МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕУЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

по выполнению лабораторных работ

по дисциплине

«ПРОГРАММИРОВАНИЕ КОМПЬЮТЕРНОЙ ГРАФИКИ»

Направление подготовки 09.03.02 Информационные системы и технологии
Направленность (профиль) Прикладное программирование и интернет-технологии

Ставрополь
2026

СОДЕРЖАНИЕ

ЛАБОРАТОРНАЯ РАБОТА №1. АРХИТЕКТУРА GPU И РАБОТА С SHADERTOY

Цели и задачи

Цель лабораторной работы: освоение интерфейса shadertoy, изучение принципов работы фрагментных шейдеров и математических основ нормализации экранных координат.

Основные задачи:

– изучить назначение входных переменных: iresolution, itime, imouse;

– освоить преобразование пиксельных координат fragcoord в нормализованное пространство [0, 1];

– научиться формировать итоговый цвет пикселя через вектор vec4 fragcolor.

Формируемые компетенции: ПК-1, ПК-2, ПК-4

Теоретическое обоснование

Перед выполнением лабораторной работы необходимо ознакомиться с лекционным материалом.

Оборудование и материалы

Для выполнения лабораторной работы рекомендуется использовать персональный компьютер со следующими программными средствами разработки (выбрать один или несколько программных продуктов для практической реализации задач лабораторной работы): google chrome / mozilla

firefox, visual studio code, clion / microsoft visual studio 2022.

Методика и порядок выполнения работы

Перед выполнением заданий рекомендуется ознакомиться с решением учебной задачи:

https://github.com/vberezina/PCG/blob/main/LabWork1.md

Порядок выполнения работы:

1. Авторизуйтесь на сайте shadertoy.com и создайте новый проект через кнопку 'new'. Работа ведется во вкладке 'image'.

2. Код шейдера компилируется нажатием кнопки 'play' или комбинацией клавиш alt+enter.

3. В случае ошибок компиляции изучите лог в нижней части редактора.

4. Каждое задание рекомендуется сохранять как отдельную итерацию или комментировать предыдущие этапы для сохранения структуры кода.

Задание 1. Нормализация и градиенты.

Создание базового визуального вывода путем преобразования дискретных координат сетки пикселей в непрерывные значения.

– Вычислить нормализованные координаты uv, учитывая соотношение сторон (aspect ratio), чтобы избежать искажений.

– Реализовать горизонтальный градиент, плавно переходящий от синего цвета к золотому.

– Добавить вертикальную составляющую градиента, чтобы цвет менялся также от нижнего края к верхнему.

Задание 2. Интерактивность и логика.

Использование встроенных функций для создания процедурных масок и взаимодействия с пользователем.

– Используя функцию distance(), реализовать отрисовку круга в центре экрана.

– Привязать положение центра круга к координатам курсора мыши (imouse.xy).

– Сделать границы круга мягкими, применив функцию smoothstep() вместо прямого сравнения.

Задание 3. Динамика и тригонометрия.

Работа с переменной времени для создания процедурной анимации.

– Реализовать циклическое изменение интенсивности одного из каналов цвета с помощью функции sin(itime).

– Создать эффект 'пульсации' круга из предыдущего задания, изменяя его радиус во времени.

– Объединить синусоидальное движение с изменением координат, чтобы объект перемещался по траектории 'восьмерки' (фигура лиссажу).

Содержание отчета и его форма

Отчет по лабораторной работе должен содержать:

1. Номер и название лабораторной работы; задачи лабораторной работы.

2. Реализация каждого задания с приведением исходного кода программы, диаграмм и графиков для визуализации данных.

3. Ответы на контрольные вопросы.

Отчет о выполнении лабораторной работы подписывается студентом и сдается преподавателю.

Контрольные вопросы

1. Что такое фрагментный шейдер и для какой части графического конвейера он предназначен?

2. В чем заключается физический смысл входного вектора fragcoord?

3. Почему для нормализации координат необходимо делить вектор на iresolution.xy?

4. Какую структуру имеет тип данных vec4 в языке glsl?

5. Как получить доступ к текущему времени работы шейдера в секундах?

6. Опишите математический смысл функции smoothstep(edge0, edge1, x)

7. Почему в шейдерах предпочтительнее использовать встроенные функции вместо условных операторов if/else?

8. В каких единицах измерения хранятся значения цвета в каналах r, g, b, a?

9. Как вычислить соотношение сторон экрана (aspect ratio) и зачем это нужно при отрисовке фигур?

10. За что отвечают компоненты x и y вектора imouse при нажатой и отжатой кнопке мыши?

11. Что такое 'нормализованное пространство' и почему оно удобно для программирования графики?

12. Как инвертировать цвет пикселя, зная его текущее значение в vec3?

ЛАБОРАТОРНАЯ РАБОТА №2. МАТЕМАТИКА 2D-ПРИМИТИВОВ

Цели и задачи

Цель лабораторной работы: изучение математического описания геометрических примитивов и способов их комбинирования через функции знакового расстояния.

Основные задачи:

– изучить аналитическое описание базовых фигур: прямоугольника, круга и сегмента линии;

– освоить операции объединения, вычитания и пересечения множеств в контексте функций дистанции;

– научиться визуализировать границы фигур и их внутреннюю область с использованием сглаживания.

Формируемые компетенции: ПК-1, ПК-2, ПК-4

Теоретическое обоснование

Перед выполнением лабораторной работы необходимо ознакомиться с лекционным материалом.

Оборудование и материалы

Для выполнения лабораторной работы рекомендуется использовать персональный компьютер со следующими программными средствами разработки (выбрать один или несколько программных продуктов для практической реализации задач лабораторной работы): google chrome / mozilla firefox, visual studio code, clion / microsoft visual studio 2022.

Методика и порядок выполнения работы

Перед выполнением заданий рекомендуется ознакомиться с решением учебной задачи:

https://github.com/vberezina/PCG/blob/main/LabWork2.md

Порядок выполнения работы:

1. При выполнении работы необходимо реализовать функции sdf как отдельные блоки кода перед основной функцией mainimage.

2. Для отладки используйте визуализацию знака дистанции: например, окрашивайте область с отрицательным значением в один цвет, а с положительным — в другой.

3. Комбинируйте примитивы, изменяя входной вектор координат p для каждой фигуры индивидуально (локальные координаты), чтобы управлять их положением независимо.

Задание 1. Создание библиотеки примитивов.

Реализация функций, вычисляющих кратчайшее расстояние от точки до границы геометрического объекта.

– Написать функцию sdbox(p, b), возвращающую расстояние до прямоугольника с заданными размерами.

– Реализовать функцию отрисовки сегмента линии заданной толщины.

– Вывести на экран композицию, где положение одной из фигур зависит от времени (itime).

Задание 2. Логические операции и морфинг.

Использование математических операторов для построения сложных форм и плавных переходов между ними.

– Реализовать вычитание круга из центра квадрата для создания полой формы.

– Использовать функцию smin (smooth minimum) для создания эффекта плавного слияния двух метасфер.

– Реализовать морфинг (плавную трансформацию) между кругом и квадратом, используя линейную интерполяцию (mix) их функций дистанции.

Задание 3. Стилизация границ и свечение.

Применение математических преобразований к значению дистанции для создания визуальных эффектов контура и объема.

– Создать эффект контура (stroke) заданной толщины, используя функцию abs(d) от значения дистанции.

– Реализовать эффект внешнего свечения, интенсивность которого падает по экспоненте при удалении от границы фигуры.

– Создать эффект повторяющихся колец вокруг фигуры, используя функцию sin() от значения sdf.

Содержание отчета и его форма

Отчет по лабораторной работе должен содержать:

1. Номер и название лабораторной работы; задачи лабораторной работы.

2. Реализация каждого задания с приведением исходного кода программы, диаграмм и графиков для визуализации данных.

3. Ответы на контрольные вопросы.

Отчет о выполнении лабораторной работы подписывается студентом и сдается преподавателю.

Контрольные вопросы

1. Что такое функция знакового расстояния (sdf) и в чем её преимущество перед растровыми масками?

2. Какой знак (положительный или отрицательный) обычно возвращает функция sdf для точек внутри фигуры?

3. Почему для объединения двух объектов в пространстве sdf используется функция min()?

4. Как математически записывается операция пересечения двух объектов а и б через их функции дистанции?

5. Опишите алгоритм получения эффекта скругления углов любой жесткой фигуры через манипуляцию со значением её sdf

6. Каким образом можно инвертировать фигуру (сделать 'дырку' во всем пространстве)?

7. В чем заключается разница между обычным булевым объединением и функцией плавного минимума (smin)?

8. Как реализовать бесконечное повторение одной фигуры по сетке, используя функцию fract()?

9. Почему при масштабировании координат p перед вычислением sdf необходимо корректировать итоговое расстояние?

10. Как получить вектор нормали к границе 2d-фигуры, зная её функцию дистанции?

11. Как реализовать зеркальное отражение объекта относительно оси координат внутри шейдера?

12. Что такое 'артефакты SDF' и почему функция дистанции должна возвращать именно кратчайшее расстояние до поверхности?

ЛАБОРАТОРНАЯ РАБОТА №3. ПРОСТРАНСТВЕННЫЕ ПРЕОБРАЗОВАНИЯ

Цели и задачи

Цель лабораторной работы: освоение методов манипуляции экранными координатами для создания повторяющихся и динамически изменяющихся структур.

Основные задачи:

– изучить применение матриц аффинных преобразований в фрагментном шейдере;

– освоить работу с полярной системой координат для создания радиальных узоров;

– научиться использовать периодические функции для бесконечного тиражирования объектов.

Формируемые компетенции: ПК-1, ПК-2, ПК-4

Теоретическое обоснование

Перед выполнением лабораторной работы необходимо ознакомиться с лекционным материалом.

Оборудование и материалы

Для выполнения лабораторной работы рекомендуется использовать персональный компьютер со следующими программными средствами разработки (выбрать один или несколько программных продуктов для практической реализации задач лабораторной работы): google chrome / mozilla firefox, visual studio code, clion / microsoft visual studio 2022.

Методика и порядок выполнения работы

Перед выполнением заданий рекомендуется ознакомиться с решением учебной задачи:

https://github.com/vberezina/PCG/blob/main/LabWork3.md

Порядок выполнения работы:

1. При выполнении работы следует разделять глобальные координаты экрана и локальные координаты объектов.

2. Для создания сетки используйте функцию fract(), которая делит пространство на ячейки [0, 1].

3. При вращении и масштабировании помните, что операции применяются к системе координат, а не к самой фигуре, поэтому порядок умножения матриц или последовательность функций (перенос -> поворот) критически важны для итогового результата.

Задание 1. Аффинные трансформации и вращение.

Управление положением и ориентацией объектов в кадре через изменение векторов координат.

– Реализовать функцию rotate2d(p, angle), возвращающую повернутый вектор координат.

– Создать анимацию, в которой квадрат вращается вокруг своего центра, смещенного относительно начала координат.

– Реализовать эффект 'масштабирования из центра' (zoom), зависящий от времени itime.

Задание 2. Бесконечный тайлинг и сетки.

Создание паттернов и структур, заполняющих все экранное

пространство без использования циклов for.

– Разделить экран на сетку ячеек с помощью функции fract(p * n), где n — количество повторений.

– Внутри каждой ячейки отрисовать круг, радиус которого плавно меняется в зависимости от глобальных координат x и y.

– Реализовать шахматный порядок (checkerboard), используя функцию floor() и оператор остатка от деления или функцию mod().

Задание 3. Полярные координаты и калейдоскоп.

Переход в альтернативные системы координат для генерации сложных радиальных симметрий.

– Перевести декартовы координаты (x, y) в полярные (r, theta) с помощью функций length() и atan().

– Реализовать эффект 'сегментации' круга на n равных секторов (эффект калейдоскопа) через манипуляцию с углом theta.

– Создать динамический лепестковый узор, где радиус границы фигуры является функцией от угла: r = f(theta, itime).

Содержание отчета и его форма

Отчет по лабораторной работе должен содержать:

1. Номер и название лабораторной работы; задачи лабораторной работы.

2. Реализация каждого задания с приведением исходного кода программы, диаграмм и графиков для визуализации данных.

3. Ответы на контрольные вопросы.

Отчет о выполнении лабораторной работы подписывается студентом и сдается преподавателю.

Контрольные вопросы

1. Почему для перемещения объекта вправо на 5 единиц нужно вычитать 5 из координаты x, а не прибавлять?

2. Какова роль функции fract() в создании бесконечных паттернов?

3. Как вычислить индекс конкретной ячейки сетки, зная только текущие координаты фрагмента?

4. В чем преимущество полярных координат при создании радиально-симметричных узоров?

5. Какой диапазон значений возвращает стандартная функция atan(y, x) в glsl?

6. Как реализовать отражение пространства относительно произвольной оси?

7. Почему порядок операций 'сначала поворот, потом перенос' дает иной результат, чем 'сначала перенос, потом поворот'?

8. Как ограничить область действия fract(), чтобы тайлинг работал только в определенном квадрате экрана?

9. Что такое 'нормализованные полярные координаты' и как их привести к диапазону [0, 1]?

10. Как с помощью функции floor() реализовать дискретное движение объекта (скачками по сетке)?

11. Как избежать искажений при тайлинге на широкоформатных мониторах (учет aspect ratio)?

12. Как передать уникальное случайное значение (seed) в каждую ячейку сетки, созданной через fract()?

ЛАБОРАТОРНАЯ РАБОТА №4. ТЕОРИЯ ОСВЕЩЕНИЯ И ЦВЕТ

Цели и задачи

Цель лабораторной работы: изучение методов расчета нормалей и реализация классических моделей освещения для придания объема трехмерным сценам.

Основные задачи:

– реализовать алгоритм численного нахождения нормали к sdf-поверхности;

– освоить модель освещения блинна-фонга (ambient, diffuse, specular);

– научиться работать с позиционными источниками света в трехмерном пространстве.

Формируемые компетенции: ПК-1, ПК-2, ПК-4

Теоретическое обоснование

Перед выполнением лабораторной работы необходимо ознакомиться с лекционным материалом.

Оборудование и материалы

Для выполнения лабораторной работы рекомендуется использовать персональный компьютер со следующими программными средствами разработки (выбрать один или несколько программных продуктов для практической реализации задач лабораторной работы): google chrome / mozilla firefox, visual studio code, clion / microsoft visual studio 2022.

Методика и порядок выполнения работы

Перед выполнением заданий рекомендуется ознакомиться с решением учебной задачи:

https://github.com/vberezina/PCG/blob/main/LabWork4.md

Порядок выполнения работы:

1. На данном этапе необходимо дополнить движок raymarching функцией расчета нормалей.

2. Нормаль вычисляется как градиент функции sdf в точке пересечения луча с поверхностью.

3. Полученный вектор нормали используется совместно с вектором направления на источник света для расчета освещенности по модели блинна-фонга.

4. Рекомендуется разделять расчет диффузной, зеркальной и фоновой составляющих для гибкой настройки материала.

Задание 1. Вычисление нормалей.

Получение вектора, перпендикулярного поверхности, для корректного взаимодействия со светом.

– Написать функцию getnormal(p), использующую метод центральных разностей (смещение на небольшую величину epsilon по осям x, y, z).

– Визуализировать нормали объекта, используя их компоненты как значения r, g, b каналов цвета.

– Убедиться, что нормали корректно отображаются при вращении объекта в сцене.

Задание 2. Диффузное освещение по ламберту.

Создание базовой освещенности, зависящей от угла падения лучей света.

– Задать положение точечного источника света (lightpos) и вычислить вектор направления от поверхности к свету.

– Реализовать расчет диффузной составляющей через скалярное произведение dot(normal, lightdir) с ограничением clamp().

– Добавить фоновое освещение (ambient), чтобы затененные участки объекта не были абсолютно черными.

Задание 3. Блики и свойства материалов.

Добавление зеркальных отражений для имитации различных типов поверхностей.

– Реализовать зеркальную составляющую (specular) по модели блинна-фонга, используя вектор полупути (half-vector).

– Ввести параметр shininess (степень блеска) и продемонстрировать его влияние на размер светового пятна.

– Создать сцену, где один объект выглядит как матовый пластик, а другой — как полированный металл.

Содержание отчета и его форма

Отчет по лабораторной работе должен содержать:

1. Номер и название лабораторной работы; задачи лабораторной работы.

2. Реализация каждого задания с приведением исходного кода программы, диаграмм и графиков для визуализации данных.

3. Ответы на контрольные вопросы.

Отчет о выполнении лабораторной работы подписывается студентом и сдается преподавателю.

Контрольные вопросы

1. Почему нормаль к поверхности можно найти через градиент функции дистанции (sdf)?

2. Как размер шага (epsilon) при вычислении нормали влияет на качество освещения и детализацию?

3. В чем заключается физический смысл закона косинусов ламберта?

4. Почему в glsl при расчете освещения важно использовать функцию clamp(..., 0.0, 1.0)?

5. Какое преимущество дает модель блинна-фонга перед оригинальной моделью фонга?

6. Как изменится освещенность, если источник света находится бесконечно далеко (направленный свет)?

7. Что такое вектор отражения (reflect) и как он используется в расчетах?

8. Как реализовать затухание света в зависимости от расстояния до источника?

9. Почему при расчете освещения все векторы (нормаль, направление на свет) должны быть нормализованы?

10. Как комбинируются цвета объекта и цвета источника света (сложение или умножение)?

11. Как можно имитировать несколько источников света в одной функции рендеринга?

12. Что произойдет с освещением, если функция sdf возвращает неверное (не кратчайшее) расстояние?

ЛАБОРАТОРНАЯ РАБОТА №5. АЛГОРИТМ RAYMARCHING (ЧАСТЬ 1)

Цели и задачи

Цель лабораторной работы: реализация базового движка для визуализации трехмерных объектов методом прохода лучей (sphere tracing).

Основные задачи:

– изучить математическую модель перспективной камеры в 3d-пространстве;

– реализовать базовый цикл raymarching (sphere tracing) с использованием функций sdf;

– научиться визуализировать глубину сцены и форму трехмерных примитивов.

Формируемые компетенции: ПК-1, ПК-2, ПК-4

Теоретическое обоснование

Перед выполнением лабораторной работы необходимо ознакомиться с лекционным материалом.

Оборудование и материалы

Для выполнения лабораторной работы рекомендуется использовать персональный компьютер со следующими программными средствами разработки (выбрать один или несколько программных продуктов для практической реализации задач лабораторной работы): google chrome / mozilla firefox, visual studio code, clion / microsoft visual studio 2022.

Методика и порядок выполнения работы

Перед выполнением заданий рекомендуется ознакомиться с решением учебной задачи:

https://github.com/vberezina/PCG/blob/main/LabWork5.md

Порядок выполнения работы:

1. При выполнении работы необходимо перейти от обработки 2d-плоскости к симуляции трехмерного пространства.

2. Определите положение камеры (ro) и направление луча (rd) для каждого пикселя.

3. Основной цикл маршинга должен итеративно увеличивать дистанцию вдоль луча до тех пор, пока значение sdf не станет меньше заданного порога (столкновение) или не превысит максимальную дальность (промах).

4. Для визуализации на данном этапе используйте количество итераций или итоговую дистанцию до объекта.

Задание 1. Генерация лучей и виртуальная камера.

Настройка системы векторов для 'взгляда' внутрь трехмерной сцены.

– Вычислить вектор направления луча rd, учитывая положение пикселя и фокусное расстояние камеры.

– Реализовать простую систему вращения камеры вокруг центра координат с использованием функции sin(itime).

– Добавить коррекцию aspect ratio, чтобы трехмерные объекты не выглядели сплюснутыми.

Задание 2. Алгоритм прохода луча (sphere tracing).

Написание функции, которая пошагово приближается к поверхности

объекта.

– Реализовать функцию raymarch(ro, rd), возвращающую общее расстояние до ближайшей поверхности.

– Ограничить количество итераций цикла (например, до 100) для предотвращения зависаний gpu.

– Визуализировать 3d-сферу, используя полученную дистанцию для управления яркостью пикселя (эффект тумана).

Задание 3. Геометрия в 3d и бесконечный пол.

Создание пространственной композиции из нескольких объектов.

– Добавить в функцию сцены (map) расчет расстояния до бесконечной плоскости (y = -1.0).

– Реализовать 3d-куб, используя функцию sdbox, адаптированную под три измерения.

– Создать эффект 'дырявого' куба, вычтя из него три цилиндра по разным осям (операция csg в 3d).

Содержание отчета и его форма

Отчет по лабораторной работе должен содержать:

1. Номер и название лабораторной работы; задачи лабораторной работы.

2. Реализация каждого задания с приведением исходного кода программы, диаграмм и графиков для визуализации данных.

3. Ответы на контрольные вопросы.

Отчет о выполнении лабораторной работы подписывается студентом и сдается преподавателю.

Контрольные вопросы

1. В чем принципиальное отличие raymarching от классического метода растеризации полигонов?

2. Что такое 'ray origin' (ro) и 'ray direction' (rd)?

3. Почему алгоритм называется sphere tracing и как радиус сферы связан со значением sdf?

4. Какие параметры влияют на точность определения поверхности в этом алгоритме?

5. Что произойдет, если шаг луча будет больше, чем значение функции дистанции?

6. Как реализовать 'отсечение' объектов, находящихся слишком далеко от камеры (far plane)?

7. Зачем в начале функции mainimage мы нормализуем координаты пикселей в диапазон от -1 до 1?

8. Как математически представить камеру, смотрящую в произвольную точку пространства (lookat)?

9. Почему для отрисовки плоскости достаточно использовать простую операцию с координатой y?

10. Какова роль 'эпсилона' (минимального порога расстояния) в цикле маршинга?

11. Как визуально отличить артефакты нехватки итераций от ошибок в функции дистанции?

12. Можно ли с помощью raymarching отрисовать объекты, находящиеся внутри других объектов?

ЛАБОРАТОРНАЯ РАБОТА №6. АЛГОРИТМ RAYMARCHING (ЧАСТЬ 2)

Цели и задачи

Цель лабораторной работы: реализация алгоритмов визуализации теней и объединение множества объектов в единую когерентную сцену.

Основные задачи:

– освоить алгоритм проверки видимости источника света (shadow rays);

– реализовать расчет мягких теней (soft shadows) без дополнительных выборок;

– научиться работать с идентификаторами объектов (object id) для раздельной раскраски.

Формируемые компетенции: ПК-1, ПК-2, ПК-4

Теоретическое обоснование

Перед выполнением лабораторной работы необходимо ознакомиться с лекционным материалом.

Оборудование и материалы

Для выполнения лабораторной работы рекомендуется использовать персональный компьютер со следующими программными средствами разработки (выбрать один или несколько программных продуктов для практической реализации задач лабораторной работы): google chrome / mozilla firefox, visual studio code, clion / microsoft visual studio 2022.

Методика и порядок выполнения работы

Перед выполнением заданий рекомендуется ознакомиться с решением учебной задачи:

https://github.com/vberezina/PCG/blob/main/LabWork6.md

Порядок выполнения работы:

1. Для реализации теней в raymarching используется запуск дополнительного луча из точки на поверхности в сторону источника света.

2. Если на пути этого луча встречается объект (дистанция до поверхности становится меньше эпсилона), точка считается затененной.

3. Для получения мягких теней необходимо анализировать минимальное расстояние до объектов вдоль теневого луча и использовать его для вычисления коэффициента освещенности.

Задание 1. Жесткие тени (hard shadows).

Базовое определение освещенности с учетом геометрических препятствий.

– Написать функцию shadow(ro, rd), которая возвращает 0.0, если луч встретил препятствие, и 1.0 в противном случае.

– Интегрировать проверку тени в основной расчет освещения из предыдущей лабораторной работы.

– Добавить в сцену плоскость и несколько сфер, чтобы увидеть падающие тени.

Задание 2. Мягкие тени (soft shadows).

Создание реалистичных полутеней на основе анализа дистанции до объектов вблизи луча.

– Модифицировать теневой цикл, вычисляя коэффициент k = min(k,

penumbra * h / t), где h — дистанция до ближайшего объекта, t — пройденный путь.

– Реализовать параметр penumbra (размытость), позволяющий динамически менять мягкость теней.

– Сравнить производительность и визуальный результат жестких и мягких теней.

Задание 3. Многокомпонентная сцена и материалы.

Организация кода для работы с различными типами геометрии и их свойствами.

– Изменить функцию map() так, чтобы она возвращала vec2 (дистанция и id объекта).

– Реализовать процедурную текстуру 'шахматная доска' для плоскости пола, используя функцию floor() от мировых координат.

– Назначить разные цвета и параметры бликов (specular) для объектов с разными id.

Содержание отчета и его форма

Отчет по лабораторной работе должен содержать:

1. Номер и название лабораторной работы; задачи лабораторной работы.

2. Реализация каждого задания с приведением исходного кода программы, диаграмм и графиков для визуализации данных.

3. Ответы на контрольные вопросы.

Отчет о выполнении лабораторной работы подписывается студентом и сдается преподавателю.

Контрольные вопросы

1. Почему для теней луч выпускается из точки p + normal * epsilon, а не просто из p?

2. Как алгоритм raymarching позволяет вычислять мягкие тени практически 'бесплатно'?

3. Какую роль играет коэффициент k в формуле мягких теней?

4. Как избежать артефактов 'полос' (banding) при расчете теней?

5. Почему при расчете тени мы не идем до максимального расстояния, а ограничиваемся дистанцией до источника света?

6. Как реализовать самозатенение (self-shadowing) для сложных объектов?

7. Можно ли использовать raymarching для создания прозрачных теней (от стекла)?

8. Что такое 'ошибка точности' (precision error) при запуске теневого луча и как её минимизировать?

9. Как логически объединить результаты нескольких функций дистанции, чтобы сохранить информацию об их цвете?

10. Как влияет количество итераций теневого луча на общее время рендеринга кадра?

11. Как реализовать тени от нескольких источников света?

12. Почему тени в raymarching выглядят 'идеально' по сравнению с shadow maps в растеризации?

ЛАБОРАТОРНАЯ РАБОТА №7. ПРОЦЕДУРНАЯ ГЕНЕРАЦИЯ И ШУМ

Цели и задачи

Цель лабораторной работы: изучение функций псевдослучайного шума для создания органических текстур и сложных ландшафтов.

Основные задачи:

– реализовать базовые функции хэширования для получения случайных чисел на gpu;

– изучить структуру и применение шума (value noise, gradient noise);

– освоить технику fbm для создания детализированных природных объектов.

Формируемые компетенции: ПК-1, ПК-2, ПК-4

Теоретическое обоснование

Перед выполнением лабораторной работы необходимо ознакомиться с лекционным материалом.

Оборудование и материалы

Для выполнения лабораторной работы рекомендуется использовать персональный компьютер со следующими программными средствами разработки (выбрать один или несколько программных продуктов для практической реализации задач лабораторной работы): google chrome / mozilla firefox, visual studio code, clion / microsoft visual studio 2022.

Методика и порядок выполнения работы

Перед выполнением заданий рекомендуется ознакомиться с решением учебной задачи:

https://github.com/vberezina/PCG/blob/main/LabWork7.md

Порядок выполнения работы:

1. Процедурный шум в шейдерах строится на основе детерминированных хэш-функций.

2. Для генерации ландшафтов используется наложение нескольких октав шума (fbm — fractional brownian motion), где каждая последующая октава имеет удвоенную частоту и уменьшенную амплитуду.

3. При работе с 3d-ландшафтами функция шума встраивается непосредственно в sdf сцены как деформация поверхности.

Задание 1. Хэш-функции и белый шум.

Создание фундамента для всех процедурных эффектов.

– Написать функцию hash21(p), принимающую вектор vec2 и возвращающую псевдослучайное число [0, 1].

– Визуализировать статический 'белый шум' и анимировать его, добавив iTime в качестве зерна (seed).

– Создать маску для 'звездного неба', используя функцию step() от значения шума.

Задание 2. Value noise и интерполяция.

Переход от резкого шума к плавным структурам.

– Реализовать функцию 2d-шума, вычисляющую случайные значения в углах сетки и интерполирующую их.

– Сравнить визуальный результат при использовании линейной

интерполяции и функции smoothstep.

– Создать эффект 'плазмы', комбинируя несколько движущихся слоев шума.

Задание 3. Фрактальный шум и ландшафты.

Генерация сложных самоподобных структур.

– Написать функцию fbm(p), суммирующую 5–8 октав шума.

– В 3d-сцене использовать fbm для изменения координаты y плоскости, превращая её в рельеф.

– Реализовать раскраску ландшафта по высоте: синий (вода), желтый (песок), зеленый (трава), белый (снег).

Содержание отчета и его форма

Отчет по лабораторной работе должен содержать:

1. Номер и название лабораторной работы; задачи лабораторной работы.

2. Реализация каждого задания с приведением исходного кода программы, диаграмм и графиков для визуализации данных.

3. Ответы на контрольные вопросы.

Отчет о выполнении лабораторной работы подписывается студентом и сдается преподавателю.

Контрольные вопросы

1. В чем отличие функции hash() от стандартной функции rand() в c++?

2. Почему для графики важно, чтобы функция шума была детерминированной (возвращала один результат для одних входных данных)?

3. Что такое билинейная интерполяция и как она применяется в алгоритме шума?

4. Как параметры 'лакунарность' (lacunarity) и 'персистентность' (persistence) влияют на вид fbm?

5. Почему шум перлина (perlin noise) часто выглядит лучше, чем обычный value noise?

6. Как использовать шум для имитации эффекта тумана или облаков?

7. Как шум помогает бороться с визуальной повторяемостью тайловых текстур?

8. Каким образом можно оптимизировать расчет fbm, если требуется много октав?

9. Как реализовать 'турбулентный' шум (функция ridge/abs)?

10. Почему при использовании шума в sdf-сценах может нарушаться условие кратчайшего расстояния?

11. Как создать 'бесконечный' ландшафт, по которому можно лететь?

12. Как привязать шум к мировым координатам, чтобы он не 'плавал' при движении камеры?

ЛАБОРАТОРНАЯ РАБОТА №8. ОПТИМИЗАЦИЯ И ПОСТ-ЭФФЕКТЫ

Цели и задачи

Цель лабораторной работы: визуализация рекурсивных геометрических систем и применение методов финальной постобработки изображения.

Основные задачи:

– изучить принципы построения итерационных фракталов в 3d;

– освоить методы сглаживания изображения (ssaa/msaa);

– научиться применять финальную цветокоррекцию.

Формируемые компетенции: ПК-1, ПК-2, ПК-4

Теоретическое обоснование

Перед выполнением лабораторной работы необходимо ознакомиться с лекционным материалом.

Оборудование и материалы

Для выполнения лабораторной работы рекомендуется использовать персональный компьютер со следующими программными средствами разработки (выбрать один или несколько программных продуктов для практической реализации задач лабораторной работы): google chrome / mozilla firefox, visual studio code, clion / microsoft visual studio 2022.

Методика и порядок выполнения работы

Перед выполнением заданий рекомендуется ознакомиться с решением

учебной задачи:

https://github.com/vberezina/PCG/blob/main/LabWork8.md

Порядок выполнения работы:

1. Фракталы в raymarching реализуются через итеративное преобразование пространства (складывание, вращение, масштабирование координат p) перед вычислением базовой sdf.

2. Из-за высокой детализации таких сцен требуется уменьшение шага луча.

3. Финальный этап работы включает добавление эффектов постобработки (антиалиасинг, гамма-коррекция), которые выполняются над итоговым цветом пикселя.

Задание 1. Фрактальный куб (menger sponge).

Использование итеративных вычитаний для создания губчатых структур.

– Реализовать функцию sdf для куба с итеративным делением пространства (использование функций abs и масштабирования).

– Добавить в цикл трансформации небольшое вращение на каждой итерации для создания сложной органической формы.

– Визуализировать фрактал с использованием мягких теней.

Задание 2. Сглаживание (anti-aliasing).

Устранение ступенчатых артефактов на границах объектов.

– Реализовать суперсэмплинг (ssaa), усредняя цвет из 4-х дополнительных лучей, пущенных со смещением внутри пикселя.

– Добавить условие включения сглаживания только для границ объектов

для экономии ресурсов.

– Сравнить визуальное качество картинки с аа и без него.

Задание 3. Финальный рендер и цветокоррекция.

Приведение изображения к художественному виду.

– Реализовать функцию гамма-коррекции (pow(color, 1.0/2.2)) для правильного отображения яркости на мониторе.

– Добавить эффект виньетирования (затемнения краев экрана) и легкое хроматическое искажение (aberration).

– Настроить экспозицию и контрастность финального кадра.

Содержание отчета и его форма

Отчет по лабораторной работе должен содержать:

1. Номер и название лабораторной работы; задачи лабораторной работы.

2. Реализация каждого задания с приведением исходного кода программы, диаграмм и графиков для визуализации данных.

3. Ответы на контрольные вопросы.

Отчет о выполнении лабораторной работы подписывается студентом и сдается преподавателю.

Контрольные вопросы

1. Почему при отрисовке фракталов стандартный raymarching может выдавать артефакты 'шума' или пропуски поверхности?

2. Что такое 'коэффициент масштабирования' в итерационном фрактале и как он влияет на итоговую дистанцию?

3. Как антиалиасинг методом суперсэмплинга (ssaa) влияет на нагрузку на видеокарту?

4. Почему гамма-коррекция важна для корректного смешивания цветов и теней?

5. Что такое 'световая ловушка' (orbit traps) и как она используется для раскраски фракталов?

6. Как ограничить количество итераций во фрактале для достижения баланса между деталями и скоростью?

7. В чем отличие экранных пост-эффектов от вычислений внутри цикла raymarching?

8. Как реализовать эффект 'глубины резкости' (depth of field) в шейдере?

9. Каким образом можно имитировать эффект линзы рыбий глаз (fisheye)?

10. Почему в шейдерах часто используют значение гаммы 2.2?

11. Как реализовать bloom-эффект (свечение ярких областей) на одном проходе шейдера?

12. Какие техники оптимизации (bounding volumes) можно применить для очень сложных фрактальных сцен?

СПИСОК ЛИТЕРАТУРЫ

Основная литература:

1. Афанасьев Г.И. Основы программирования графических процессоров: учебное пособие. — Спб: Лань, 2023

2. Боресков А.В. Расширения OpenGL. — М.: БХВ-Петербург, 2005. (главы по glsl и шейдерам)

3. Ростовцев В.А. Математические основы компьютерной графики. — М.: Издательство юрайт, 2024

4. The book of shaders. Patricio Gonzalez Vivo and Jen Lowe. [электронный ресурс] — https://thebookofshaders.com/

Дополнительная литература:

5. Inigo Quilez. Articles on computer graphics and fractals. [электронный ресурс] — https://iquilezles.org/articles/

6. Gpu gems 1, 2, 3. Randima Fernando, Matt Pharr, Hubert Nguyen. Nvidia press. [электронный ресурс] — https://developer.nvidia.com/gpugems/

7. Real-time rendering. Tomas Akenine-Moller, Eric Haines, Naty Hoffman. 4th edition, a k peters, 2018

8. Shadertoy beta. Official documentation and community tutorials. [электронный ресурс] — https://www.shadertoy.com/howto

Интернет-ресурсы:

9. Платформа shadertoy:

https://www.shadertoy.com/

Основная среда для написания, тестирования и обмена фрагментными шейдерами на языке glsl с доступом к мировому сообществу разработчиков

10. The book of shaders (патрисио гонсалес виво):

https://thebookofshaders.com/

Пошаговое руководство по созданию процедурной графики и фрагментных шейдеров с интерактивными примерами и визуализацией кода

11. Статьи иниго килеса по компьютерной графике:

https://iquilezles.org/articles/

Фундаментальный образовательный ресурс по функциям дистанции (sdf), алгоритмам raymarching, освещению и математической оптимизации шейдеров

12. Scratchapixel 3.0:

https://www.scratchapixel.com/

Образовательная платформа, объясняющая математику и алгоритмы компьютерной графики с нуля: от векторов до построения полноценных систем рендеринга

13. Серия книг gpu gems (nvidia):

https://developer.nvidia.com/gpugems/gpugems/part-i-natural-effects/chapter-1-effective-water-simulation-physical-models

Сборник глубоких технических статей от ведущих инженеров индустрии о продвинутых техниках рендеринга и физических симуляциях на gpu

14. Спецификация opengl/glsl (khronos group):

https://registry.khronos.org/OpenGL/specs/gl/glspec46.core.pdf

Официальная техническая документация по стандарту языка шейдеров для точного понимания типов данных, точности вычислений и встроенных функций

15. Vulkan tutorial:

https://vulkan-tutorial.com/

Руководство по низкоуровневому графическому api, полезное для понимания современного конвейера рендеринга, который стоит за шейдерными языками