**Министерство науки и высшего образования Российской Федерации**

**ФГАОУ ВО «УрФУ имени первого Президента России Б.Н. Ельцина»**

Кафедра Школа наук ИЕНиМ

Оценка работы \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель от УрФУ Ушакова О.В.

Тема задания на практику

Исследование технических средств для визуализации сеточных данных с помощью редактора узлов

ОТЧЕТ

Вид практики Преддипломная практика

Тип практики Производственная практика, научно-исследовательская работа

Руководитель практики Ушакова О.В., д.ф.-м.н., профессор

**ФИО руководителя Подпись**

Студент Вяткин Д.Е.

**ФИО студента**

Специальность (направление подготовки) 09.04.03 Прикладная информатика в аналитической экономике

Группа МЕНМ-290901

Екатеринбург 2021

ОГЛАВЛЕНИЕ

[ВВЕДЕНИЕ 2](#_Toc72763294)

[ОСНОВНАЯ ЧАСТЬ 3](#_Toc72763295)

[1 Концепция редактора узлов 4](#_Toc72763296)

[2 Описание редактора узлов 5](#_Toc72763297)

[3 Генерация шейдеров 7](#_Toc72763298)

[4 Библиотека узлов-операций 8](#_Toc72763299)

[4 Примеры 9](#_Toc72763300)

[ЗАКЛЮЧЕНИЕ 14](#_Toc72763301)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 15](#_Toc72763302)

# ВВЕДЕНИЕ

В рамках прохождения преддипломной практики в весеннем семестре была поставлена задача завершения разработки специального программного обеспечения для визуализации сеточных данных, применяемых при изучении математических объектов — расчетных сеток. Для этого были использованы результаты, полученные в прохождении производственных и учебных практик в предыдущих семестрах.

Программа практики включала в себя изучение принципов работы редакторов узлов, определение методов их применения для научной визуализации. После поиска информации и планирования работы последовала разработка системы редактора узлов. По промежуточным результатам работы были запланированы два доклада на семинарах сектора визуализации ИММ УрО РАН, а также участие на молодежной конференции по современным проблемам математики и ее приложениям.

В рамках практик был проведен ряд экспериментов, направленных на обеспечение гибкости и расширяемости разрабатываемой системы. Концепция редактора узлов — продолжение этих разработок.

# ОСНОВНАЯ ЧАСТЬ

**1 Концепция редактора узлов**

В рамках предыдущей практики была исследована концепция материалов, в рамках которой отображение сетки задается комбинацией алгоритма отображения и набора настраиваемых параметров. Было показано, что такой подход обладает определенным потенциалом для решения поставленных задач. При этом, существует недостаток, мешающий полной реализации этого потенциала.

Схема, которая требует ручное написание шейдеров (программ, выполняемых на графическом ускорителе), подразумевает наличие у пользователя базовых знаний понятий программирования, а также требует изучение концепций программирования шейдеров и языка GLSL. Это недостаточно дружелюбный к пользователю подход, который может привести к возникновению множества мелких ошибок, что замедлит работу и не позволит сфокусироваться на цели исследователя, то есть изучении расчётных сеток.

Одним из возможных путей могло быть развитие синтаксического анализа написанных шейдеров и создание расширенного диалекта языка GLSL, специфичного для данной системы [1]. Но это не решило бы обозначенную проблему. Во-первых, дополнительные синтаксические конструкции не упрощают изучение работы с программой. Во-вторых, расширение синтаксиса шейдерного языка делает разрабатываемый пакет несовместимым с другими программами.

Необходим способ дальнейшего упрощения создания новых визуальных отображений. Это приводит к выводу, что нужен некоторый визуальный способ задания алгоритмов для материалов, то есть некоторый аналог визуального программирования.

Если обратиться к популярным программам для 3D-моделирования, в них чаще всего можно увидеть редактирование материалов, основанное на создании графов, состоящих из узлов и связей между ними [2]. Пример редактора материала в программе Blender можно увидеть на рисунке 1.

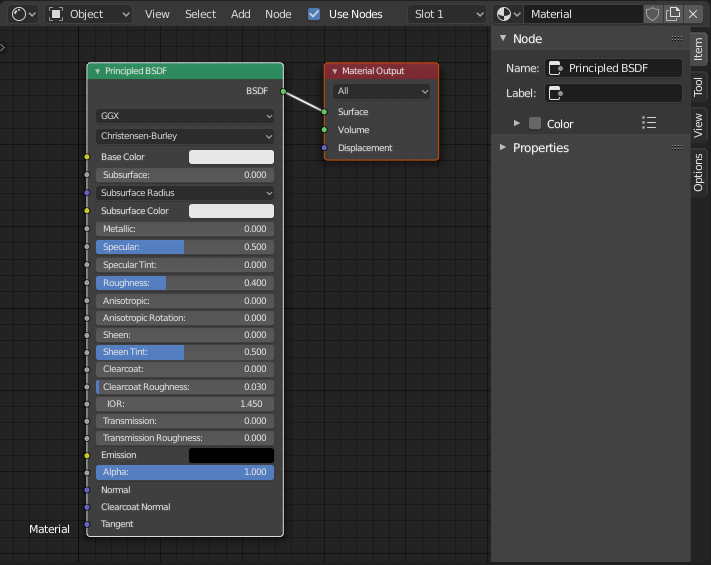


Рисунок 1 — визуальный редактор материалов в программе Blender.

**2 Описание редактора узлов**

Как было ранее сказано, редактор узлов включает в себя два типа сущностей. Это узлы и связи.

Узел представляет собой прямоугольный блок. Узел обладает типом, определяющим функциональное предназначение узла и его визуальное отображение.

Есть несколько типов узлов:

1. uniform-узлы — предназначены для настройки uniform-переменных, общих для всей сетки [3];
2. узлы-атрибуты — предназначены для получения данных, хранящихся в атрибутах, которые меняют значения для каждого узла сетки [3];
3. узлы-операции — предназначены для преобразования входных данных и получения выходных;
4. выходные узлы — предназначены для получения итогового результата работы материала.

У каждого узла может быть несколько входов и выходов. У входных uniform-узлов и узлов-атрибутов есть только выходы. У выходных узлов есть только входы. У узлов-операций может быть произвольное число входов и выходов. У каждого входа и выхода есть тип данных. Набор типов данных ограничен типами данных, доступных в языке GLSL.

Узлы соединяются между собой с помощью связей через входы и выходы. Типы данных соединенных входа и выходы должны совпадать. Связи между узлами указывают на направление движения данных. При создании связей следует избегать появления циклов, так как такие ситуации не имеют смысла в рассматриваемой концепции. Говоря формально, корректный материал представляет собой ациклический направленный граф.

У пользователя есть возможность определять собственные узлы с помощью создания файлов, содержащих конфигурацию нового узла. Об этом написано более подробно далее.

Пример материала, настроенного в редакторе узлов, можно увидеть на рисунке 2. Это простейший материал, который совершает преобразование некоторого настроенного цвета в вектор, который затем используется в качестве результата работы материала.

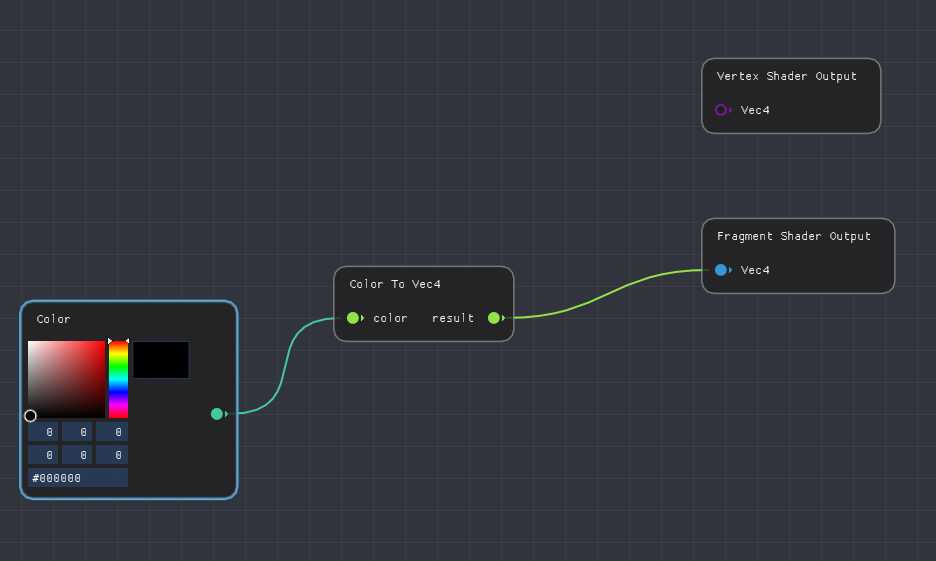


Рисунок 2 — пример простейшего материала, созданного в редакторе узлов.

**3 Генерация шейдеров**

Графы материалов, настроенные в редакторе узлов, преобразуются в шейдеры [4]. Таким образом достигается отображение материала. Параметры, настраиваемые через входные узлы, в реальном времени передаются на графический ускоритель.

Основная техническая сложность заключается в процессе преобразования графа в исходный код шейдера. Остальные элементы системы в целом уже были реализованы в рамках предыдущих практик.

Процесс генерации исходного кода шейдера выглядит следующим образом:

1. находится выходной узел материала;
2. начинается рекурсивный обход ациклического направленного графа в глубину;
3. если посещаемый узел — узел-операция, то в главную функцию шейдера добавляется новая часть кода в соответствии с типом операции, про то, как обрабатываются узлы операции, можно подробнее прочитать в следующем подразделе;
4. если посещаемый узел — uniform-узел или узел-атрибут, то в код шейдера добавляется объявление uniform-переменной либо атрибута соответственно;
5. при возвращении из рекурсии для каждого узла-операции подставляются переменные в код операции в нужных местах исходя из того, с какими узлами связаны входы данного узла;
6. на последнем этапе отдельные секции кода шейдера, такие как объявления атрибутов, объявления uniform-переменных, главная функция шейдера, объединяются в единый шейдер, добавляются необходимые технические спецификации.

**4 Библиотека узлов-операций**

Каждый тип узла-операции представляет собой некоторое преобразование данных. Важный момент для обеспечения расширяемости — возможность добавлять новые виды преобразований.

Все узлы-операции организованы в библиотеку текстовых файлов, где каждый файл определяет один узел-операцию. Добавление, удаление и изменение текстовых файлов — понятный для каждого пользователя компьютера процесс. Некоторый набор стандартных узлов-операций поставляется вместе с программой, пользователи вольны добавлять собственные и делиться ими друг с другом.

Файл любого корректного узла-операции содержит описание входов, выходов узла и выполняемый код, подставляемый в исходный код шейдера. Пример описания простого узла, выполняющего умножения вектора на число:

Multiply Vec4

in vec4 vector

in float value

out vec4 result

vec4 {2} = {0} \* {1};

Первая строчка содержит название операции, которое видит пользователь при выборе операции из меню. Далее несколько строчек определяют входы и выходы узла. Сначала идут входы, обозначаемые ключевым словом «in» в начале строки, затем выходы, обозначаемые ключевым словом «out». Для каждого входа и выхода необходимо указать тип данных и осмысленное имя. Остаток файла содержит исходный код, который будет подставлен в исходный код шейдера. Он может занимать произвольное количество строчек. В местах кода, где должны быть подставлены определенные ранее входные и выходные переменные, указываются числа, указывающие на индекс переменной в списке, заключенные в фигурные скобки.

**4 Примеры**

Ниже можно увидеть пример отображения сетки с помощью графа, показанного на рисунке 2.

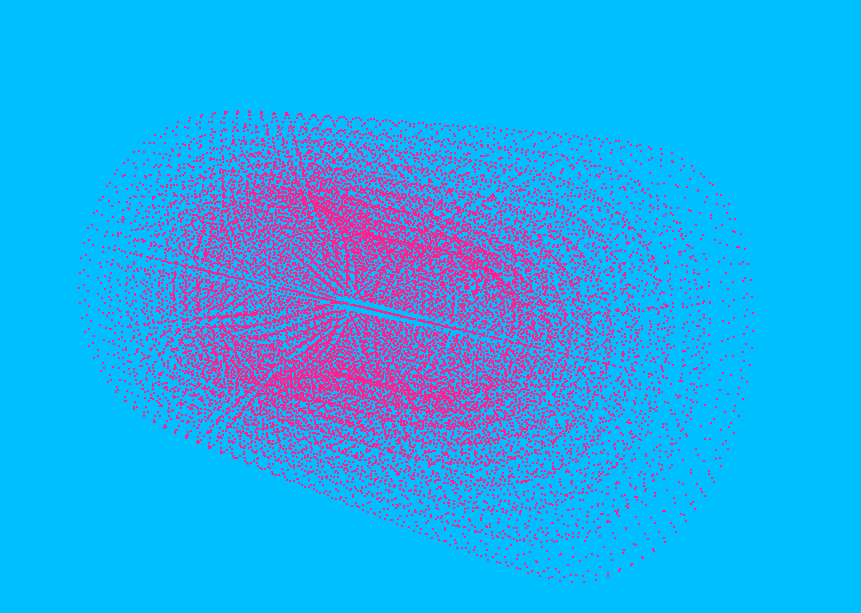


Рисунок 3 — простейшее отображение сетки одним цветом.

Это минимальный пример, который позволяет просто отобразить данные каким-то одним настраиваемым цветом.

Пример отображения некоторого параметра, заданного для каждого узла сетки можно увидеть на рисунке 4. На рисунке 5 проиллюстрирован граф в редакторе узлов.

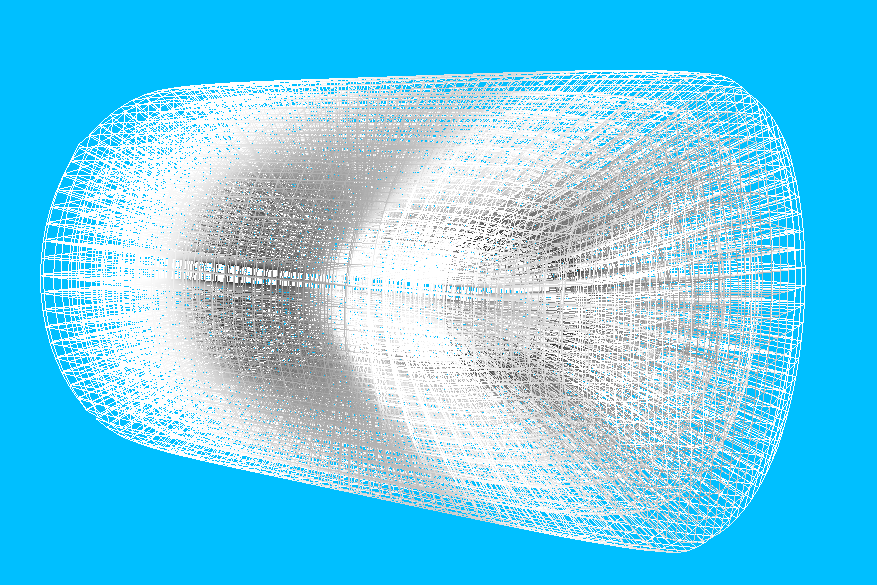


Рисунок 4 — отображение с параметром.

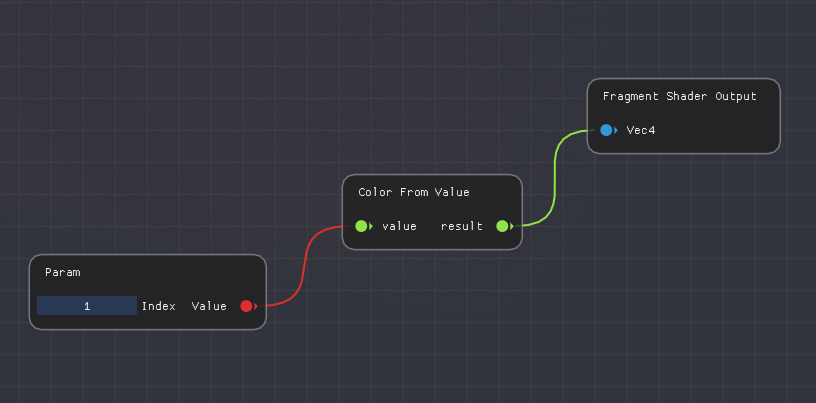


Рисунок 5 — настройка отображения с параметром.

Если поменять отображаемый параметр, соответственно меняется отображение, как видно на рисунке 6 (видно небольшие вкрапления темных оттенков серого цвета внутри сетки).

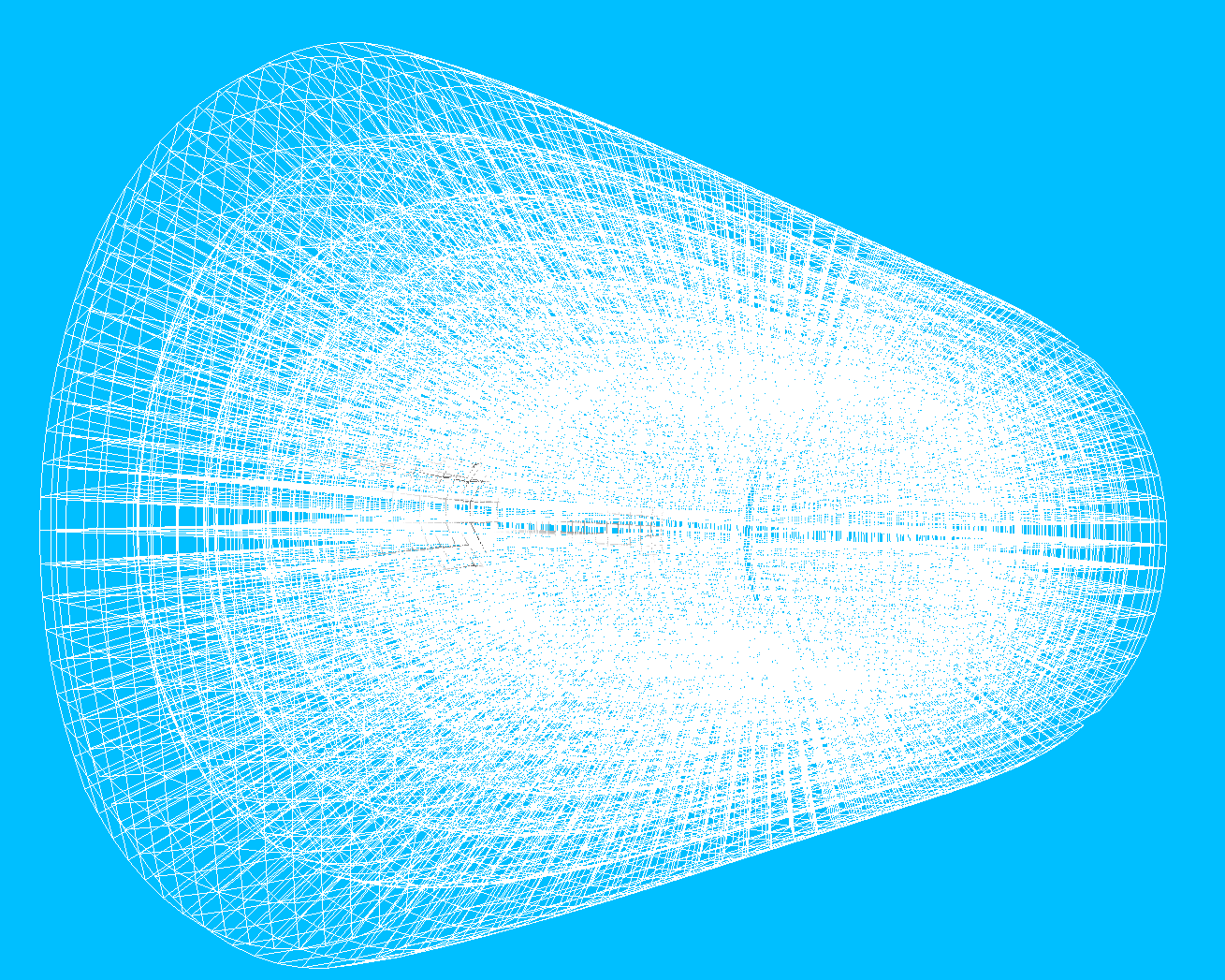


Рисунок 6 — отображение с другим параметром.

Объединив отображения данных цветом и параметром, как показано на рисунке 8, мы получим результат, который видно на рисунке 7. Как видно, комбинирование различных операций происходит посредством соединения заранее созданных отображений с помощью некоторого объединяющего узла.

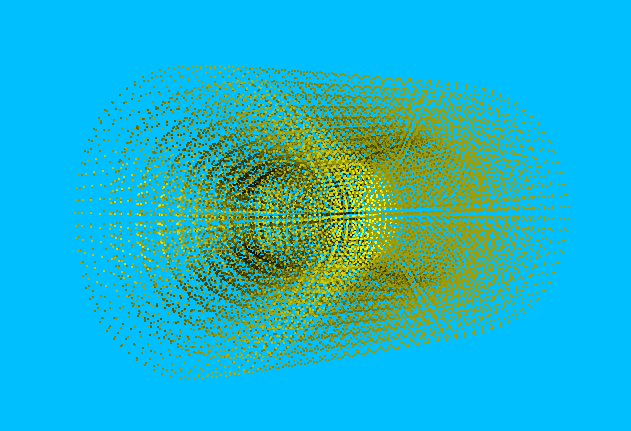


Рисунок 7 — отображение с параметром и цветом.

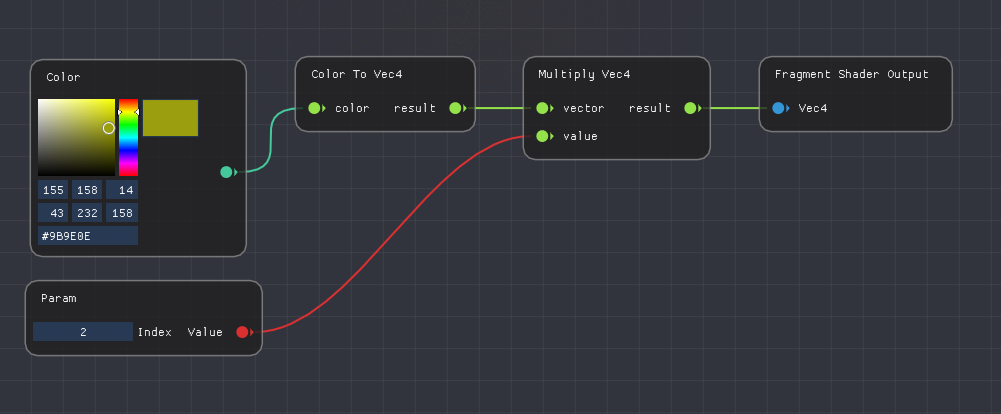


Рисунок 8 — настройка отображения с параметром и цветом.

С помощью условных узлов мы можем некоторым образом фильтровать данные, манипулирую значениями индексов узлов. Пример показан на рисунках 9 и 10. Если граф становится громоздким, можно выделить часто используемые наборы операций в отдельные узлы.

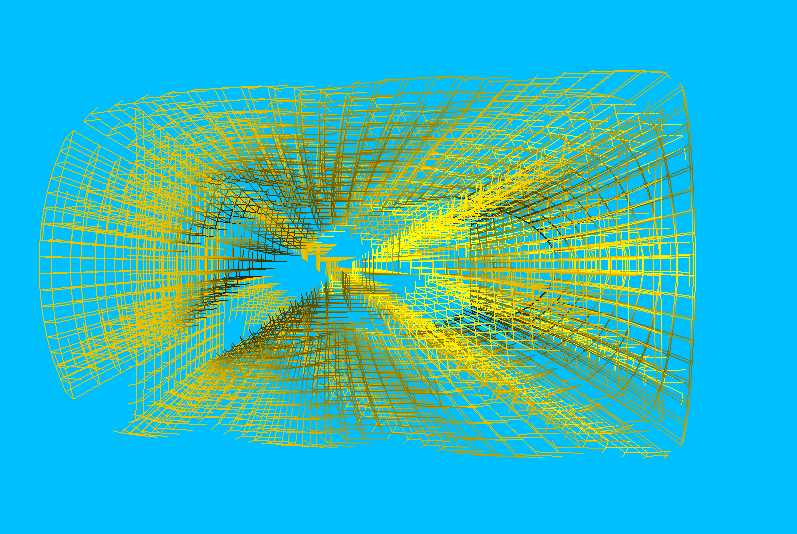


Рисунок 9 — простейшая фильтрация сетки по индексам узлов.

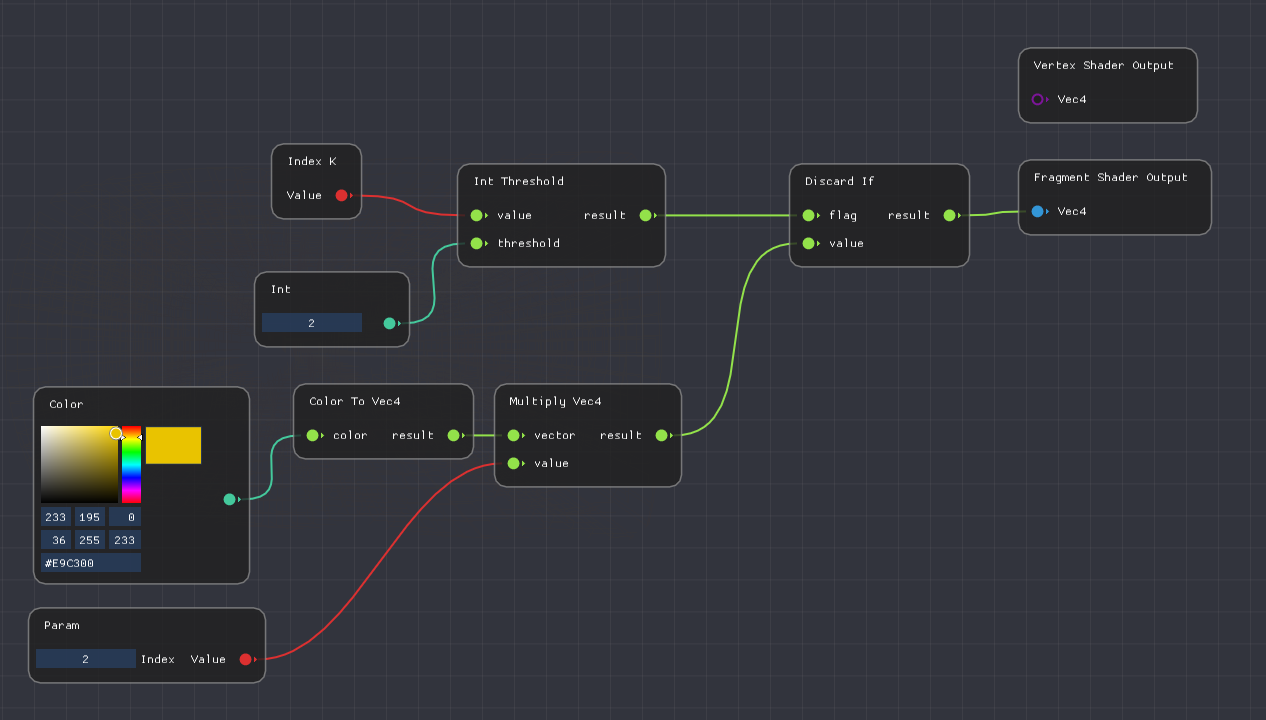


Рисунок 10 — настройка фильтрации сетки по индексам узлов.

# ЗАКЛЮЧЕНИЕ

В результате прохождения преддипломной практики была разработана система редактора узлов для научной визуализации расчетных сеток. Полученные результаты в данной и предыдущих практиках будут систематизированы и полностью в рамках магистерской диссертации.

С помощью редактора узлов достигается возможность визуального задания новых визуальных отображений. Такой подход делает систему доступной для широкого круга пользователей. Возможность сохранения и загрузки материалов позволяет делиться результатами работы с коллегами.

В теории, разработанный редактор узлов может быть использован не только для расчетных сеток в предоставленном формате данных, но и для других видов научных данных.

В течение семестра были подготовлены и представлены два доклада по промежуточным результатам выполненной работы в рамках семинаров сектора компьютерной визуализации ИММ УрО РАН. На первом семинаре обсуждались методические вопросы применения редактора узлов и были показаны практические примеры. На втором семинаре были рассмотрены технические и алгоритмические вопросы реализации данной системы.

Автор принял участие в молодежной научной конференции по современным проблемам математики и ее приложений, где также обсуждались вопросы, связанные с применением редактора узлов для визуализации научных данных.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Shreiner D., OpenGL Programming Guide // Addison-Wesley, Boston, 2013.
2. Wright R.S., Haemel N., Sellers G., Lipchak B., OpenGL SuperBible // Addison-Wesley, Boston, 2010.
3. Bailey M., Cunningham S., Graphics Shaders. Theory and Practice. Second Edition // CRC Press, New York, 2012.
4. Lengyel E., Mathematics for 3D Game Programming and Computer Graphics // Course Technology PTR, Boston, 2012.