**«Санкт-Петербургский государственный электротехнический университет**

**«ЛЭТИ» им. В.И.Ульянова (Ленина)»**

**(СПбГЭТУ «ЛЭТИ»)**

|  |  |  |
| --- | --- | --- |
| **Направление** | 09.03.04 - Программная Инженерия | |
| **Профиль** | Разработка программно-информационных систем | |
| **Факультет** | КТИ | |
| **Кафедра** | МО ЭВМ | |
| *К защите допустить* |  | |
| Зав. кафедрой |  | А.А. Лисс |

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

БАКАЛАВРА

Тема: Разработка программы восстановления трёхмерной поверхности в плоской области с использованием триангуляционных сетей

|  |  |  |  |
| --- | --- | --- | --- |
| Студент |  |  | Ш.-К. Амежее |
|  |  | *подпись* |  |
| Руководитель | д.т.н., профессор |  | А.-В.И. Середа |
|  |  | *подпись* |  |
| Консультанты | к.э.н. |  | О.С. Артамонова |
|  |  | *подпись* |  |
|  | к.т.н. |  | М.М. Заславский |
|  |  | *подпись* |  |

Санкт-Петербург

2024

**ЗАДАНИЕ**

**на выпускную квалификационную работу**

|  |  |
| --- | --- |
|  | Утверждаю |
|  | Зав. кафедрой МО ЭВМ |
|  | \_\_\_\_\_\_\_\_\_\_\_\_ А.А. Лисс |
|  | «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2024 г. |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Студент | Амежее Ш.-К. | | | |  | Группа | 0303 |
| Тема работы: Разработка программы восстановления трёхмерной поверхности в плоской области с использованием триангуляционных сетей | | | | | | | |
| Место выполнения ВКР: СПбГЭТУ «ЛЭТИ» кафедра МО ЭВМ | | | | | | | |
| Исходные данные (технические требования): Необходимо разработать программное обеспечения для восстановления трёхмерной поверхности в плоской области из нерегулярно расположенных точек с использованием триангуляционных сетей. | | | | | | | |
| Содержание ВКР:  Введение, Обзор предметной области, Анализ предметной области, Реализация программы, Исследование разработанной программы, Заключение. | | | | | | | |
| Перечень отчетных материалов: пояснительная записка, иллюстративный материал | | | | | | | |
| Дополнительные разделы: Обеспечение качества разработки программного обеспечения | | | | | | | |
| Дата выдачи задания | | | Дата представления ВКР к защите | | | | |
| «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_ г. | | | «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_ г. | | | | |
| Студент | |  | | Ш.-К. Амежее | | | |
| Руководитель к.т.н., профессор | |  | | А.-В.И. Середа | | | |
| Консультант к.э.н. | |  | | О.С. Артамонова | | | |

**календарный план выполнения**

**выпускной квалификационной работы**

|  |  |
| --- | --- |
|  | Утверждаю |
|  | Зав. кафедрой МО ЭВМ |
|  | \_\_\_\_\_\_\_\_\_\_\_\_ А.А. Лисс |
|  | «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2024 г. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | Амежее Ш.-К. |  | Группа | 0303 |
| Тема работы: Разработка программы восстановления трёхмерной поверхности в плоской области с использованием триангуляционных сетей | | | | |

|  |  |  |
| --- | --- | --- |
| № п/п | Наименование работ | Срок выполнения |
| 1 | Обзор литературы по теме работы | 02.04 – 04.04 |
| 2 | Обзор предметной области | 04.04 – 10.04 |
| 3 | Требования к решению и постановка задачи | 10.04 – 20.04 |
| 4 | Реализация программы | 20.04 – 25.04 |
| 5 | Исследование разработанного инструмента | 25.04 – 30.04 |
| 6 | Обеспечение качества разработки | 30.04 – 05.05 |
| 7 | Оформление пояснительной записки | 05.05 – 07.05 |
| 8 | Оформление иллюстративного материала | 07.05 – 20.05 |
| 9 | Предзащита | 01.06.2024 |

|  |  |  |
| --- | --- | --- |
| Студент |  | Ш.-К. Амежее |
| Руководитель к.т.н., профессор |  | А.-В.И. Середа |
| Консультант к.э.н. |  | О.С. Артамонова |

**РЕФЕРАТ**

Пояснительная записка 57 стр., 18 рис., 2 табл., 20 ист.

ДЕЛОНЕ, ТРИАНГУЛЯЦИЯ, 3D, РЕКОНСТРУКЦИЯ.

**Объектом исследования** является трёхмерная реконструкция поверхности.

**Предметом исследования** является программное обеспечение для восстановления поверхности в плоской области с использованием триангуляционных сетей.

**Цель работы:** Разработка программного обеспечения для восстановления трёхмерной поверхности в плоской области для разных измеренных признаков.

В ходе исследования проведён сравнительный анализ программных средств для реконструкции трёхмерных поверхностей из набора нерегулярно распределенных точек с использованием триангуляционных сетей. На основании выявленных недостатков аналогов был разработан программный инструмент, использующий триангуляцию Делоне для создания сетки из треугольников и визуализации 3D-поверхностей. В проекте используются библиотеки Delaunator и Three.js, а также фреймворк Electron для создания кроссплатформенного пользовательского интерфейса.

**ABSTRACT**

In this paper, a comparative analysis of software tools for reconstructing three-dimensional surfaces from a set of irregularly distributed points using triangulation networks was conducted. Based on the identified drawbacks of existing solutions, a software tool was developed, utilizing Delaunay triangulation to create a mesh of triangles and visualize 3D surfaces. The project incorporates the Delaunator and Three.js libraries, along with the Electron framework, for creating a cross-platform user interface.

**СОДЕРЖАНИЕ**

[ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ 9](#_Toc19756)

[ВВЕДЕНИЕ 10](#_Toc30399)

[1 Обзор предметной области 12](#_Toc22223)

[1.1 История и понятие реконструкции поверхности 12](#_Toc28473)

[1.2 Принцип реконструкции поверхности 14](#_Toc20315)

[1.2 Область применения 15](#_Toc31049)

[1.2.1 Геодезия и картография 15](#_Toc2356)

[1.2.2 Архитектура и городское планирование 15](#_Toc10817)

[1.2.3 Геология и науки о Земле 15](#_Toc9951)

[1.2.4 Медицинская визуализация и хирургия 15](#_Toc23338)

[1.2.5 Робототехника и автоматизация 16](#_Toc9097)

[1.2.6 Виртуальная и дополненная реальность 16](#_Toc12185)

[1.2.7 Искусство и культурное наследие 16](#_Toc23821)

[1.3 Сравнение аналогов 16](#_Toc7220)

[1.3.1 Принцип отбора аналогов 16](#_Toc4841)

[1.3.1.1 MeshLab 17](#_Toc12868)

[1.3.1.2 Blender 17](#_Toc10740)

[1.3.1.3 Autodesk 17](#_Toc24847)

[1.3.1.4 CloudCompare 18](#_Toc17127)

[1.3.2 Критерии сравнения аналогов 18](#_Toc8914)

[1.3.2.1 Сложность интерфейса 18](#_Toc25067)

[1.3.2.2 Цена 18](#_Toc31362)

[1.3.2.3 Платформа для использования 19](#_Toc6087)

[1.3.2.4 Эффективность 19](#_Toc4300)

[1.3.3 Выводы по итогам сравнения 20](#_Toc29166)

[2 Формулировка требований к решению и постановка задачи 22](#_Toc22767)

[2.1 Постановка задачи 22](#_Toc15186)

[2.2 Требования к решению 22](#_Toc27431)

[2.3 Новизна и практическая значимость работы 24](#_Toc16155)

[3 Описание метода решения 25](#_Toc8088)

[3.1 Стек используемых технологий 25](#_Toc5379)

[3.2 Реализация программы 26](#_Toc11342)

[3.2.1 Исходные данные программы 26](#_Toc2871)

[3.2.2 Создание сцены 27](#_Toc7625)

[3.2.3.1 Теория триангуляции Делоне 28](#_Toc16157)

[3.2.3.2 Алгоритмы триангуляции 29](#_Toc23350)

[3.2.4 Оптимизация рендеринга 32](#_Toc9500)

[3.2.5 Градиентные поверхности 33](#_Toc20870)

[3.2.5.1 Создание цветовой карты 33](#_Toc16357)

[3.2.5.2 Cоздание материала с помощью настраиваемого шейдера 34](#_Toc9249)

[3.2.5 Контроль сцены 35](#_Toc2770)

[3.2.6 Реализация интерфейса 36](#_Toc30789)

[3.2.6.1 Импорт данных 37](#_Toc30794)

[3.2.6.2 Управление цветом сцены 37](#_Toc22613)

[3.2.6.3 Настройки освещения 37](#_Toc11349)

[3.2.6.4 Другие параметры 38](#_Toc22177)

[3.2.6.5 Настройки координатных осей 38](#_Toc7365)

[3.2.6.6 Оконный интерфейс 38](#_Toc19352)

[3.2.6.7 Другие информации 39](#_Toc10322)

[3.3 Упаковка программы. 39](#_Toc13699)

[4 Исследование разработанной программы 41](#_Toc19200)

[4.1 Характеристики разработанной программы 41](#_Toc19549)

[4.2 Тестирование на реальные данные 41](#_Toc27078)

[4.2.1 Землетрясения в Японии 41](#_Toc31823)

[4.2.2 Землетрясения в Греции 42](#_Toc4567)

[4.2.3 Температура в Индии 43](#_Toc392)

[4.2.4 Население городов США 44](#_Toc6793)

[4.3 Стратегии для совершенствования разработанного инструмента 46](#_Toc27772)

[4.3.1 Поддержка квадратичных сеток 46](#_Toc24626)

[4.3.2 Другие типы интерполяция в треугольниках 47](#_Toc18193)

[4.3.3 Веб-хостинг приложения 47](#_Toc4195)

[4.3.4 Функции экспорта 48](#_Toc11984)

[5 Обеспечение качества разработки программного продукта 50](#_Toc18386)

[5.1 Понятие качества 50](#_Toc13727)

[5.2 Операциональные определения характеристик качества 55](#_Toc16557)

[5.3 Методы выявления требований потребителей 56](#_Toc25934)

[5.4 Расчёт быстродействия программного обеспечения 58](#_Toc7290)

[ЗАКЛЮЧЕНИЕ 60](#_Toc25721)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 61](#_Toc28135)

[ПРИЛОЖЕНИЕ А 63](#_Toc28556)

## ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

ЯП – язык программирования;

IDE – Integrated Development Environment;

JSON – JavaScript Object Notation;

3D – Third Dimension;

Рендеринг – термин в компьютерной графике, обозначающий процесс получения изображения по модели с помощью компьютерной программы;

Шейдер – компьютерная программа, предназначенная для исполнения процессорами видеокарты (GPU);

JSON – JavaScript Object Notation

TXT – Text file;

CSV – Comma-separated values.

## ВВЕДЕНИЕ

В современном мире программы для реконструкции трёхмерных моделей становятся все более востребованными в различных областях, таких как инженерное проектирование, медицина, архитектура и развлекательная индустрия. Эти программы позволяют пользователям создавать, редактировать и взаимодействовать с трёхмерными объектами, обогащая их работу множеством возможностей. Одной из ключевых проблем при создании таких программ является эффективное отображение характеристик объектов. Возникает необходимость представления различных данных о объектах, таких как их положение, размеры, форма, цвет и текстура. Пользователи ожидают возможности взаимодействия с этими данными и получения всей необходимой информации в удобном для них формате.

**Цель работы** – разработка программного обеспечения для восстановления трёхмерной поверхности в плоской области для разных измеренных признаков.

Для достижения этой цели ставятся следующие задачи:

1. Поиск и обзор существующих решений;
2. Формирование требований к программному обеспечению;
3. Реализация программного обеспечения;
4. Тестирование программы на реальные данные.

**Объектом исследования** является трёхмерная реконструкция поверхности.

**Предметом исследования** является программное обеспечение для восстановления поверхности в плоской области с использованием триангуляционных сетей.

**Практическая ценность работы** заключается в возможности интеграции разработанных программных средств в существующие веб-приложения для визуализации трёхмерных моделей. Это позволит пользователям получать доступ к разнообразной информации о трёхмерных объектах и удобно взаимодействовать с ними.

## 1 Обзор предметной области

### 1.1 История и понятие реконструкции поверхности

С начала развития фотограмметрии в конце 19 века возникла концепция трёхмерной реконструкции. Научно-техническая дисциплина фотограмметрия направлена на извлечение из фотографий геометрической информации. В 1890-х годах фотограмметрию использовали для таких отраслей как геодезия и картографирование. Анализ стереопара изображений помогает измерить расстояние и создать 3D-модель ландшафта и сооружений. Такой метод закладывает основу для основных принципов 3D-реконструкции. С появлением в середине 20-го века компьютеров, появились уникальные возможности для создания 3D-реконструкции. Основная движущая сила прогресса в области 3D-реконструкции – компьютерное зрение, как область, направленная на обучение компьютеров интерпретации визуальной информации. Учёные начали изучение использования цифровых изображений для создания 3D-моделей, отдавая предпочтение следующим тематикам: стереосогласование и сопоставление объектов; определение границ.

В периоде с 1970-го по 1980-е года исследователями были разработаны алгоритмы, направленные на извлечение информации из изображений о глубине, а также соотнесение стереоизображений. Произошло образование методов "Структуры из движения" (SfM), которые позволили создавать реконструкции 3D-сцены с помощью последовательности изображений путём анализирования между ними относительного движения. Также, в этом временном периоде 3D-реконструкция нашла своё применение в области робототехники, где для взаимодействия и навигации с окружающей средой роботы могли использовать зрение.

В конце 20-го года, следующим по значимости, стало достижение по внедрению технологий лазерного сканирования. К таким технологиям отнесли лазерные сканеры,а также систему обнаружения света и определения дальности, которые получили своё применение в лесном хозяйстве, охране культурного наследия и топографиях для получения 3D данных. Эти инструменты используются для генерации плотных облаков точек, которые обрабатываются для создания высокодетализированных 3D-моделей.

В это период доступными становятся программные платформы для компьютерной графики и 3D-реконструкции.

Технологии OpenGL предъявили стереотипированный способ для создания 3D-графики и координации ею, пока библиотеки, такие как OpenCV, предлагают набор инструментов для решения задач компьютерного зрения, в том числе 3D-реконструкции.

XXI век – стал веком развивающегося машинного и глубокого обучения, которое произвело революцию в 3D-реконструировании. Модели глубокого обучения, а также сверточные нейронные сети (CNN) дали возможность для использования более сложных и точных методов 3D-реконструкции.

Такие методы позволили обрабатывать крупные наборы данных для автоматических извлечений объектов и создания 3D-моделей с исключительным вмешательством человеческого ресурса.

Углублённое изучение также привело к использованию новых подходов в 3D-реконструкции: генеративных состязательных сетей (GAN) и вариационных автоэнкодеров (VAEs), способных генерировать уникальные реалистичные 3D-модели в составе из зашумленных и неполных данных. Данные технологии применяются в медицинской визуализации, где точные 3D-реконструкции, исходя из результатов КТ и МРТ-сканирования имеющих для диагностики и планирования лечения, решающее значение.

### 1.2 Принцип реконструкции поверхности

Реконструкция поверхности является процессом создания 3D моделей объекта или сцены, основываясь на данных, которые были собраны из различных источников информации, зачастую представленные в некорректной форме в двумерном пространстве.

В числе основных методов реконструкции поверхности главным является применение триангуляционных сеток. Настоящий подход подразумевает формирование сетки, состоящей из треугольников, которая соединяет точки данных на плоскости, а также модифицирует преобразование этой сетки в трёхмерную структуру. Триангуляционные сетки гарантируют высокий уровень детализации и точности, при элементарном потреблении ресурсов.

В 1970-х годах впервые были выдвинуты триангуляционные сетки, но ввиду сложных алгоритмов и ограниченности вычислительной мощности не смогли найти широкого применения. В последние несколько лет, в связи с появлением более высокопроизводительных компьютеров и современных алгоритмов, данные сетки стали основой современных методов реконструкции поверхности.

Процесс реконструкции поверхности состоит из нескольких ключевых этапов. Первый этап включает в себя сбор данных из набора точек, которые были получены при помощи сканирования, GPS и других измерительных методов. На втором этапе происходит обработка данных для устранения различных отклонений и шумов. Третий этап начинается с создания триангуляционной сетки, которая соединяет точки в плоском пространстве. На заключительном этапе происходит преобразование плоской сетки в трёхмерное представление высоты и прочих характеристик.

Современные технологические достижения, компьютерное зрение и машинное обучение позволяют повысить скорость и точность, несмотря на давность разработки основных аспектов реконструкции поверхности, включающих в себя триангуляцию, сбор данных и преобразование в 3D.

Современные методы помогают использовать сложные алгоритмы для повышения автоматизации различных аспектов, позволяющих создавать реалистичные и точные модели.

### 1.2 Область применения

**1.2.1 Геодезия и картография**

Ключевую роль в картографии и геодезии играет система реконструкции поверхности с использованием триангуляционных сеток, позволяющая создать подробную топографическую карту и цифровую модель местности, которая будет способствовать наиболее качественному анализу, быстрому развитию инфраструктуры, планированию землепользования, а также мониторингу окружающей среды.

**1.2.2 Архитектура и городское планирование**

Важную роль в городском планировании и архитектуре играет реконструкция поверхности, основанная на триангуляции. С помощью преобразования 2D-архитектурных планов в 3D-модели, проектировщики и архитекторы имеют возможность для правильной визуализации и оценивания проектов в виртуальной среде, тем самым помогая обеспечить точное и эффективное строительство, а также использование пространства.

**1.2.3 Геология и науки о Земле**

Триангуляционные сетки, при изучении геологических образований используются для создания линий разломов, горных пород и трёхмерных изображений ландшафтов. Это является незаменимой помощью геологам, для анализа рельефа и понимания геологических процессов, что в результате помогает в разведке ресурсов, а также оценке предполагаемой опасности.

**1.2.4 Медицинская визуализация и хирургия**

В области медицины реконструкция поверхности помогает создавать 3D-модели на основе данных медицинской визуализации, таких как КТ и МРТ. Это облегчает планирование операции, позволяя хирургам виртуально исследовать анатомические структуры и готовиться к сложным процедурам, снижая риски во время операции.

**1.2.5 Робототехника и автоматизация**

Реконструкция поверхности полезна в робототехнике, где 3D-модели используются для программирования движений роботов и моделирования взаимодействия с окружающей средой. Этот подход помогает при проектировании и тестировании автоматизированных систем для производства, логистики и даже автономных транспортных средств.

**1.2.6 Виртуальная и дополненная реальность**

Триангуляционные сетки являются ключевым компонентом приложений виртуальной и дополненной реальности. Реконструируя 3D-среду на основе 2D-данных, разработчики могут создавать захватывающие виртуальные миры и накладывать цифровую информацию на реальный мир, открывая новые возможности для игр, образования и тренингов.

**1.2.7 Искусство и культурное наследие**

Реконструкция поверхности позволяет сохранять объекты культурного наследия и произведения искусства в цифровом формате. Благодаря созданию 3D-моделей исторических памятников, статуй и артефактов эти сокровища могут быть сохранены в цифровом виде, что позволяет проводить реставрационные работы и осуществлять вир.

**1.3 Сравнение аналогов**

#### 1.3.1 Принцип отбора аналогов

В современных условиях развития технологий 3D-моделирование играет все более важную роль в различных отраслях, таких как архитектура, строительство, инжиниринг, геология и других. В основе большинства 3D-моделей лежит триангуляция, которая позволяет создавать трёхмерные поверхности из точек, расположенных в пространстве. В зависимости от способа реализации такой технологии могут использоваться различные программы и методы.

Для сравнения и оценки эффективности различных подходов к 3D-реконструкции поверхностей была проведена выборка программ, наиболее отличающихся друг от друга с точки зрения реализации и методологии. Эти похожие решения используются в различных областях и применяют различные методы для создания триангуляционных моделей.

#### 1.3.1.1 MeshLab

Одно из первых решений для работы с триангуляцией в 3D-пространстве, MeshLab [1], представляет собой программное обеспечение с открытым исходным кодом, используемое для обработки и редактирования сеток. Оно особенно популярно в научной и академической среде, а также среди энтузиастов 3D-моделирования. MeshLab позволяет импортировать различные типы данных, включая файлы точек, и предоставляет инструменты для создания и обработки триангуляционных сеток. Однако это программное обеспечение имеет определённые ограничения в интерфейсе, которые могут усложнить работу новичкам.

#### 1.3.1.2 Blender

Blender [2] - ещё одно широко используемое программное обеспечение с открытым исходным кодом, но с более широкими возможностями, включая анимацию, рендеринг и, конечно же, создание 3D-моделей из точек. Blender поддерживает различные методы триангуляции, что делает его универсальным инструментом для 3D-реконструкции. Несмотря на свою гибкость и богатство функций, Blender может быть легко освоен, особенно теми, кто не знаком с более сложными 3D-инструментами.

#### 1.3.1.3 Autodesk

Autodesk ReCap [3] - это коммерческое программное обеспечение, предназначенное для обработки облаков точек и реконструкции 3D-моделей. Это решение широко используется в таких профессиональных областях, как архитектура и строительство, где важна интеграция с другими продуктами Autodesk. ReCap позволяет создавать триангуляционные сетки и комбинировать данные из разных источников. Однако коммерческий характер продукта означает, что его использование может быть дорогостоящим.

#### 1.3.1.4 CloudCompare

CloudCompare [4] - это ещё один инструмент с открытым исходным кодом, специализирующийся на обработке облаков точек и создании триангуляционных поверхностей. Он позволяет сравнивать облака точек и создавать различные типы визуализаций. CloudCompare часто используется в академической и научной среде благодаря своей гибкости и открытости. Однако, как и в случае с MeshLab, это может быть сложно для начинающих пользователей из-за отсутствия интуитивно понятного интерфейса.

#### 1.3.2 Критерии сравнения аналогов

#### 1.3.2.1 Сложность интерфейса

Когда речь заходит о программном обеспечении для 3D-реконструкции поверхности, сложность интерфейса является важным критерием. Этот критерий включает в себя удобство взаимодействия пользователя с программой, интуитивно понятную навигацию, а также простоту выполнения основных функций. Если интерфейс слишком сложный или требует много времени для освоения, это может снизить производительность и сделать программу менее привлекательной для пользователей.

#### 1.3.2.2 Цена

Цена является ключевым фактором при выборе программы. Этот критерий учитывает стоимость лицензии или подписки, а также возможные дополнительные расходы на обновления или техническую поддержку. Более доступные программы могут быть предпочтительнее, особенно для небольших команд или индивидуальных пользователей. Однако низкая цена может означать ограничения в функциональности или поддержке.

#### 1.3.2.3 Платформа для использования

Этот критерий подразумевает совместимость программы с различными операционными системами и устройствами. Чем шире спектр поддерживаемых платформ, тем больше возможностей для использования программы в различных средах. Некоторые программы могут быть доступны только для определённых операционных систем, что ограничивает их универсальность. Также важно учитывать, есть ли у программы версии для мобильных устройств, которые могут быть полезны для работы на ходу.

#### 1.3.2.4 Эффективность

Производительность - ещё один важный критерий при выборе программы для 3D-реконструкции поверхностей. Она включает в себя скорость обработки данных, оптимизацию использования ресурсов (таких как процессор и память), а также способность программы работать с большими объёмами данных. Если программа работает медленно или потребляет много ресурсов, это может негативно сказаться на эффективности работы.

Для разработки будущего решения, важно учитывать баланс между сложностью интерфейса, платформой использования и производительностью. В соответствии с вышеуказанными критериями была составлена сравнительная таблица (в таблице 1) вышеперечисленных аналогов.

Таблица 1 – Сравнение аналогов по критериям

|  |  |  |  |
| --- | --- | --- | --- |
| Аналоги | Сложность интерфейса | Стоимость лицензии | Документация |
| MeshLab | Средняя до высокой: Сложный интерфейс, менее интуитивный для новичков | Бесплатно | Ограниченная документация, много сторонних ресурсов |
| Blender | Высокая: Крутая кривая обучения из-за множества функций и опций | Бесплатно | Обширная документация, большая сообщество |
| Autodesk ReCap | Низкая до средней: Удобный интерфейс, ориентирован на пользователя | Подписка (дорогая) | Хорошая официальная документация |
| CloudCompare | Средняя до высокой: Менее интуитивный интерфейс, требует обучения | Бесплатно | Ограниченная официальная документация, много сторонних ресурсов |

#### 1.3.3 Выводы по итогам сравнения

Несмотря на распространённость различных программных решений, их также можно разделить на несколько категорий в зависимости от их подхода к реализации 3D-реконструкции. Вот несколько основных типов:

* Программы с ограниченной функциональностью, предназначенные для выполнения конкретных задач (например, MeshLab, предназначенный для базового редактирования 3D-сеток).
* Программы с широкой функциональностью, способные выполнять широкий спектр задач в 3D-пространстве (например, Blender, который можно использовать для моделирования, анимации и рендеринга).
* Профессиональные коммерческие решения, предназначенные для корпоративного использования и интеграции с другими инструментами (например, Autodesk ReCap, часто используемыми в профессиональных контекстах).

Каждая из этих категорий имеет свои преимущества и недостатки. Программы первой категории могут быть просты в использовании, но часто имеют ограниченные возможности. Программы второй категории более универсальны, но могут быть сложными для начинающих из-за сложного процесса обучения. Профессиональные решения третьей категории часто отличаются высокой производительностью и интеграцией, но могут быть дорогостоящими.

Анализируя такие критерии, как сложность интерфейса, цена, наличие документации и производительность, можно сказать, что часто предпочтение отдаётся универсальным программам, которые охватывают широкий спектр задач, особенно если они сочетают в себе доступность, интуитивность и производительность.

## 2 Формулировка требований к решению и постановка задачи

### 2.1 Постановка задачи

Необходимо разработать и внедрить программу, поддерживающую следующие функции:

* Ввод и обработка данных,
* Построение триангуляции и поверхности исследуемого признака,
* Визуализация данных,
* Создание удобного интерфейса,
* Упаковка и дистрибуция программы.

### 2.2 Требования к решению

После тщательного анализа аналогичных программ 3D-реконструкции были выявлены общие недостатки каждой программы, а именно неинтуитивный интерфейс и относительно ограниченная документация, что значительно снижает удобство работы пользователя. Тем не менее, само собой разумеется создать приложение, отвечающее этим требованием. Данная программа будет называться Surf3D. Также будут предусмотрены другие необходимые функции, такие как:

* Упрощённое использование. Программа должна быть простой в использовании, чтобы пользователь не заблудился, как в некоторых профессиональных программах. Визуализация поверхности должна быть получена в несколько кликов.
* Эффективность. Будущий продукт должен быть высокопроизводительным и оптимизированным, чтобы даже большие данные обрабатывались довольно быстро, а реконструкция поверхности выполнялась за миллисекунды.
* Поддержка различных форматов данных. Программа должна иметь возможность импортирования данных в различных форматах, которые достаточно доступны и просты в использовании и редактировании.
* Качество результатов. Программа для восстановление поверхности должна обладать высоким качеством поверхности, сводя к минимуму шумы, искажения и к неправильным измерениям, чтобы пользователь мог получать правильные и надёжные результаты для лучшей интерпретации.
* Документация и поддержка. Программа должна сопровождаться документацией, доступной как сразу в программе, так и во внешних источниках. Она должна быть простой и содержать всю информацию, необходимую для правильного использования программы, о различных типах файлов которые принимает программа, а также о разных возможных типовых ошибках и их устранения.

### 2.3 Новизна и практическая значимость работы

Большинство программ для реконструкции 3D поверхностей безусловно эффективны и оснащены мощными алгоритмами, но относительно дороги в приобретении (лицензии) и не имеют достаточно богатой документации.

Научная новизна этой работы заключается в разработке производительной программы, основана на интернет технологий, которая способна быстро реализовать трёхмерные поверхности из неравномерно расположенных точек в плоской области с отличной документацией и с открытым исходным кодом чтобы развивать проект желающим.

## 3 Описание метода решения

### 3.1 Стек используемых технологий

Средой разработки выбранной для реализации программ восстановления поверхности является Visual Studio Code [5]. Знаменитый фреймворк (IDE) был выбран за широкий спектр функциональных возможностей, его упрощённый интерфейс, а также за личные предпочтения.

Основным языком программирования, используемым для реализации программы Surf3D, является JavaScript. Он был выбран для разработки программы, чтобы в будущем её можно было адаптировать как веб-решение, которое можно будет использовать непосредственно в онлайном режиме через браузер. На данный момент программа будет разрабатываться только как кроссплатформенная программа для компьютеров.

Основной стек используемых технологий:

* JavaScript [6] – основной язык программирования для всей логики программы, манипулирования данными и реализации интерфейса.
* Three.js [7] – 3D библиотека для графического отображения поверхностей, полученных путём реконструкции.
* Delaunator [8] – Библиотека на Javascript для быстрой и эффективной триангуляции набора точек.
* Electron.js [9] – Библиотека для создания интерфейса программы и обеспечения мультиплатформенного аспекта программы, основана на веб-языках HTML, CSS и Javascript.
* Electron Forge [10] – Инструмент для автоматизации сборки, упаковки и развёртывания программы Electron.

### 3.2 Реализация программы

#### 3.2.1 Исходные данные программы

Для того чтобы разработать программу восстановления поверхности, нужно определить форму данных для дальнейшего использования.

Коллекция измерений представляет собой набор точек, характеризующихся тремя величинами: абсцисс ***x*** и ординат ***y*** в области исследования, а также величина ***z*** которое является значением исследуемого признака в точке (x, y) области испытания. Это может быть например топографические данные о высоте горы, температура поверхности океана в определённой области, или данные о демографии в какой то регионе. Для того чтобы впоследствии было легче манипулировать данными, необходимо создать два массива.

Первый набор хранит точки в двумерном виде, т.е. абсцисс и ординат. Из этого набора точек с помощью нужной библиотеки можно будет получать триангуляцию Делоне для этих точек, а затем использовать результат для восстановления поверхности.

Второй набор уже в трёхмерном виде для того чтобы уже сделать восстановление поверхности и визуализации в сцене с помощью библиотеки Three.js.

#### 3.2.2 Создание сцены

Чтобы визуализировать поверхности или любые объекты, нужно создать сцену. Сцена представляет собой контейнер который хранит в себе 3D объекты. Это виртуальное место где будут располагаться элементы и освещение (рис. 2). Сцена также должна содержать камеру, потому что именно от неё зависит конечный рендеринг, то что важное пользователю.

Используя объект **Scene** библиотеки **Three.js**, будет создана сцена где будет хранится поверхность, и камера, а также их инициализация. Пример инициализации сцены представлен ниже (рис. 1)



Рисунок 1 – Пример кода для создания и инициализации сцены и камеры

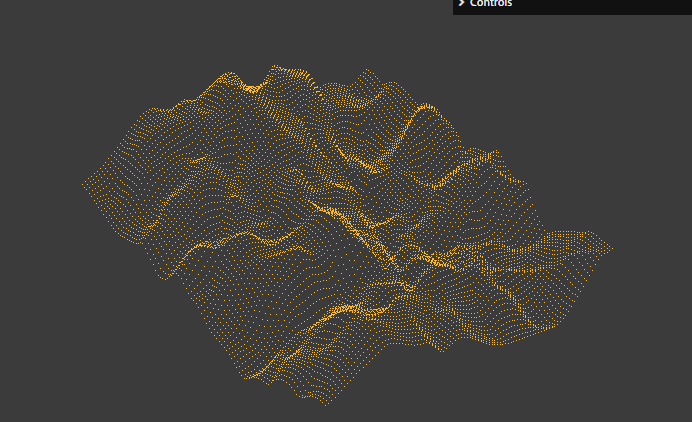


Рисунок 2 – Пример сцены в которой расположены точки

**3.2.3 Построение триангуляционных сеток и поверхностей**

#### 3.2.3.1 Теория триангуляции Делоне

Для создания быстрой и эффективной триангуляции была выбрана триангуляция Делоне [11]. При разработке программы использовалась библиотека **Delanautor**, которая достаточно хорошо оптимизирована и эффективна.

Триангуляция Делоне может быть определена как набор треугольников, в которой каждый треугольник удовлетворяет условию что окружность, описывающая треугольник, не содержит точек, других от самых вершин треугольника. Благодаря этому свойству мы имеем справедливое распределение углов в триангуляции для уменьшения острых углов.

#### 3.2.3.2 Алгоритмы триангуляции

Библиотека **Delaunator** использует достаточно эффективные методы для создания триангуляции Делоне [12, 13, 14]:

* **Алгоритм Бойера-Уотсона:** Итерационный алгоритм начинается с построения супер-треугольника, охватывающего все точки. Затем на каждой итерации он вставляет одну точку и удаляет все треугольники, которые не удовлетворяют свойству триангуляции Делоне. Иногда также необходимо «перевернуть» некоторые ребра, если нарушено правило окружности.
* **Разделение-рекурсия:** Метод, известный своей простотой и эффективностью, заключается в разбиении множества точек на подмножества для построения небольших триангуляций, которые затем объединяются для получения конечной поверхности.

**3.2.3.3 Имплементация**

Для любого набора точек, преобразуются точки в виде длинного массива для построения триангуляции (рис. 3).

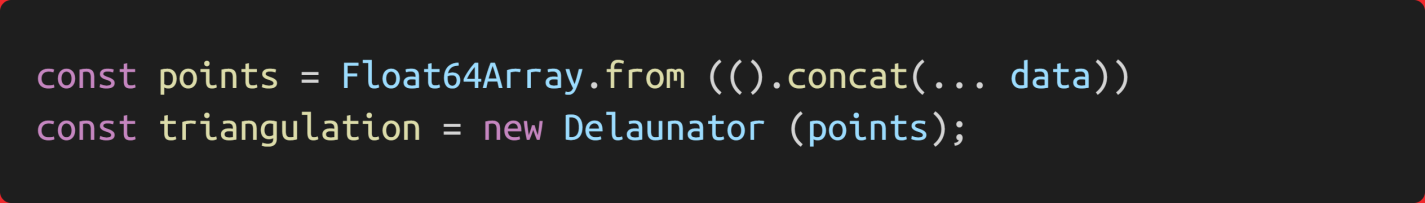


Рисунок 3 – Построение триангуляции Делоне с помощью Delaunator.js

После создания триангуляции, есть теперь информация о том, как построить поверхность. Чтобы представить объект с помощью Three.js, нужны две важные компоненты: геометрия и материал. Геометрия - это форма объекта, которая характеризуется точками, определяющими её. Чтобы создать геометрию поверхности, будут храниться точки в векторе **Vec3** от библиотеки **Three.js**. Для этого выполняется цикл, который будет перебирать все элементы полученной триангуляции. На каждой итерации мы добавляем координаты в наш вектор. Порядок очень важен, так как от него зависит достоверность поверхности.

Геометрия создаётся из вектора с помощью метода setFromPoints в классе **BuffeGeometry**. Этот класс был оптимизирован для эффективной работы с большим количеством точек. Далее нам нужно реализовать логику нормалей поверхности. Нормаль – это вектор, перпендикулярный поверхности, который используется для создания эффекта отражения при наличии освещения сцены. Для этого в библиотеке предусмотрен метод, выполняющий эту роль: **метод computeVertexNormals**.

С другой стороны, чтобы этот эффект мог сработать, необходимо предварительно выбрать материал, который, очевидно, позволяет проводить рефлексию. Если бы это было не так, реконструированная поверхность была бы лишена деталей, и было бы очень трудно интерпретировать результаты. Материалом, выбранным для рендеринга является **MeshPhongMaterial.**

С помощью геометрии и материалу, наконец, строится поверхность типа **Mesh**. Затем поверхность добавляется в сцену для рендеринга (рис. 4, 5).

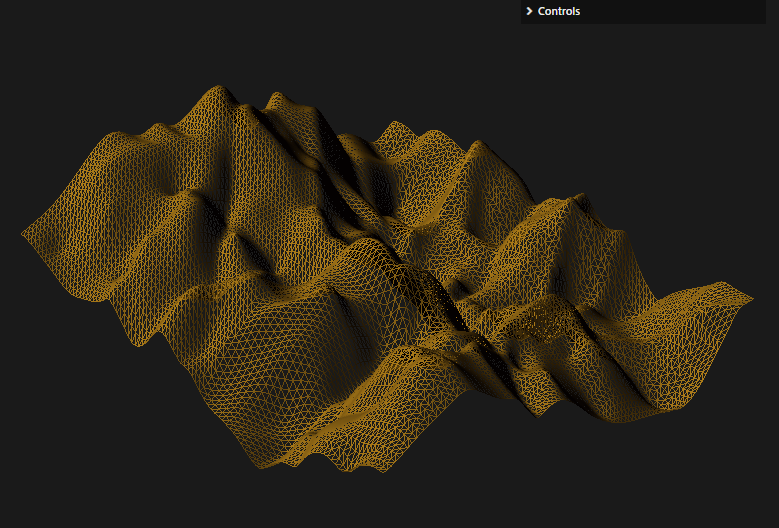
****

Рисунок 4 – Полученный результат после построения треугольников

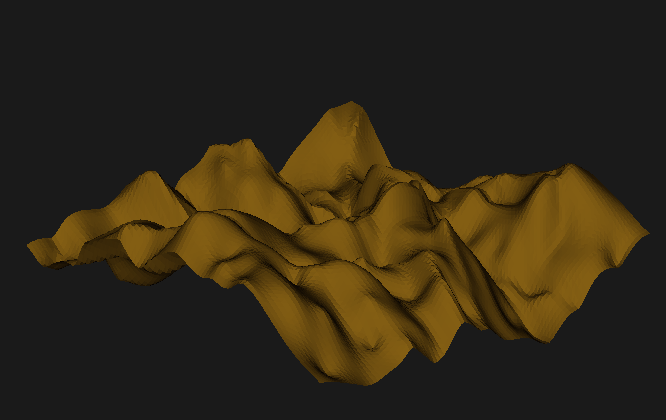


Рисунок 5 – Полученный результат после построения поверхности

При создании сетки **mesh** и материала **material** с Three.js библиотека поддерживает внутренние процессы, которые обеспечивают точные вычисления и интерполяцию для оценки значений координат для точек внутри каждого треугольника. Это позволяет **Three.js** создавать гладкие и визуально согласованные поверхности. Однако, учитывая, что Three.js использует только линейную интерполяцию, результаты не всегда могут быть оптимальными. Чтобы получить более гладкие поверхности и более естественные переходы, будет использована интерполяция другой формы, как кубическая, квадратичная или сплайн, для уточнения деталей и увеличения качества рендеринга.

#### 3.2.4 Оптимизация рендеринга

В сложных трёхмерных геометриях часто встречаются вершины, которые почти идентичны, но технически считаются разными. Эти небольшие различия могут вызывать визуальные артефакты или разрывы в отображении поверхности, создавая « швы » или резкие переходы.

Метод **BufferGeometryUtils.mergeVertices** принимает геометрию в качестве входных данных и объединяет вершины, которые находятся очень близко друг к другу. Этот процесс уменьшает количество вершин в геометрии, что также может повысить эффективность вычислений рендеринга, обеспечивая при этом более гладкие и однородные поверхности (рис. 6).

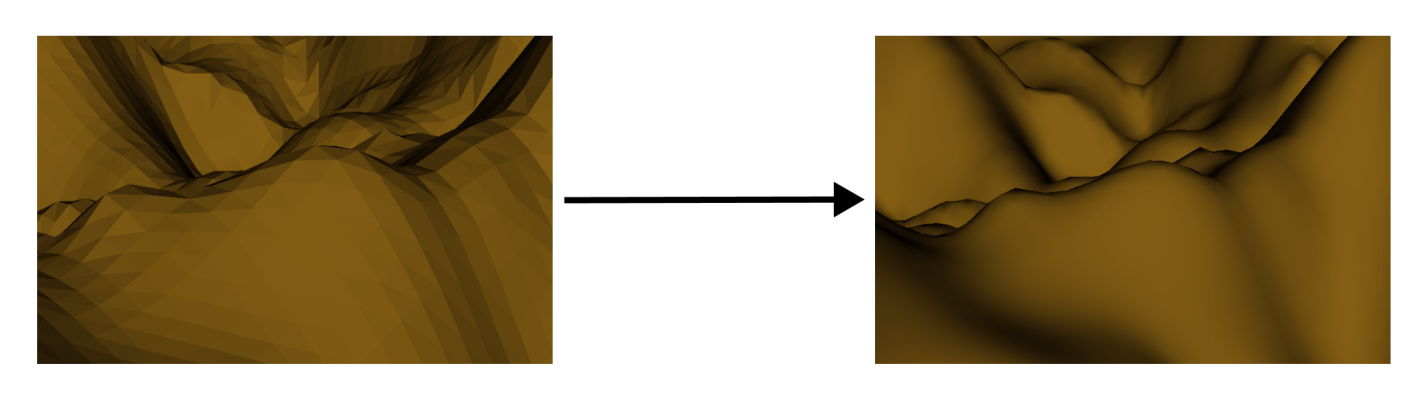


Рисунок 6 – Оптимизация рендеринга

**3.2.5 Градиентные поверхности**

Чтобы создать эффект градиентного цвета, отражающий высокие и низкие значения трёхмерной поверхности, в коде используются пользовательские шейдеры в Three.js. Цель состоит в том, чтобы сопоставить высокие и низкие значения (например, высоты или температуры) с определёнными цветами, создавая визуальный цветовой градиент.

#### 3.2.5.1 Создание цветовой карты

Первым шагом является создание « цветовой карты », которая определит цветовой градиент, используемый для затемнения. Код использует элемент **canvas** для генерации градиента (рис. 7).

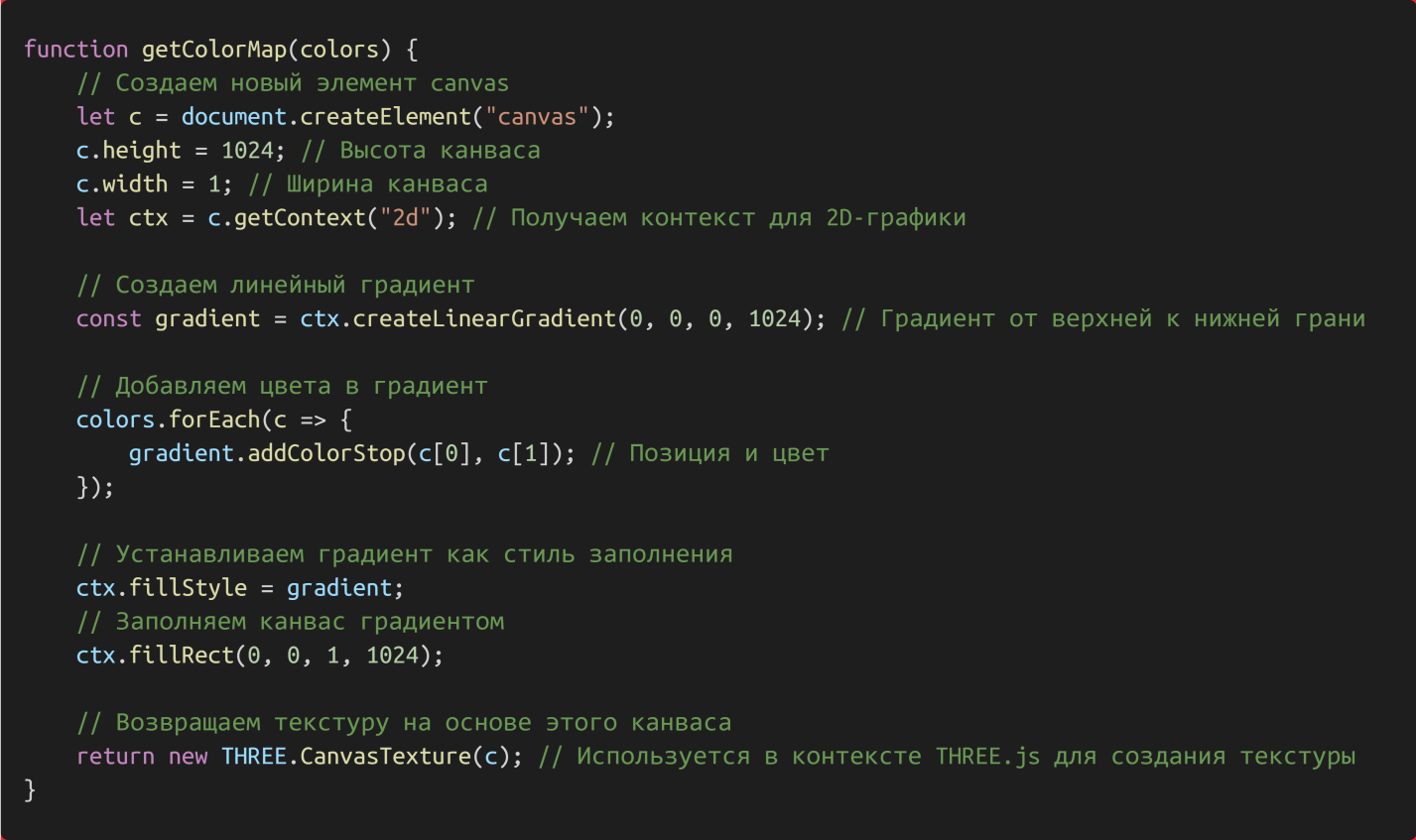
****

Рисунок 7 – Cоздание карты

**Canvas** используется для создания линейного градиента высотой 1024 пикселя. Цвета добавляются в определённые позиции для создания желаемого градиента.

**Gradient** используется для создания **Three**.**CanvasTexture**, который будет использоваться в шейдере материала для отображения цвета.

#### 3.2.5.2 Cоздание материала с помощью настраиваемого шейдера

В материале, используемом для сетки **mesh**, используется **THREE.MeshPhongMaterial** с настраиваемыми шейдерами. Эти шейдеры позволяют изменять поведение рендеринга по умолчанию для добавления цветовых эффектов на основе высоты (рис. 8). Полученный эффект представлен ниже (рис. 9).

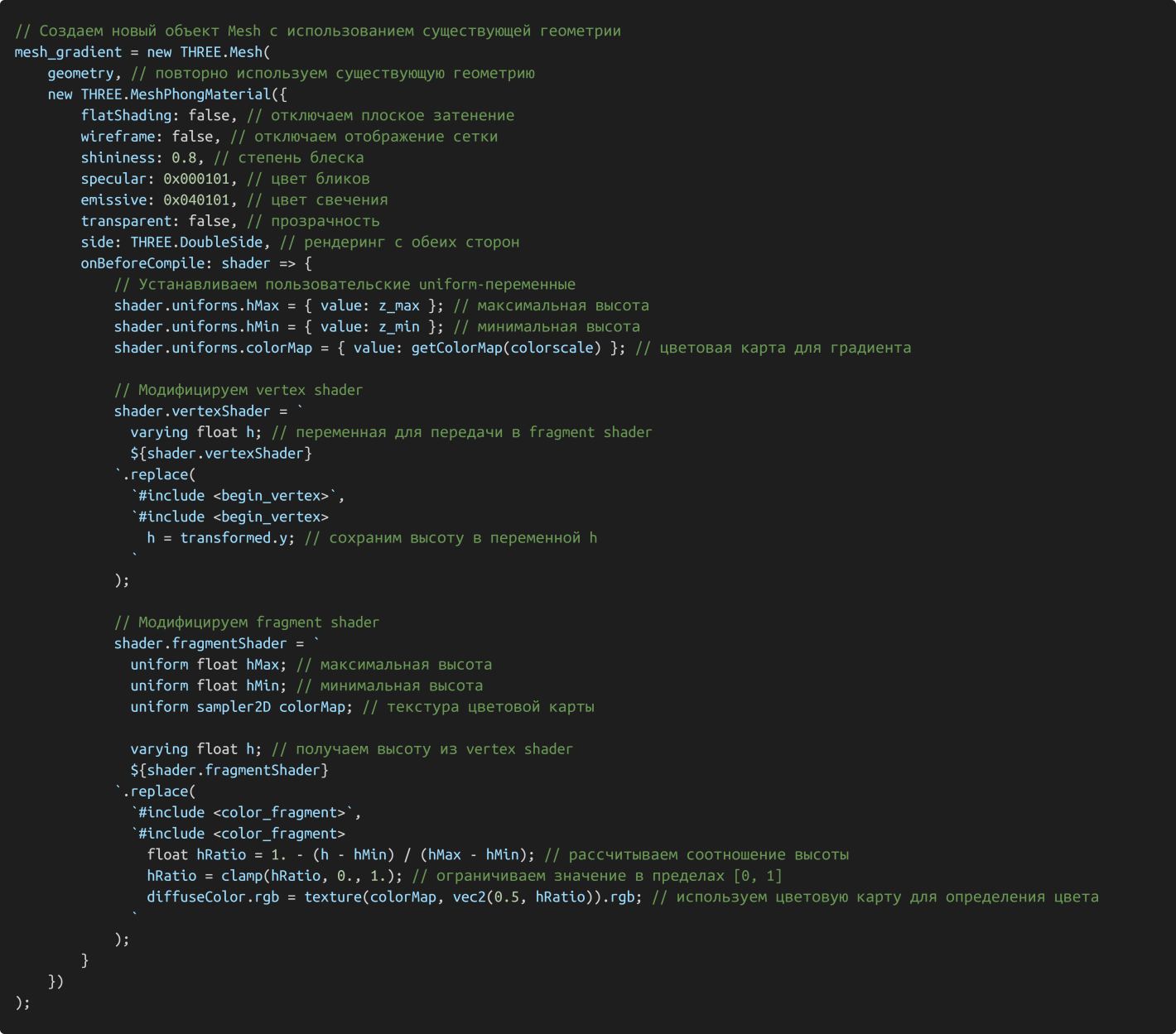


Рисунок 8 – Создание поверхности с градиентного цвета

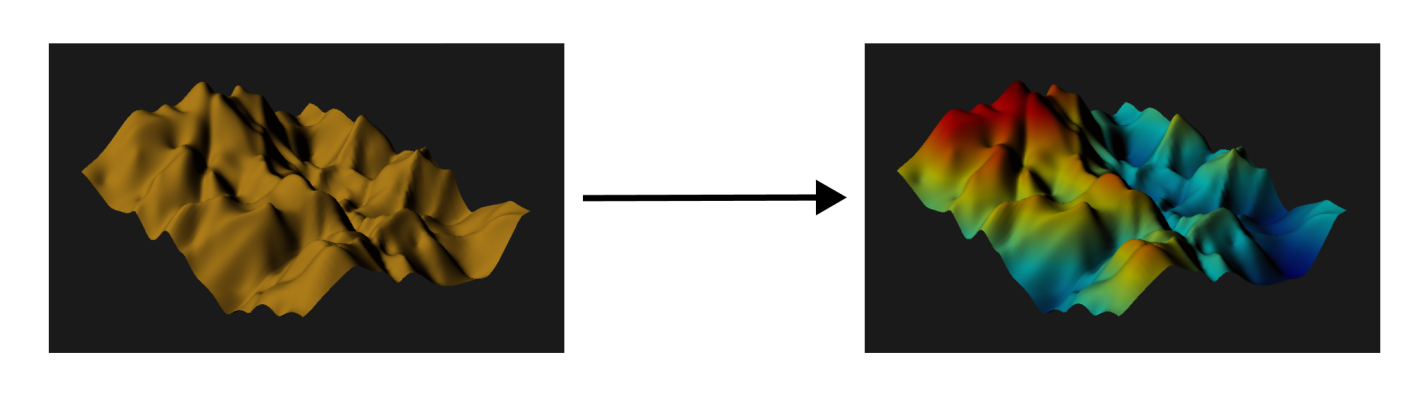
****

Рисунок 9 – Градации цвета

#### 3.2.5 Контроль сцены

Для контроли камеры в процессе визуализации результатов, был использован плагин **OrbitControls** [15]. Он позволяет управлять камерой в 3D-сцене с помощью простых взаимодействий, таких как масштабирование, поворот и перемещение. Он даёт возможность интуитивно взаимодействовать со сценой, что улучшает взаимодействие с пользователем.

Настройка камеры и средства визуализации чтобы использовать **OrbitControls**, вам понадобятся уже настроенные камера и средство визуализации. **OrbitControls** объединяется с камерой и элементом DOM (средством визуализации) для управления взаимодействиями.

После настройки камеры и средства визуализации создаётся экземпляр **OrbitControls**. Этот экземпляр принимает камеру в качестве первого аргумента, а **domElement** средства визуализации в качестве второго аргумента. Элемент **domElement** - это часть DOM, которая реагирует на взаимодействия с пользователем, такие как щелчки, движения мыши и прокрутка. Пример инициализации плагина представлен ниже (рис. 10).

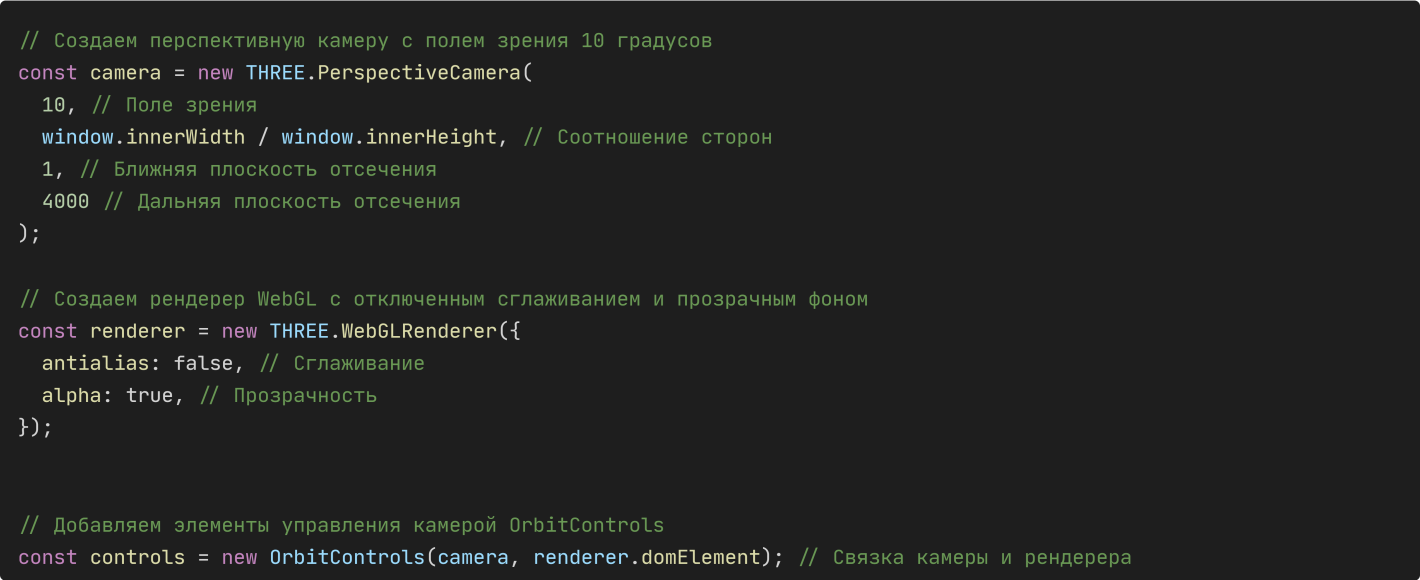


Рисунок 10 – Вывоз плагина OrbitControls

Чтобы **OrbitControls** реагировал на взаимодействия с пользователем, было реализовано обновление элементов управления в цикле анимации (рис. 11). Это гарантирует, что движения камеры обрабатываются в каждом кадре.

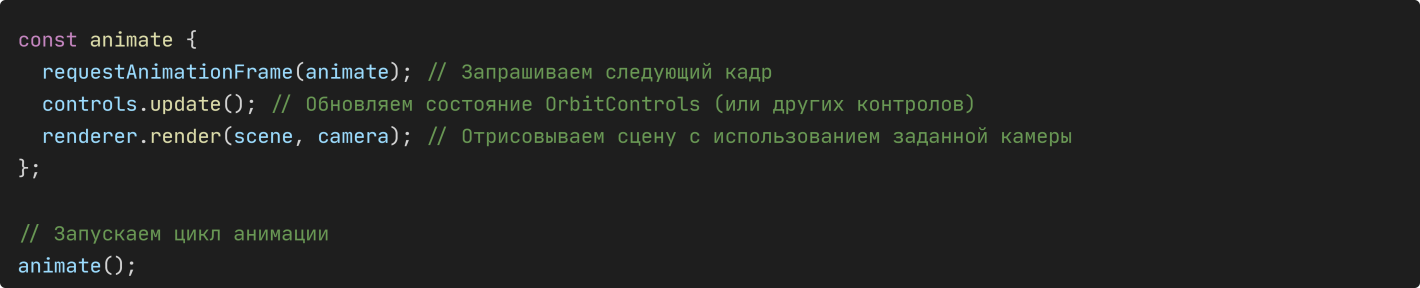
****

Рисунок 11 – Анимации сцены на действия пользователя

#### 3.2.6 Реализация интерфейса

Чтобы пользователи могли контролировать и настраивать различные аспекты 3D-сцены, программа использует библиотеку **lil-gui** [16] для обеспечения интуитивно понятного графического пользовательского интерфейса. Эта библиотека предназначена для обеспечения динамического и настраиваемого просмотра импортированных 3D-данных с рядом параметров конфигурации и взаимодействия.

#### 3.2.6.1 Импорт данных

Программа позволяет импортировать данные из трёх распространённых типов файлов: **json**, **txt**и **csv**. Эти файлы должны быть в определённом формате, чтобы приложение могло правильно их интерпретировать. В документации в разделе « Помощь / Справка » описан формат, необходимый для каждого типа файла. Если файл имеет неправильный формат, программа отобразит сообщение об ошибке, объясняющее, почему файл не удалось импортировать.

#### 3.2.6.2 Управление цветом сцены

Программа предлагает функции настройки цвета для улучшения восприятия поверхности и повышения удобства работы. Пользователь сможет изменить цвет сцены. В интерфейсе будет представлена цветовая палитра. Также можно будет изменить цвет исследуемой поверхности. Кроме того, пользователь может выбрать один из двух типов поверхности: монохромную или градиентную. Градиентные поверхности очень полезны для визуализации экстремальных значений некоторых данных.

#### 3.2.6.3 Настройки освещения

Что касается освещения, то программа предлагает несколько вариантов. Можно изменить положение освещения. Это необходимо, когда объект, представленный в 3D, находится не на пути источника света. Для этого предусмотрен интерфейс для управления положением источника света в пространстве.

#### 3.2.6.4 Другие параметры

#### Программа может отображать входные данные несколькими способами. Функция отображения данных в виде точек. Интерфейс позволяет настраивать размер и цвет этих точек. Также можно будет отображать поверхность с треугольниками или без них. Таким образом, можно будет просматривать только края поверхности (сетку). Это может быть интересно для лучшей интерпретации некоторых результатов.

#### 3.2.6.5 Настройки координатных осей

#### Координатные оси могут быть активированы или деактивированы. Они относительно ненавязчивы и используются для быстрого ориентирования в сцене. В некоторых случаях оси могут совпадать с самой поверхностью и искажать интерпретацию. Оси имеют три цвета: красный - для оси *x*, зелёный - для оси *y* и синий - для оси высоты.

#### 3.2.6.6 Оконный интерфейс

В программе предусмотрена панель меню с различными функциями. Кнопка « Перезапустить », которая обновляет приложение. Также имеется раздел « Отладка », позволяющий разработчикам быстро проанализировать различные проблемы, которые могут возникнуть в запущенном приложении. Также имеются кнопки « Масштаб по умолчанию », « Увеличить масштаб » и « Уменьшить масштаб », которые позволяют настроить масштаб отображения в программе в соответствии с вашими предпочтениями. Наконец, есть функция « Полный экран », позволяющая получить захватывающий рендеринг без границ, что может быть очень полезно, если вы хотите полностью погрузиться в интерпретацию результатов.

#### 3.2.6.7 Другие информации

Далее идут классические окна программы: окна справки и информации. Окно справки содержит все необходимые сведения о формате входных данных и советы по работе с программой. Также здесь есть FAQ, который поможет пользователю справиться с любыми проблемами. В окне «О программе» представлена информация о разработчике и используемых технологиях, а также их соответствующие версии.

#### 3.3 Упаковка программы.

Для упаковки проекта был использован компонент Electron Forge, специально разработанный для приложений на базе Electron. Он прост в использовании и преобразует исходный код в исполняемый файл, который можно установить без сложной настройки. При разработке была предусмотрена возможность кроссплатформенного распространения, чтобы пользователи разных платформ могли использовать программу без особых ограничений. Кроме того, это позволяет определить иконку для приложения.

## 4 Исследование разработанной программы

### 4.1 Характеристики разработанной программы

Приложения Electron.js и Three.js используют треугольные сетки для визуализации трёхмерных поверхностей с данными, которые можно импортировать из файлов JSON, TXT или CSV. С помощью OrbitControls пользователи могут настроить трёхмерную сцену, изменять цвета, регулировать освещение и управлять камерой. С помощью программы можно отображать или скрывать поверхности, настроить точки и перейти в полноэкранный режим.

Архитектура основана на технологии Electron, и основные и рендеринговые процессы работают вместе с помощью межпроцессной связи (IPC). Защита обеспечивается изолированным контекстом и проверкой данных, чтобы избежать уязвимостей.

Electron Forge, включенный в программу, облегчает её распространение на Windows, macOS и Linux. Вы можете сотрудничать и обмениваться исходным кодом и упакованным приложением на GitHub. Документация приложения содержит полезные руководства для пользователей.

### 4.2 Тестирование на реальные данные

В рамках оценки правильной работы программы и проверка её производительности была протестирована программа на различные наборы данных (наборы данных). Эти наборы данных охватывают такие темы, как землетрясения в Японии и Греции, мировые температурные рекорды, температуры в Индии и население городов США по данным переписи населения 2021 года. Вот подробное описание различных используемых наборов данных, проведённых тестов и полученных результатов.

#### 4.2.1 Землетрясения в Японии

Первый набор данных о землетрясениях в Японии [17]. Этот набор данных содержит подробную информацию о крупных землетрясениях, произошедших в Японии. Данные включают в себя такую информацию, как величина, местоположение, глубина и дата каждого сейсмического события.

Полученные результаты (рис. 12) показывают, что приложение может интерпретировать и отображать географические данные, связанные с землетрясениями, например, визуализируя местоположения на карте.

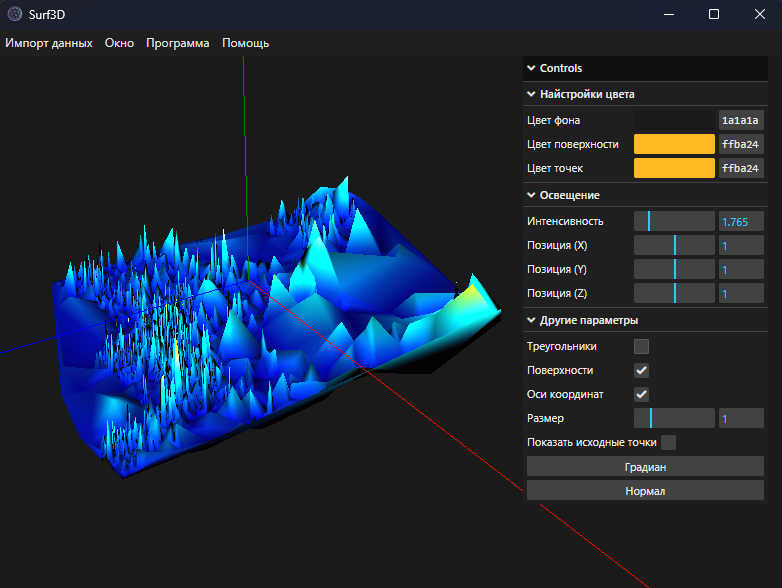


Рисунок 12 – Пример визуализации данных о землетрясении в Японии

### 4.2.2 Землетрясения в Греции

Другой используемый набор данных касается землетрясений в Греции [18]. Эти данные содержат аналогичные данные, в том числе о магнитуде, местоположении, глубине и дате землетрясений, произошедших в Греции.

Тесты, проведённые с этим набором данных, были направлены на проверку гибкости программы для обработки данных из разных регионов. Результаты (рис. 13) показали, что приложение также может точно визуализировать эти землетрясения, подтверждая способность программы обрабатывать разнообразные географические данные.

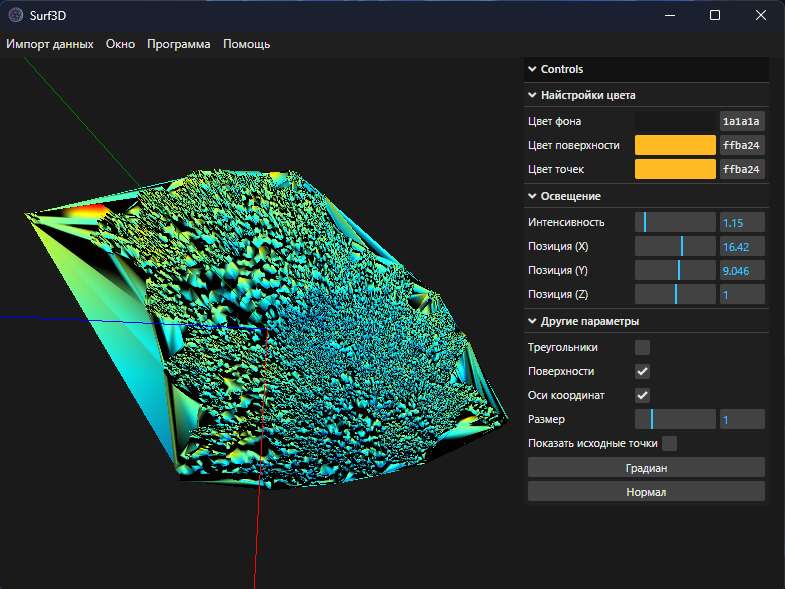


Рисунок 13 – Пример визуализации данных о землетрясения в Греции

#### 4.2.3 Температура в Индии

Набор данных основан на измерениях температуры в различных регионах Индии [19]. Набор данных содержит такие измерения, как средняя, минимальная и максимальная температура, а также некоторую информацию об осадках. Однако в нашем случае мы ограничимся только температурой. Цель этого теста - изучить поведение программы на большом количестве точек и сравнить полученную поверхность с реальными поверхностями. Глядя на результаты восстановления поверхности (рис. 14), видно, что модель, полученная путём реконструкции поверхности, похожа на реальную поверхность Индии. Из этого можно сделать вывод о высоких значениях температуры (красный цвет), а также о низких температурах (синий цвет).

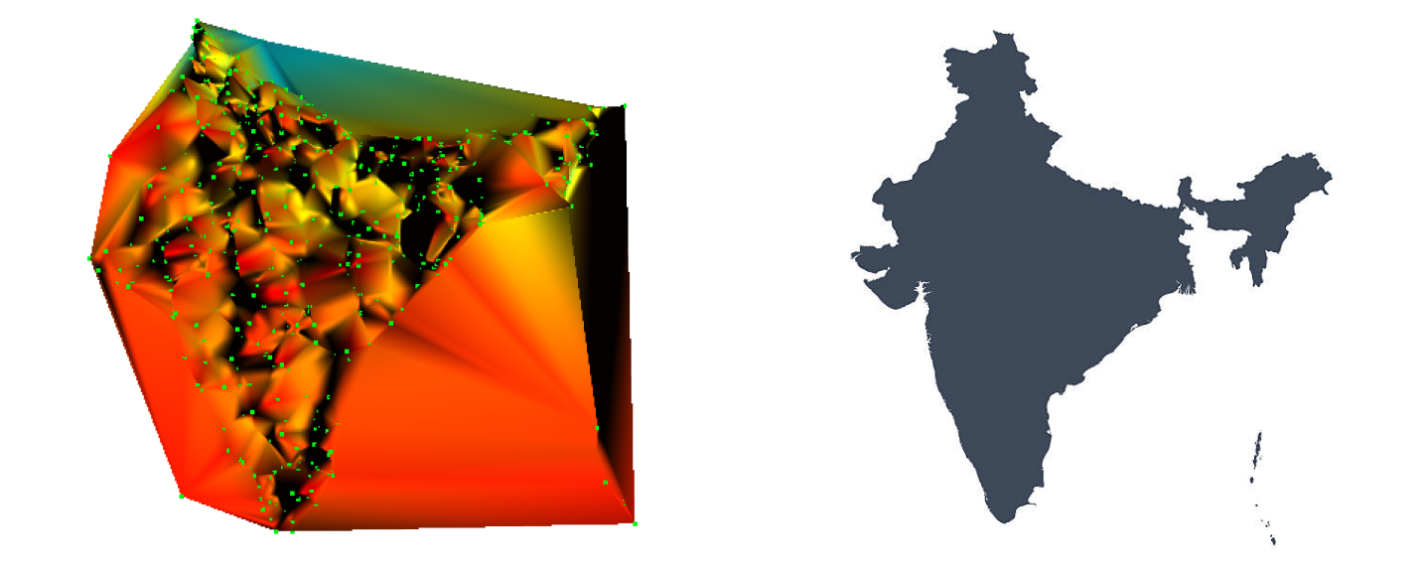


Рисунок 14 – Полученная модель о температуре в Индии и настоящая модель

#### 4.2.4 Население городов США

Последний пакет данных - это набор данных, содержащий данные о населении США по данным переписи 2021 года [20]. Этот набор данных содержит географические положения (широта и долгота) точек демографических измерений , а также население в этих точках. После тестирования программы на этом наборе (рис. 15) данных можно заметить, что по трём полученным точкам поверхности наблюдается значительная высота: они совпадают с самыми густонаселёнными городами Соединённых Штатов, а именно с Нью-Йорком, Лос-Анджелесом и Чикаго (рис. 16). Можно сделать вывод, что программа работает правильно.

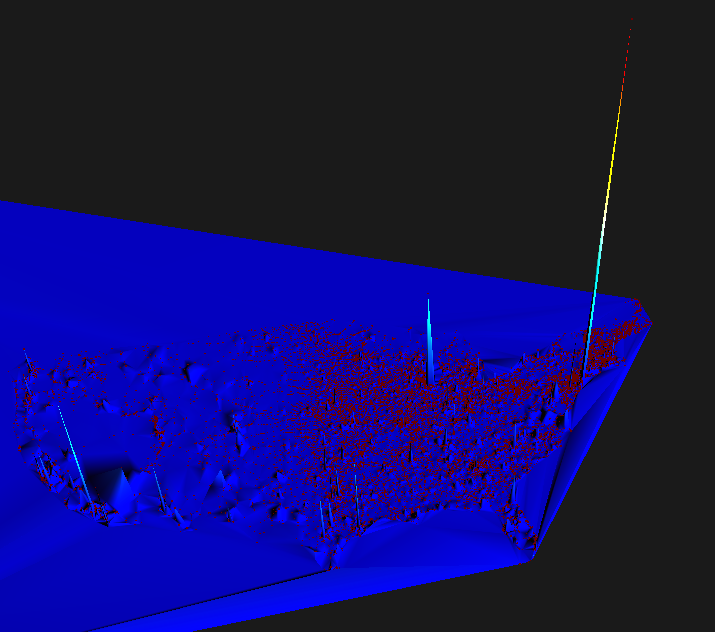


Рисунок 15 – Пример визуализации данных о населении в Греции

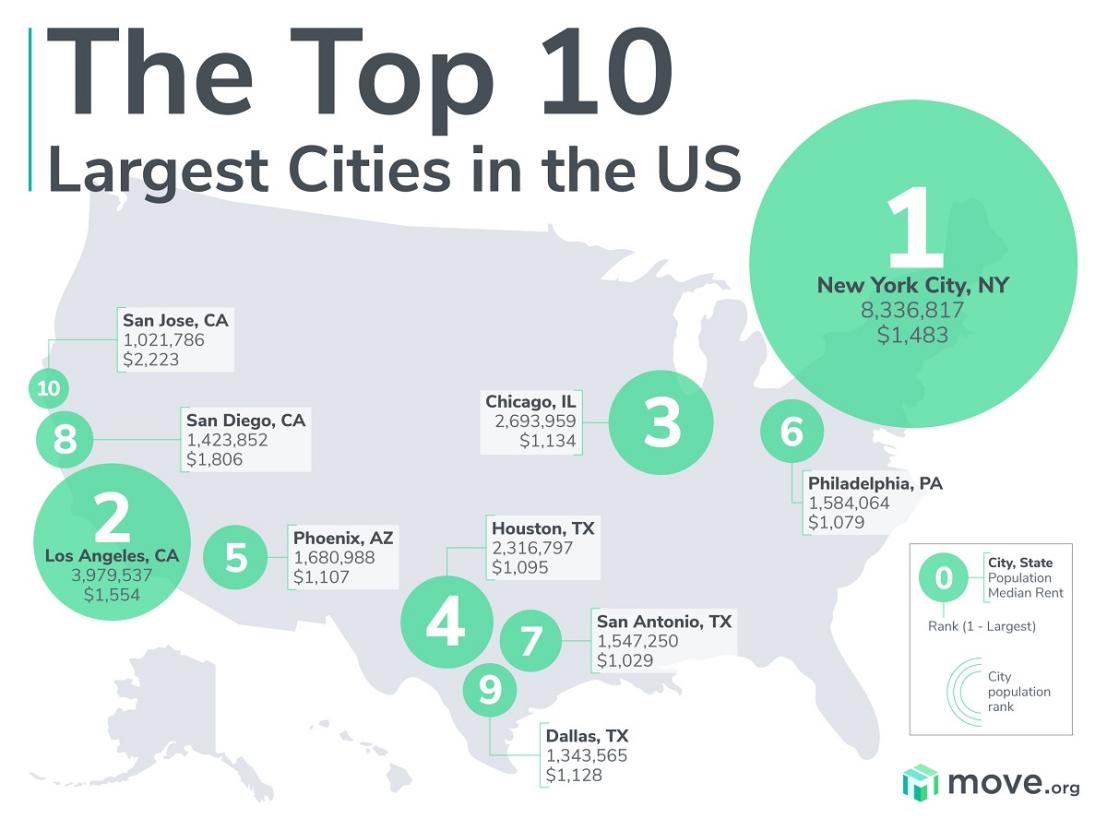


Рисунок 16 – Карта с информацией о самых густонаселённых городов США

**4.2.5 Вывод**

Тесты, проведённые на различных наборах данных, показали эффективность и точность программы. Программа успешно восстановила данные о температурах, магнитудах землетрясений и населении, поэтому она может работать с реальными данными.

### 4.3 Стратегии для совершенствования разработанного инструмента

Чтобы сделать программу более эффективной и привлечь больше пользователей, возможно и необходимо пересмотреть некоторые аспекты программы и добавить другие функции, чтобы сделать её более конкурентоспособной по сравнению с аналогами.

#### 4.3.1 Поддержка квадратичных сеток

Программа использует триангуляцию Делоне для генерации поверхностей. Несмотря на быстрое выполнение, этот тип сетей ограничен с точки зрения точности. Для этого было бы довольно интересно предусмотреть возможность выбора другого типа сети. Как, например, квадратичные сетки. Они более точные, чем треугольные, но гораздо более ресурсоёмкие. Эта функция привлечёт больше пользователей, у которых есть проекты, требующие такой точности.

#### 4.3.2 Другие типы интерполяция в треугольниках

В настоящее время программа для выполнения реконструкции поверхностей использует линейную интерполяцию, чтобы иметь возможность генерировать значения точек внутри каждого треугольника. Недостаток заключается в том, что полученные поверхности могут иметь слишком острые углы, когда существует большая разница между значениями, используемыми для построения поверхностей. Одним из способов исправить это было бы использование других типов интерполяции, таких как, например, кубическая интерполяция или сплайны. Они, безусловно, более ресурсоёмкие, но гарантируют лучший рендеринг.

#### 4.3.3 Веб-хостинг приложения

Поскольку программа была создана с использованием веб-технологий, а именно Javascript, HTML и CSS, её можно развернуть не как настольную программу, а как онлайн-программу, как, например, Figma. Таким образом, пользователь сможет просматривать свою виртуальную рабочую область и ее последние версии в режиме онлайн только с помощью браузера без установки дополнительных элементов. Это также упростит обновление программы.

#### 4.3.6 Вывод

Чтобы улучшить и масштабировать своё приложение, необходимо сосредоточиться на стратегиях, повышающих гибкость, совместимость, доступность и инновации. Добавив поддержку квадратичных сеток, улучшив интерполяцию в треугольниках, разместив приложение в интернете, внедрив функции экспорта в популярные форматы и внедрив новые технологии, инструмент станет более надёжным и универсальным. Эти стратегии помогут расширить аудиторию приложения, повысить его привлекательность для различных типов пользователей и обеспечить его адаптацию к будущим изменениям в технологиях и потребностях пользователей.

## 5 Обеспечение качества разработки программного продукта

### 5.1 Понятие качества

В различных областях человеческой деятельности слово «качество» используется по-разному. Этот термин может быть использован для описания различных видов продукции, таких как качественная одежда, услуги, такие как качественное образование, процессы, такие как выполнение проекта, и даже более абстрактных вещей, таких как качество жизни.

Тем не менее, каждый, кто использует этот термин, обнаруживает, что качество объекта всегда субъективно. Что-то, что является качественным для одного человека, может быть некачественным для другого, а третий человек может вообще не оценивать объект, потому что он не планирует его использовать.

Есть ещё одно важное свойство качества. Качество объекта, как правило, не ограничивается его отдельными характеристиками, а связано с объектом как целым, охватывающим его полностью и неотделимо от него. В результате понятие «качество» связано с существованием предмета, объекта или явления.

Оцениваемый объект не может быть назван качественным, если он является надёжным, но не выполняет своё функциональное предназначение. И это утверждение верно, и наоборот.

Повседневные понятия качества в человеческой практике неполны, многообразны и неточны из-за бесконечного разнообразия явлений и объектов окружающей действительности, но в каждом случае они отвечают определенным потребностям общения.

Несмотря на значительные трудности, определение качества объектов является существенной необходимостью при проектировании и производстве продукции, услуг, процессов и т.п. Это достигается между застройщиком и производителем без общего понимания качества объекта. Невозможно достичь потребительского удовлетворения или экономического успеха в развитии.

В 1986 году Международной организацией по стандартизации (ISO) были утверждены ключевые термины в сфере качества, опубликованные в специальных стандартах. Согласно стандарту ISO 9000 «Системы менеджмента качества. Основные положения и словарь», термин «качество» определён как «степень соответствия совокупности присущих характеристик требованиям». Однако, данное базовое определение оказалось недостаточным, и было необходимо стандартизировать ключевые термины, включённые в него.

Термин « **требование** » определяется как потребность или ожидание, которое установлено, обычно предполагается или является обязательным. В дополнение к стандартизации ключевых слов, в стандарте сопровождаются разъяснениями и комментариями. Например, термин « качество » может быть оценено прилагательными « плохое », « хорошее » или « отличное ».

Понятие "собственный, присущий", в отличие от "назначенного, присвоенного", означает наличие неизменной характеристики. Исключить комментарии по важным терминам, таким как "требование" и "характеристика", невозможно. Например, по термину "требование" отмечается, что фраза "обычно предполагается" означает, что это распространённая практика для организации, её клиентов и других заинтересованных сторон, что соответствующая потребность или ожидание подразумеваются или ожидаются.

Для изменения восприятия потребителей по отношению к продукции или услугам, организация может модифицировать характеристики своей продукции и услуг. Поэтому в словаре уделено значительное внимание определениям терминов «характеристика» и «характеристика качества», а также сопровождающим их примечаниям. В частности, заметно расширенное примечание к термину «характеристика»:

* ПРИМЕЧАНИЕ 1. Характеристика может быть присущей (собственной) или назначенной (присвоенной).
* ПРИМЕЧАНИЕ 2. Характеристика может быть качественной (описывающей качество) или количественной (описывающей количество).
* ПРИМЕЧАНИЕ 3. Существуют различные классы характеристик, включая:
* Физические (механические, электрические, химические или биологические характеристики);
* Органолептические (связанные с запахом, вкусом, осязанием, зрением или слухом);
* Этические (вежливость, честность, правдивость);
* Временные (пунктуальность, надёжность, доступность);
* Эргономические (физиологические характеристики или характеристики, связанные с безопасностью человека);
* Функциональные (например, максимальная скорость самолёта).

Среди бесчисленного множества характеристик выделяются характеристики качества, которые отличаются тем, что вытекают из требований. Как определено:

« **Характеристика качества** - это собственная (присущая) характеристика продукции, процесса или системы, относящаяся к требованию ». Иными словами, характеристика качества формируется с учётом потребностей потребителей.

В стандарте также содержится определение термина «потребитель»:

« **Потребитель** - это организация или лицо, которые получают продукцию. Примеры включают клиента, заказчика, конечного пользователя, розничного торговца, бенефициара и покупателя ». В примечании к этому термину отмечается, что потребитель может быть как внутренним, так и внешним по отношению к организации.

Для более наглядного представления термина «соответствие» можно визуализировать графически одну из интерпретаций термина качества. Единство трёх элементов термина качества - характеристики, требования и соответствия - отражает сущность взаимодействия между поставщиком и потребителем (рис. 17).

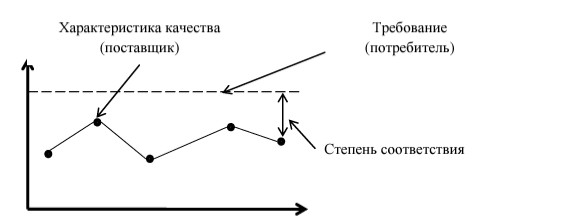


Рисунок 17 – Графическая интерпретация связи элементов термина качество

**Под термином « планирование качества** » **стандарт понимает часть управления качеством**, которая нацелена на установление целей в области качества и определение необходимых операционных процессов в жизненном цикле продукции, а также соответствующих ресурсов для достижения этих целей.

Важно отметить, что понятия «качество», «характеристика качества» и «планирование качества» всегда связаны с объектом, который может представлять собой продукцию, услугу, процесс и так далее. Стандарт ИСО 9000 предоставляет определения для этих терминов.

Под термином « **продукция** » понимается результат процесса. Стандарт определяет четыре общие категории продукции:

* Услуги (например, перевозки).
* Программные средства (например, компьютерная программа, словарь).
* Технические средства (например, узел двигателя).
* Перерабатываемые материалы (например, смазка).

Многие виды продукции могут включать элементы, относящиеся к различным общим категориям продукции. Классификация продукции как услуг, программных или технических средств или перерабатываемых материалов зависит от доминирующего элемента.

Например, продукция «автомобиль» включает в себя технические средства (например, шины), перерабатываемые материалы (горючее, охлаждающая жидкость), программные средства (программное управление двигателем, инструкция для водителя) и услуги (объяснения по эксплуатации, предоставляемые продавцом).

Услуга представляет собой результат, как минимум, одного действия, обязательно осуществлённого в ходе взаимодействия поставщика и потребителя, и, как правило, она нематериальна. Предоставление услуги может включать в себя следующее:

* Деятельность, проводимую над материальной продукцией, предоставленной потребителем (например, ремонт автомобиля);
* Деятельность, проводимую над нематериальной продукцией, предоставленной потребителем (например, заполнение налоговой декларации для расчёта налога);
* Предоставление нематериальной продукции (например, передача информации или знаний);
* Создание благоприятных условий для потребителей (например, в гостиницах и ресторанах);
* Программное средство содержит информацию и обычно является нематериальным, но также может принимать форму подходов, операций или процедур.

### 5.2 Операциональные определения характеристик качества

Важной задачей для разработчика продукции или услуги является преобразование требований потребителя в характеристики качества, а для производителя - обеспечение необходимой степени соответствия этих характеристик требованиям. Например, если заказчик программного продукта выдвигает требование к его быстродействию, разработчик преобразует это требование в характеристику "время отклика", устанавливает с потребителем целевое значение для этой характеристики и разрабатывает процедуру измерения, которую использует тестировщик программного продукта. Таким образом, можно быть уверенным, что заказчик, разработчик и тестировщик понимают данное требование одинаково.

Однако существует множество требований, для которых гораздо сложнее определить характеристики качества, таких как удобство, вежливость, честность и т. д. Некоторые требования могут быть выражены несколькими характеристиками. Например, удобство при ношении обуви зависит как от материала, так и от пошива и конструкции ботинка.

Доктор Эдвард Деминг, работавший в Японии после Второй мировой войны и сыгравший значительную роль в успехе японской промышленности, ввёл в практику менеджмента качества понятие операциональности определений. Он считал работы по созданию операциональных определений крайне важными, поскольку без них практически невозможно повысить удовлетворённость потребителей. Операциональность определений играет важную роль при сборе данных о качестве, особенно когда неоднозначные характеристики, такие как своевременность, стабильность и однородность, требуется измерить.

Операциональные определения (ОО) необходимы, когда требуется ответить на вопросы о том, является ли оцениваемый объект «удобным», «хорошим» или «дефектным».

Операциональное определение представляет собой конкретное уточнение значения термина в контексте конкретной системы, учитывая специфические условия и участников этой системы. Оно должно оставаться согласованным для поставщика и потребителя в течение времени. Следовательно, хорошо сформулированное ОО обеспечивает однозначное понимание термина всем участникам процесса в любое время.

Проверить работоспособность ОО можно только через его применение на практике. Если ОО помогает достичь единого понимания терминов, связанных с качеством, то оно эффективно. Если же нет - то нет. Например, понятие "чистоты" может иметь разные значения в разных ситуациях. В мастерской "чистый" означает отсутствие беспорядка, в то время как на операционном столе "чистый" может означать обработанный антисептиком для предотвращения инфекции. Операциональное определение термина «чистый» будет различаться в каждой из этих ситуаций.

При операционализации определения важен контекст, то есть условия применения термина. Э. Деминг отмечал существенные трудности при формировании операциональных определений: 'Каждый считает, что знает, что такое загрязнение, до тех пор, пока не попытается объяснить это кому-то ещё'. В своих работах Деминг выделил три элемента, которые необходимы для формирования операционального определения:

* Тест: метод испытания или процедура измерения свойства объекта.
* Критерий: стандарт, относительно которого оценивается результат теста.
* Решение: процедура принятия решения о том, соответствует ли результат теста критерию.

### 5.3 Методы выявления требований потребителей

Для выявления требований потребителя могут использоваться следующие методы:

Опрос, анкетирование – эти методы позволяют быстро и оперативно анализировать состояние и тенденции отношения к продукту больших групп людей. К достоинствам можно отнести сравнительную экономичность, оперативность и организационную простоту. Однако к недостаткам относится зависимость качества получаемой информации от аккуратности и внимательности респондентов, а также от их восприятия текста анкеты.

Анализ отзывов пользователей – этот метод позволяет определить факторы, приводящие к позитивным и негативным сценариям использования продукта. Достоинства данного метода включают получение информации без необходимости вложения средств, так как все отзывы находятся в открытом доступе. Однако недостатком является то, что пользователи чаще пишут негативные отзывы, что затрудняет получение полной картины пользовательского опыта.

Варианты использования (use case) представляют собой модель, описывающую взаимодействие внешних лиц с разрабатываемой системой. Они удобны для описания функциональных требований, однако не всегда подходят для систем, критичных к безопасности или реального времени, где требуется высокая точность.

Прототипирование включает создание программного прототипа, который демонстрирует часть функциональности будущей системы. Этот метод позволяет пользователям выявить недочёты и предпочтения, которые могут быть оформлены в виде требований к ПО. Горизонтальный прототип демонстрирует внешний вид системы, а вертикальный - функциональность и интерфейс. Прототипирование помогает сократить время и затраты, улучшает качество требований и может сэкономить деньги на дальнейших доработках, хотя требует дополнительных затрат на его разработку.

Фокус-группа собирается для решения конкретных проблем, часто связанных с взаимодействием подсистем программного продукта. Она позволяет оперативно сформулировать требования к системе, однако может привести к упущению пользовательских требований, так как требования определяются не потребителями.

Наиболее подходящим из предложенных методов является прототипирование. Этот способ позволяет получить более точные пользовательские требования к системе, поскольку пользователь может взаимодействовать с прототипом. Анкетирование и опрос менее эффективны, так как респондентам нужно представлять, как будет функционировать система. Анализ отзывов не подходит, поскольку программное обеспечение является частью продукта компании и может быть недоступно для общественного доступа. Варианты использования не учитывают нефункциональные требования, такие как скорость работы. Фокус-группа, состоящая из членов команды, может пропустить важные пользовательские требования.

### 5.4 Расчёт быстродействия программного обеспечения

Для обеспечения стабильной работы программных средств разработано

ОО “Быстродействие”, приведённое в таблице 2.

Таблица 2 – Разработка ОО «Быстродействие»

|  |  |
| --- | --- |
| Элементы ОО | Комментарий |
| Критерий: Набор программных средств работает быстро и беззаметных задержек. | Стандарт заключается в измерении  метрики производительности FPS.  Минимальное количество FPS не должно быть ниже среднего показателя (30 FPS). |
| Тест:  Измерение FPS при  использовании программных  средств | измерение производится с помощью  инструмента Chrome DevTools при  открытых окнах из системы окон, а также при работе с редактором фигур. |
| Решение: Если показатель не соответствует критерию, учётная система не соответствует ОО “Быстродействие”. | Для принятия решения необходимо  посчитать минимальное значение всех полученных показателей. |

Измерены значения FPS при использовании визуализатора поверхности и системы окон. Самые низкие значения FPS наблюдены составляли примерно 40 кадров в секунд. (рис. 18). Среднее значение FPS составило около 60 по итогам измерения. Наименьшие показатели достигались в ходе построения поверхности, так как основная нагрузка приходится на отображение 3D – графики.

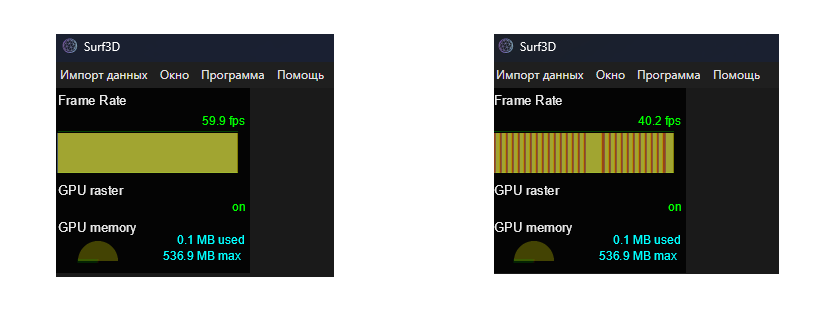


Рисунок 18 – Значения FPS в процессе измерения

**Вывод о качестве разработки.**

Проведя расчёт среднего значения показателя FPS в процессе работы Surf3D, можно заключить, что она достаточно оптимизирована и способна эффективно функционировать.

## ЗАКЛЮЧЕНИЕ

В ходе выполнения выпускной квалификационной работы была достигнута поставленная цель – была разработана программа для реконструкции поверхности по нерегулярным наборам точек с использованием треугольной сетки.

Для достижения цели были выполнены следующие задачи:

1. Поиск и обзор существующих решений;
2. Формирование требований к программному обеспечению;
3. Реализация программного обеспечения;
4. Тестирование программы на реальные данные.

Программа для реконструкции поверхности по нерегулярным наборам точек с использованием треугольной сетки была реализована на языке программирования Javacsript с использованием специализированных библиотек такие как Three.js, Electron.js и Delaunator.js. Разработанное программное обеспечение позволяет загружать нерегулярные наборы точек из файлов и автоматически строить на их основе треугольную сетку и поверхность.

Для удобства пользователя был разработан графический интерфейс, соответствующий современным стандартам пользовательского опыта. Интерфейс позволяет загружать наборы точек, управлять параметрами построения треугольной сетки и визуализировать результаты реконструкции поверхности. Дополнительно были реализованы механизмы проверки входных данных на корректность, что позволяет предотвратить ошибки при обработке наборов точек и построении треугольной сетки.

В результате получилось функциональное решение, которое позволяет производить реконструкцию поверхности по нерегулярным наборам точек с использованием треугольной сетки. Однако, есть потенциал для дальнейшего совершенствования и расширения функционала приложения.

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

.

.\.\.

.

.

.

.

.

.

.

.

## ПРИЛОЖЕНИЕ А

Исходный код реализации расположен в GitHub репозитории:

https://github.com/Christian03031/Surf3D