически интерпретировать и структурировать данные своей модели в массивы трёхмерных данных. Таким образом, несмотря на хранение в программе трёхмерных данных в файлах формата OBJ, они сразу бы могли

структурироваться под стандарты программы, при инициализации элементов класса Model с указанием нужного файла.

3.6 Проектирование пользовательского интерфейса

Графический прототип интерфейса, показанный на рис. 3.3 демонстрирует, какие элементы пользовательского интерфейса будут включены в конечную реализацию программы.

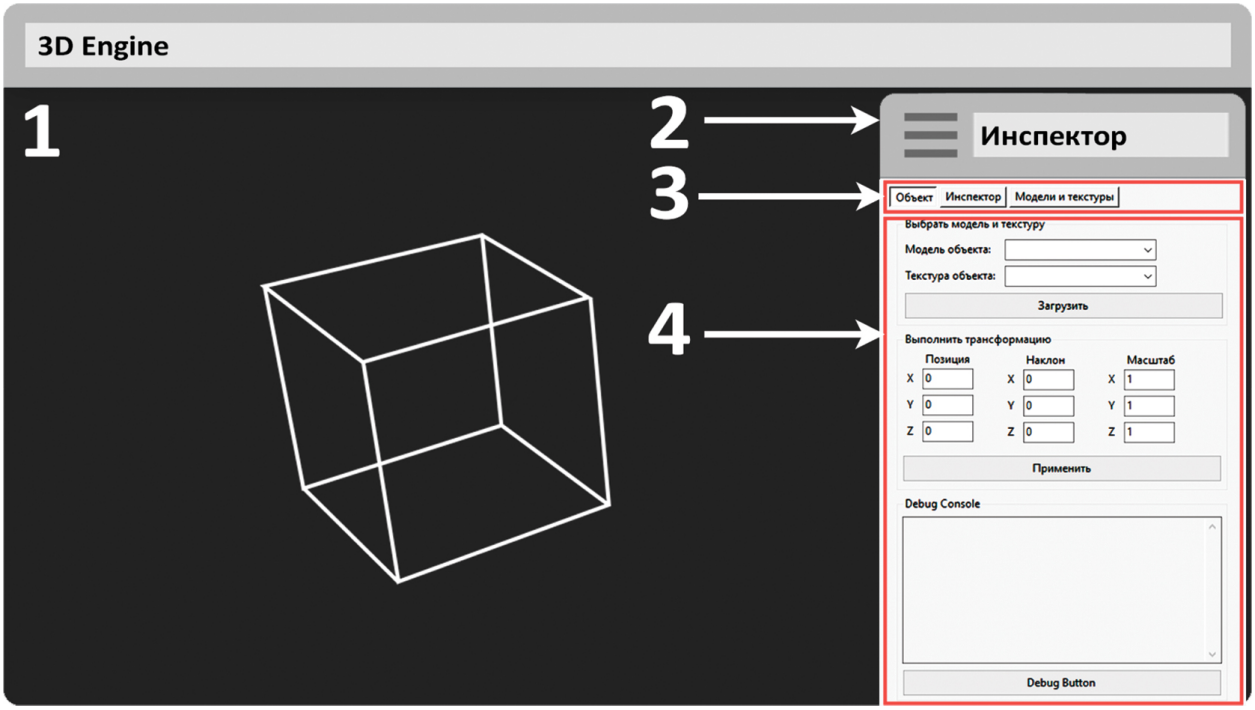

Рисунок 3.3 – Прототип пользовательского интерфейса

Описание элементов интерфейса, показанного на рис. 3.3:

1. Главное окно программы. Выводит визуализированные данные на экран.
2. Окно инспектора.
3. Вкладки окон инспектора.
4. Активное меню вкладки инспектора.

3.6.1 Главное окно программы

Главное окно программы представляет из себя виртуальную трёхмерную среду с одноцветным фоном. Оно служит для отображения визуализации

трёхмерных моделей, которые пользователь загрузит в программу. Также данное окно является частью пользовательского интерфейса с интуитивной функцией ввода - пользователь может управлять направлением перспективы взгляда камеры, зажав правую кнопку мыши и направляя курсор в нужную сторону, чтобы управлять камерой и осматриваться в виртуальном пространстве. А используя колесо прокрутки, пользователь может отдалять и приближать угол обзора.

Управление перемещением камеры в пространстве также осуществляется в главном окне программы, с помощью клавиатуры:

- клавиша «W» – переместить камеру вперёд по направлению взгляда;
- клавиша «A» – переместить камеру вправо относительно направления взгляда;
- клавиша «S» – переместить камеру назад по направлению взгляда;
- клавиша «D» – переместить камеру влево относительно направления взгляда;
- клавиша «левый Shift» – поднять камеру вверх, по координате +Y;
- клавиша «левый Ctrl» – опустить камеру вниз, по координате -Y;

3.6.2 Окно инспектора

Инспектор - это дополнительное окно, расположенное по краю справа, поверх главного окна. Оно служит вспомогательным элементом интерфейса, чтобы пользователь мог взаимодействовать и управлять элементами внутри виртуальной сцены. Например, выполнять преобразование массива данных трёхмерной модели - перемещать её в пространстве, наклонять под определённым углом и прозводить растяжение или сжатие, и всё это по всем трём координатам.

3.6.3 Вкладки окон инспектора

Меню со вкладками окон инспектора, где пользователь может выбрать одну трёх вкладок, чтобы переключать и менять содержание активного меню инспектора.

3.6.4 Активное меню вкладки инспектора

Показывает основной интерфейс меню инспектора. Инспектор имеет три разных активных меню, и содержание окна инспектора будет меняться, в зависимости от выбранной вкладки:

- Вкладка «Объект» - в данной вкладке пользователь может выбрать из выпадающего списка модели и текстуры, которые он хочет загрузить и визуализировать на виртуальной сцене. Также данная вкладка имеет ещё два раздела: раздел трансформации, в котором пользователь может задать значения для преобразования данных на сцене и раздел консоли, в которую будут выводиться данные о выполненных операциях, и информация об ошибках, если в программе произойдут сбои.

- Вкладка «Инспектор» - это информационная вкладка, в которой показаны точные координаты и наклон камеры, а также данные о произведенных преобразованиях над моделью: смещении, наклона и масштабировании.

- В последней вкладке «Модели и текстуры» содержится весь список доступных моделей и текстур в программе, загруженных пользователем. Также в ней пользователь может осуществлять непосредственно саму загрузку массивов трёхмерных данных с расширением *.obj в программу и изображений текстур с расширением *.png и *.jpg.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Дэвид Вольф: OpenGL 4. Язык шейдеров. Книга рецептов / Д. Вольф – Москва : ДМК-Пресс, 2015. – 368 с. – ISBN 978-5-97060-255-3. – Текст : непосредственный.

2. OpenGL. Официальный справочник / Д. Шрайнер – Москва : Диа-Софт, 2002. – 512 с. – ISBN 978-5-93772-048-1. – Текст : непосредственный.

3. Программирование на C# для начинающих. Особенности языка / А.Н. Васильев – Москва : Бомбора, 2022. – 528 с. – ISBN 978-5-04-092520-9. – Текст : непосредственный.

4. C#. Программирование 2D и 3D векторной графики /Н. А. Тюкачев, В. Г. Хлебостроев – Санкт-Петербург : Лань, 2022. – 320 с. – ISBN 978-5-8114-8988-6. – Текст : непосредственный.

5. Объектно-ориентированное программирование с примерами на C#. Учебное пособие. Студентам ВУЗов. / П.Б. Хорев – Москва : Форум, 2023. – 200 с. – ISBN 978-5-00091-680-3. – Текст : непосредственный.

6. М. Ву, Т. Девис, Дж. Нейдер, Д. Шрайнер ”OpenGL. Руководство по программированию”/ Т. Девис, Дж. Нейдер, Д. Шрайнер – Санкт-Петербург : Питер, 2006. – 624 с. – ISBN 978-5-94723-827-6. – Текст : непосредственный.

7. Макфарланд, Д. Большая книга CSS / Д. Макфарланд. – Санкт-Петербург : Питер, 2012. – 560 с. – ISBN 978-5-496-02080-0. – Текст : не посредственный.

8. Лоусон, Б. Изучаем HTML5. Библиотека специалиста / Б. Лоусон, Р. Шарп. – Санкт-Петербург : Питер, 2013 – 286 с. – ISBN 978-5-459-01156-2. – Текст : непосредственный.

9. Титтел, Э. HTML5 и CSS3 для чайников / Э. Титтел, К. Минник. – Москва : Вильямс, 2016 – 400 с. – ISBN 978-1-118-65720-1. – Текст : непосредственный.

10. Титтел, Э. HTML5 и CSS3 для чайников / Э. Титтел, К. Минник. – Москва : Вильямс, 2016 – 400 с. – ISBN 978-1-118-65720-1. – Текст : непосредственный.

11. Титтел, Э. HTML5 и CSS3 для чайников / Э. Титтел, К. Минник. – Москва : Вильямс, 2016 – 400 с. – ISBN 978-1-118-65720-1. – Текст : непосредственный.

12. Титтел, Э. HTML5 и CSS3 для чайников / Э. Титтел, К. Минник. – Москва : Вильямс, 2016 – 400 с. – ISBN 978-1-118-65720-1. – Текст : непосредственный.

13. Титтел, Э. HTML5 и CSS3 для чайников / Э. Титтел, К. Минник. – Москва : Вильямс, 2016 – 400 с. – ISBN 978-1-118-65720-1. – Текст : непосредственный.