СОЗДАНИЕ ЭФФЕКТА КОНТУРА ДЛЯ ОБЪЕКТОВ С ИСПОЛЬЗОВАНИЕМ МАТЕРИАЛА ПОСТОБРАБОТКИ НА UNREAL ENGINE 4

CREATING OUTLINE EFFECT FOR OBJECTS USING POST-PROCESSED MATERIAL ON UNREAL ENGINE 4

УДК 004.021

Черняев Роман Валерьевич, студент 4 курс, факультет «Робототехника и комплексная автоматизация», Московский государственный технический университет имени Н.Э. Баумана, Россия, г. Москва

Chernyaev Roman Valerievich, kayl.sander@gmail.com

Аннотация: В статье рассматриваются возможные способы создания эффекта выделения границ объекта и аргументируется выбор метода, основанного на механизме постобработки в Unreal Engine 4. Описывается алгоритм работы выбранного способа рендеринга, более подробно раскрывается его суть с использованием пользовательской карты глубины графического движка, сравнение которой производится со стандартными механизмами отрисовки объектов. Демонстрируется процесс добавления материала постобработки в Unreal Engine 4, наполнения его необходимыми пользовательскими функциями, реализующими этапы математического исчисления в алгоритме, и создания объекта на сцене, отвечающего за активацию эффектов постобработки. Также в статье приводятся графические примеры финального результата и выводы, полученные в ходе работы над материалом постобработки.

Annotation: The article discusses possible ways to create the effect of highlighting object boundaries and justifies the choice of a method based on the post-

processing mechanism in Unreal Engine 4. The algorithm of the selected rendering method is described, its essence is described in more detail using a custom depth map of the graphics engine, which is compared with standard mechanisms drawing objects. The process of adding post-processing material to Unreal Engine 4, filling it with the necessary custom functions that implement the stages of mathematical calculus in the algorithm, and creating an object on the scene responsible for activating post-processing effects is demonstrated. The article also provides graphic examples of the final result and conclusions obtained during the work on the post-processing material.

Ключевые слова: графический движок, контур, эффект выделения, оператор собеля, материал постобработки, глубина отрисовки, рэндеринг, фильтр.

Keywords: graphics engine, outline, highlight effect, sobel operator, post-processing material, draw depth, rendering, unreal engine 4, ue4.

Введение

В этой статье я хотел бы представить способ создания эффекта выделения объектов по контуру (outline эффект) на Unreal Engine 4. На данный момент в сети можно найти два достаточно популярных метода создания контура вокруг объекта или outline эффекта [2]:

- Двойной рендеринг объектов.
- Добавление контура к объектам в постобработке.

Суть первого метода заключается в использовании двух объектов. Первый, являющийся основным, рендерится на переднем плане, а второй, обладающий чуть большим размером, рендерится позади, тем самым создавая эффект свечения границы.

Второй же метод использует алгоритмы нахождения границ объектов, отличающийся куда большей простотой интеграции в другие проекты посредством ассета и меньшей требовательностью к ресурсам движка, поэтому было принято решение остановить выбор на нём [3].

Предполагается, что при прочтении данной статьи пользователь уже знаком с основными понятиями движка Unreal Engine 4 (в частности – с редактором материалов и постобработкой). Для полного понимания работы и реализации алгоритма обнаружения краёв объектов в качестве материала Unreal Engine 4 Вам пригодятся базовые знания по обработке изображений.

Основные шаги

Для реализации эффекта контура на основе постобработки необходимо выполнить следующие шаги:

- Создать материал для постобработки, который добавит эффект обводки для объектов со включенным параметром «Render Custom Depth».
- Добавить материал постобработки в список материалов класса PostProccessVolume, который должен быть добавлен на сцену.
- Для каждого primitive, skeletal или static объекта, должен быть включён параметр «Render Custom Depth».

Каждый из пунктов будет более подробно разбираться далее.

Пользовательская карта глубины

Одна из замечательных особенностей Unreal Engine 4 – это возможность визуализировать отдельные сетки в отдельную карту глубины, которая будет использоваться в материалах. Карта глубины хранит информацию о расстоянии между каждым отдельным пикселем в мировом пространстве и камерой [4]. Пример типичной карты глубины представлен на рисунке ниже (рис.1).

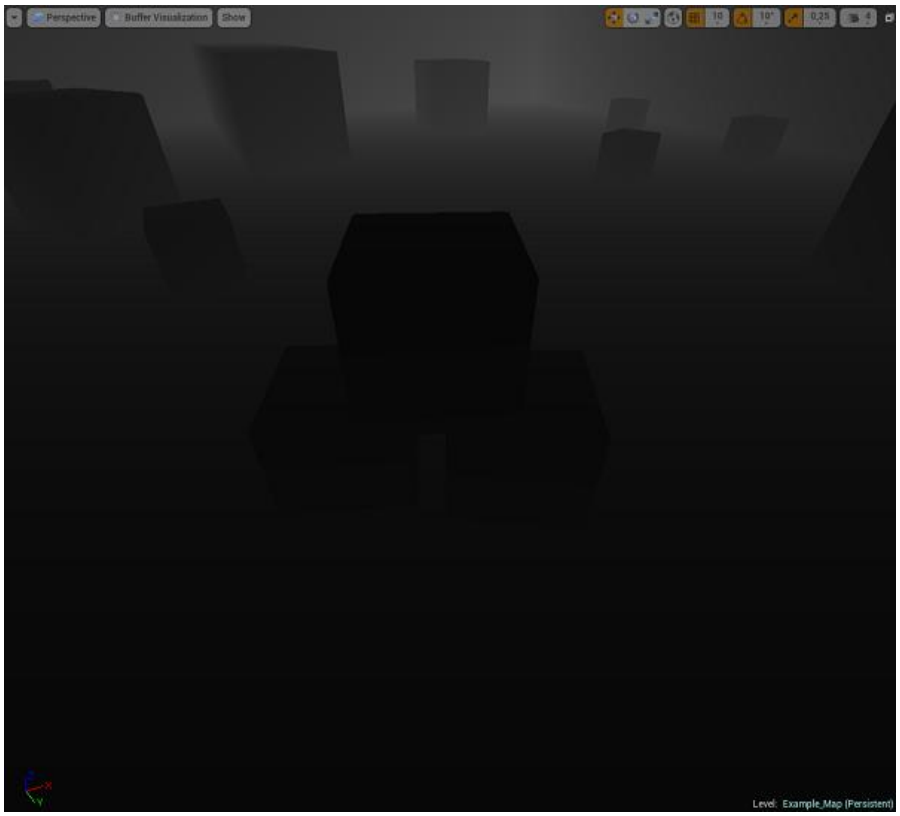


Рисунок 1. Пример визуального представления карты глубины [7].

Пользовательская карта глубины будет выглядеть аналогичным образом (рис.2), за исключением того, что останутся видны только те объекты, для которых в их свойствах выбрана опция «Render Custom Depth». Это позволяет очень удобно использовать карты глубин для реализации таких эффектов, как outline, в пространстве экрана.



Рисунок 2. Пример визуального представления пользовательской карты глубины [7].

Создание материала

Данный пункт является самой сложной частью всей работы. Основой для материала постобработки станет оператор Собеля [1], необходимый для создания свёртки с пользовательской картой глубины. Другими словами, к ней будет применена скользящая маска-фильтр для обнаружения краев, вокруг которых и необходимо рисовать обводку.

Первым делом нужно разобраться с алгоритмом свертки, представленный довольно сложной математикой. Однако, все действия можно свести к следующим шагам:

1. Выбрать пиксель на карте глубины, для которого будет определяться новое значение. (начало свертки)
2. Для выбранного пикселя и всех пикселей, смежных с ним, произвести поэлементное перемножение с пикселями маски-фильтра.
3. Суммировать все найденные произведения

4. Присвоить полученное значение выбранному изначально пикселю карты глубины. (конец свертки)

5. Произвести свертку для каждого пикселя карты глубины.

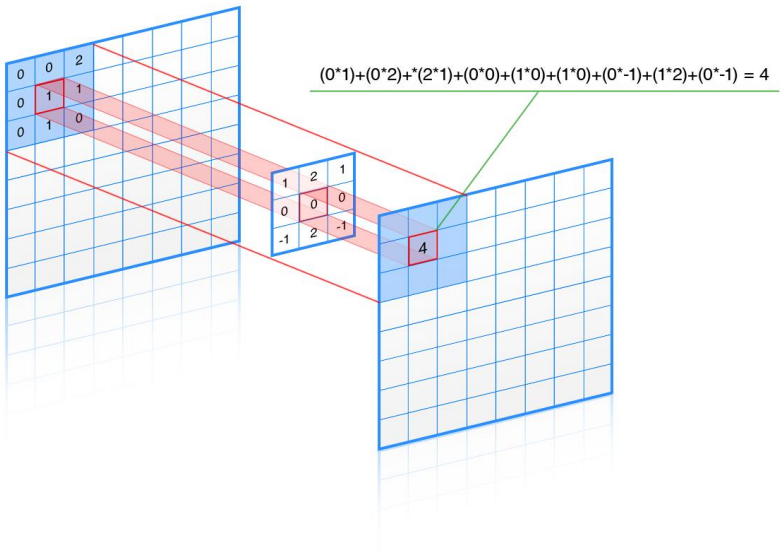


Рисунок 3. Наглядное представление алгоритма свёртки [7].

Сэмплирование пользовательской карты глубины представлено достаточно просто: необходимо поместить в редактор блок SceneTexture и подключить его вывод Color к выходу Emissive (рис.4). В свойствах блока нужно установить для TextureID значение CustomDepth [4].

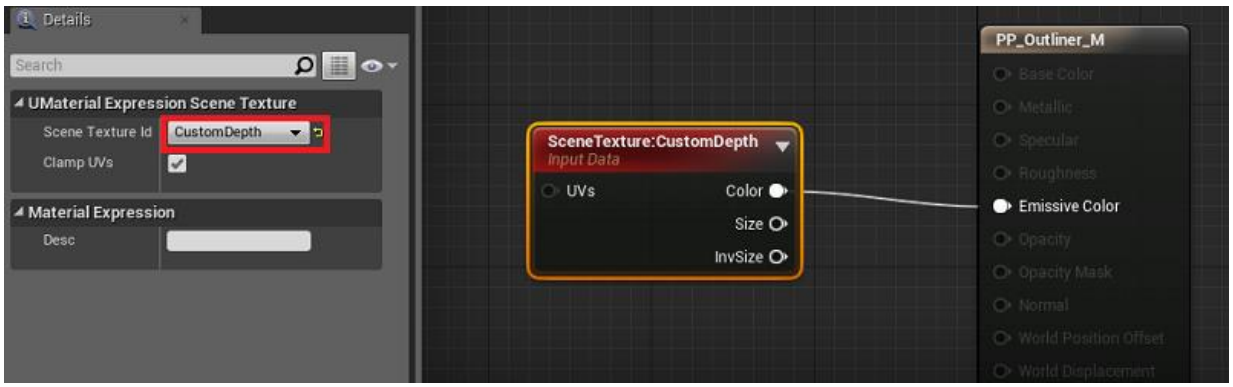


Рисунок 4. Сэмплирование текущего пикселя пользовательской карты глубины [7].

Теперь необходимо сэмплировать соседние пиксели. Для этого можно использовать параметр UV узла SceneTexture. Единственная проблема заключается в том, что параметры UV работает в текстурном пространстве,

принимая значения от верхнего левого угла (0,0) до нижнего правого угла (1,1) текстуры. Чтобы привести всё к правильной размерности, нужно взять обратную ширину и высоту текстуры, умножить на смещение и добавить его в координаты UV текущего пикселя.

В редакторе материалов это будет выглядеть следующим образом (рис.5):

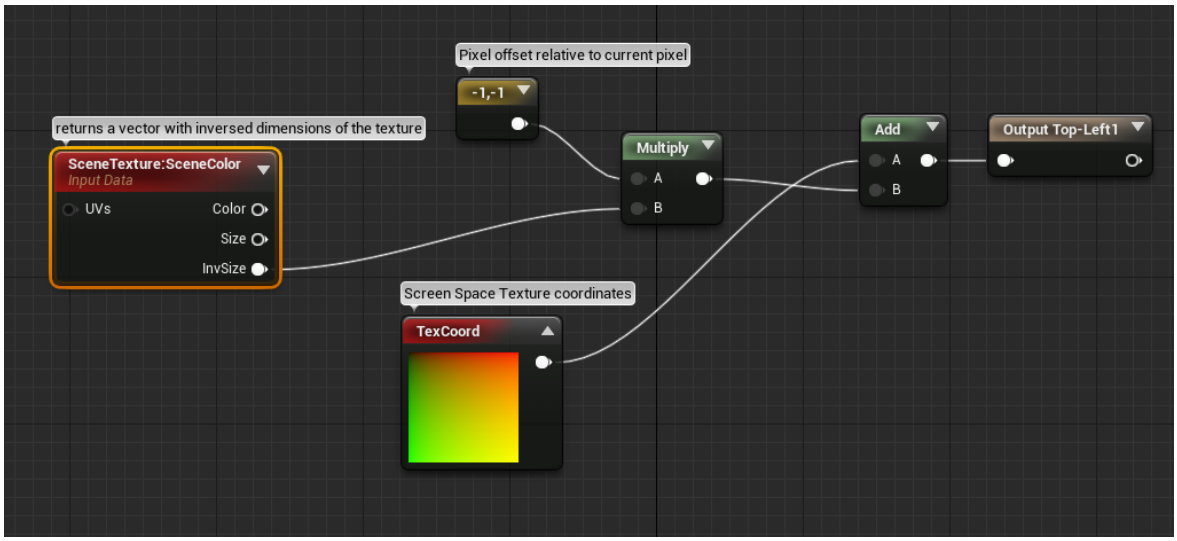


Рисунок 5. Сэмплирование соседнего пикселя карты глубины [7].

После достижения результата необходимо продублировать эти выражения для оставшихся семи смежных пикселей. Окончательная настройка для вычисления смещения пикселей будет выглядеть следующим образом (рис.6):

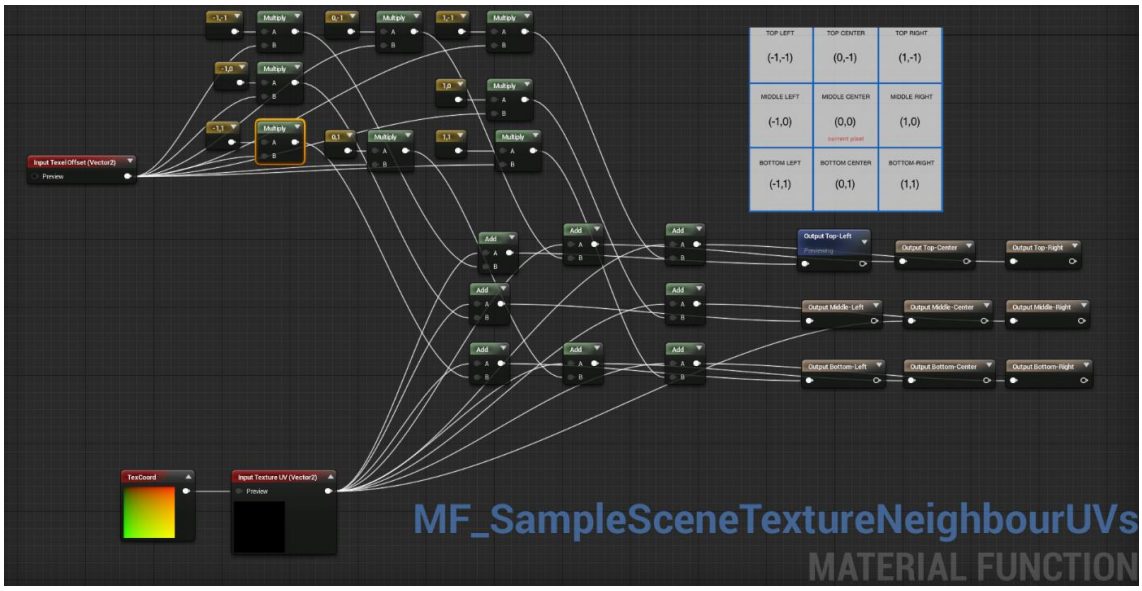


Рисунок 6. Сэмплирование текущего и соседних пикселей карты глубины [7].

После выполнения сэмплирования в нашем распоряжении есть набор UV координат для всех соседних пикселей – это позволяет получать фактические значения каждого пикселя из CustomDepthMap [5]. Для упрощения blueprint схемы было принято решение реализовать простую функцию материала, принимающая на вход UV координаты и возвращающая значение выбранного пикселя (рис.7).

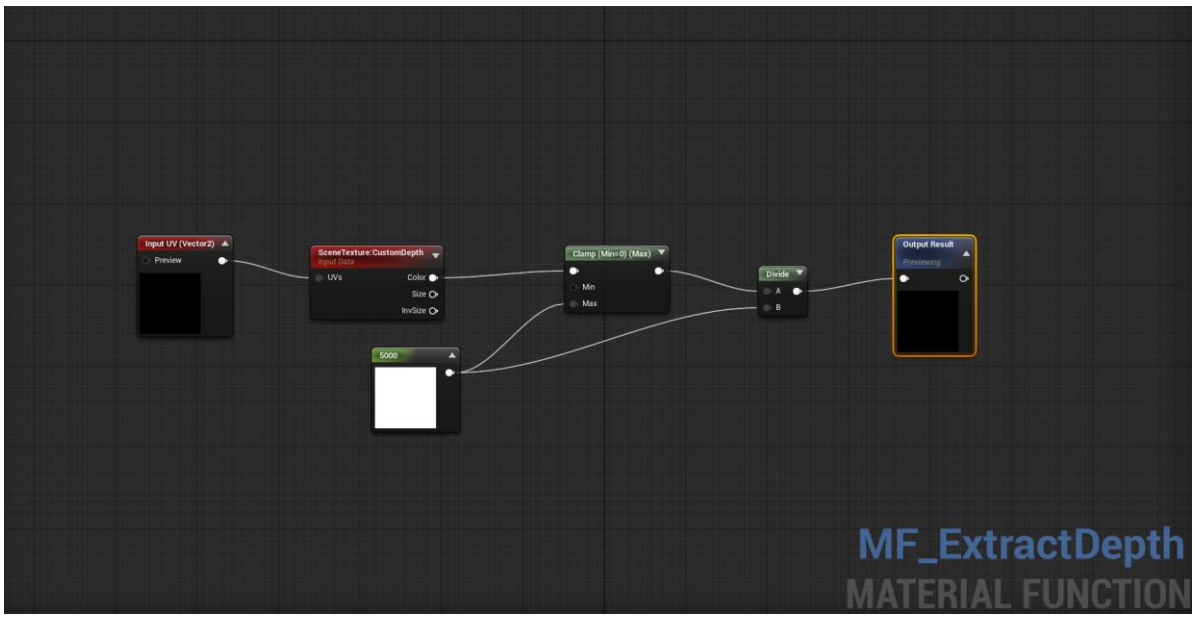


Рисунок 7. Функция получения значений пикселя из карты глубины [7].

Если использовать написанную функцию для выборки смежных значений, можно получить все данные для фактического обнаружения края. Одним из заключительных пунктов осталось создание ещё одной функции материала, которая будет выполнять свертку (рис.8).

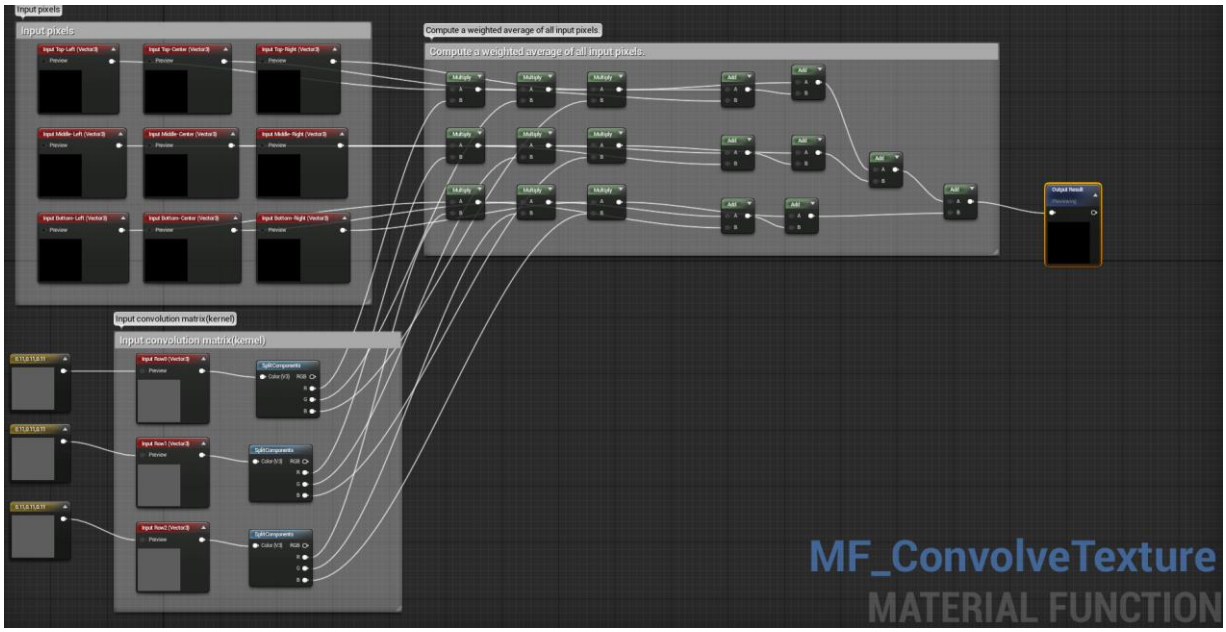


Рисунок 8. Функция материала, выполняющая свёртку [7].

Слева представлены два набора входных параметров. Первые 9 входных векторов – это значения пикселей, для которых будет производиться обработка. Остальные три параметра снизу предоставляют значения нашего ядра. На самом деле таким образом просто представлена матрица 3×3, но, поскольку в редакторе материалов Unreal Engine 4 нет типа «матрица», пришлось воспользоваться тройкой переменных типа Vector3 [6]. На текущем этапе мы уже имеем все необходимые нам части для реализации конечного варианта алгоритма, поэтому последним шагом нужно просто собрать всё воедино в материале (рис.9).

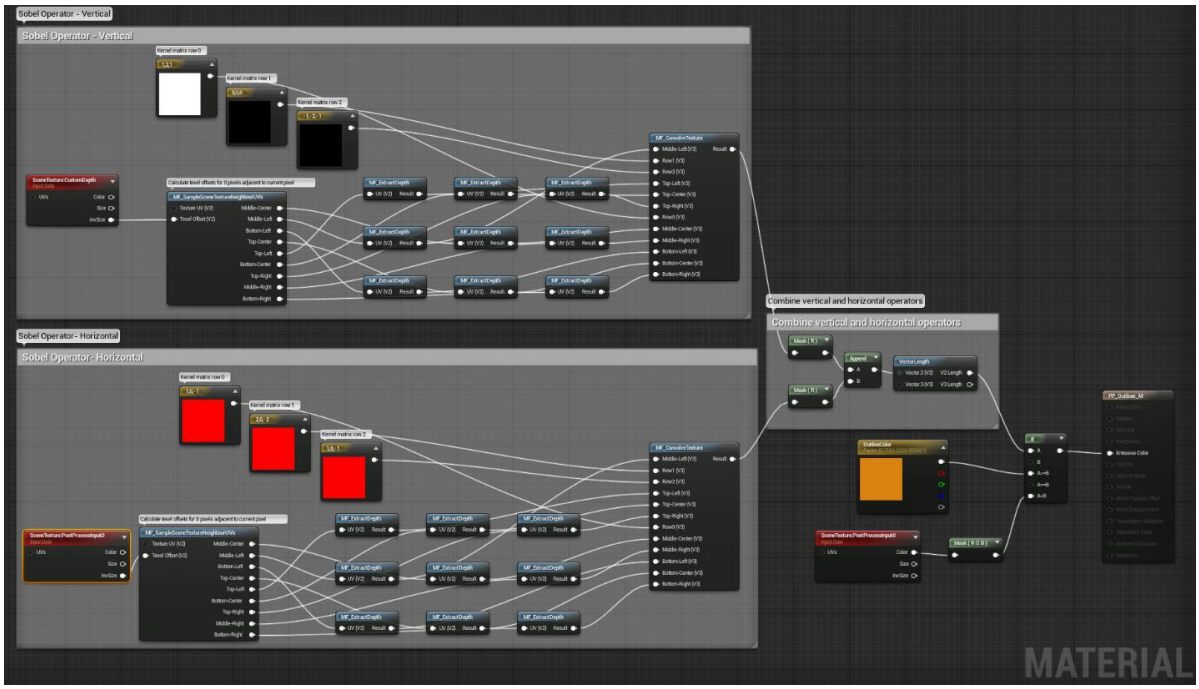


Рисунок 9. Финальный вид создаваемого материала постобработки [7].

Оператор Собеля фактически включает в себя две операции свертки: вертикальную и горизонтальную [1]. Можно заметить, что единственная разница между этими двумя – это ядро ввода.

Теперь необходимо объединить наши два значения, пользуясь тривиальной формулой (1):

$$G = \sqrt{G_{horizontal}^2 + G_{vertical}^2} \qquad (1)$$

В редакторе материалов также есть возможность создать вектор из рассчитанных значений и вернуть его длину, в результате чего будет получено черное изображение с белыми контурами вокруг объектов, для которых включена функция рендеринга глубины.

Последнее, что необходимо сделать для достижения результата, это добавить найденный контур в окончательное изображение. В этом случае я использовал выражение «IF», чтобы получить плавный обвод вокруг объектов [5]. Однако, при необходимости, можно довольно легко настроить эффект под конкретные нужды.

Настройка сцены

После создания материала его необходимо включить в перечень используемых для процесса постобработки сцены, чтобы увидеть получившиеся результаты.

На уровень необходимо добавить экземпляр класса PostProcessVolume, в котором отыскать параметр PostProccessMaterials и добавить в него новый элемент массива с созданным материалом.

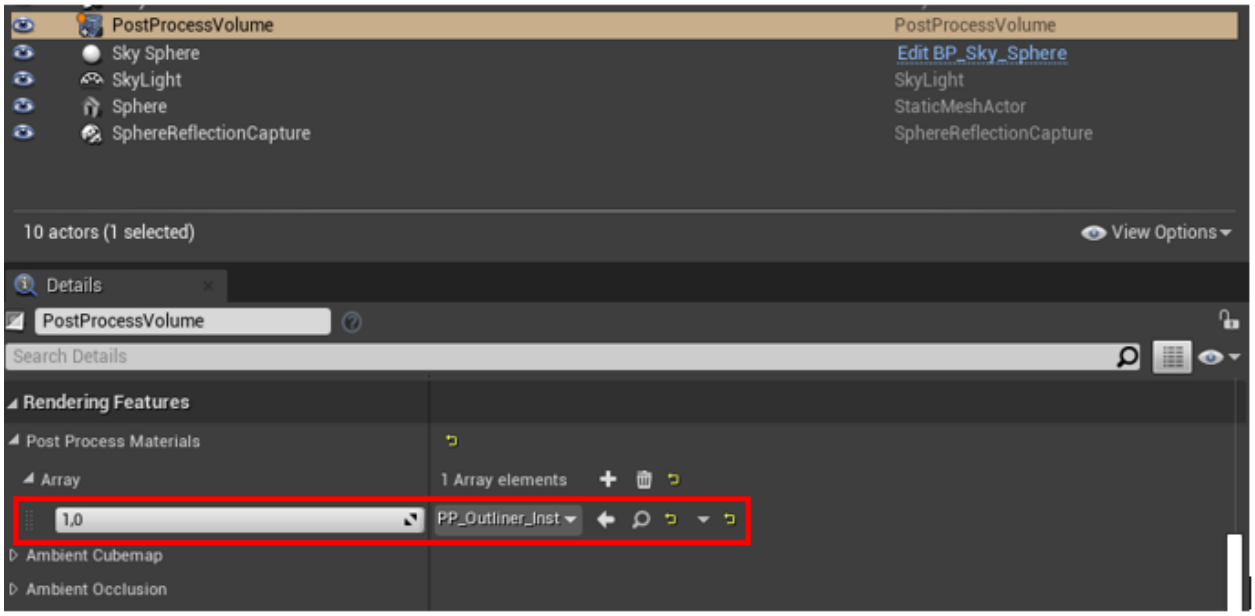


Рисунок 10. Выставление настроек класса постобработки [7].

Результат работы созданного материала продемонстрирован на рисунках ниже (рис.11.1 – 11.2).

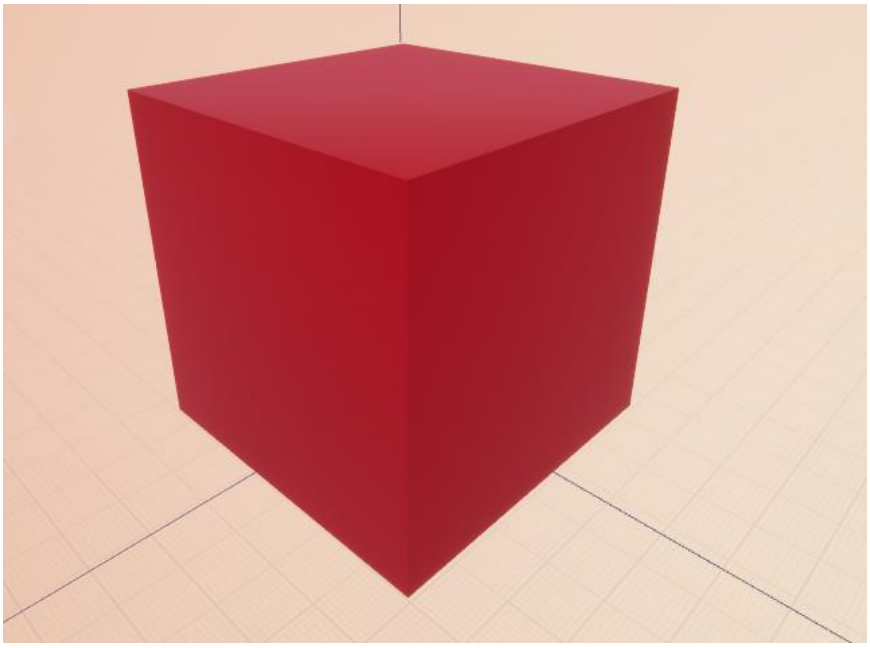


Рисунок 11.1. Объект без использования PostProccess материала [7].

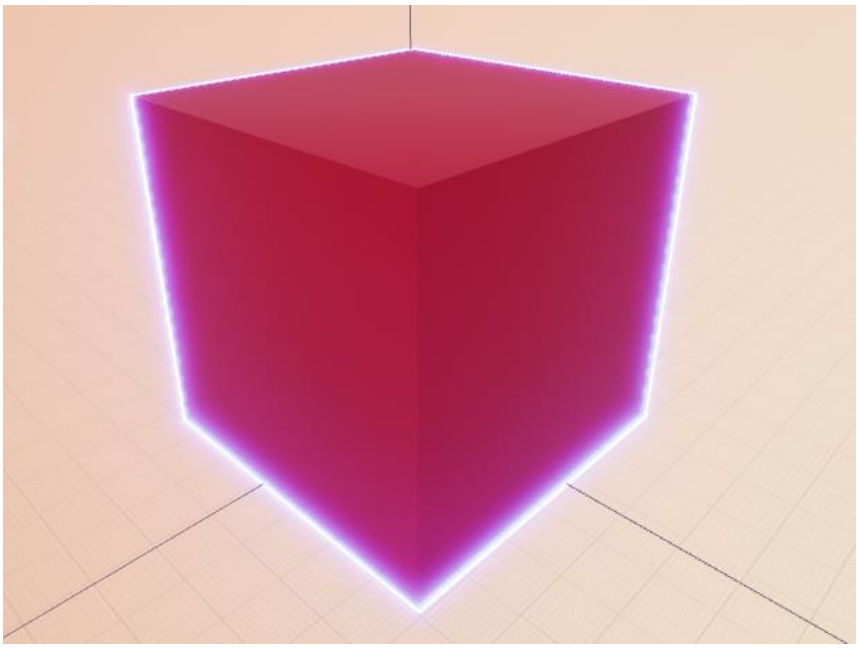


Рисунок 11.1. Объект с использованием PostProccess материала [7].

Вывод

В этой статье я представил довольно простой и легко интегрируемый метод рендеринга контуров вокруг объектов на Unreal Engine 4. Однако, созданный материал способен сделать гораздо больше – свернув изображение с разными ядрами, можно добиться различных эффектов: размытие, тиснение или резкость.

Литература

1. Оператор Собеля: статья на сайте [Электронный ресурс] URL: https://en.wikipedia.org/wiki/Sobel_operator (дата обращения: 20.07.20)
2. Алгоритмы выделения контуров изображений: статья на сайте [Электронный ресурс] URL: https://habr.com/ru/post/114452/ (дата обращения 20.07.20)
3. Кирсанов М.Н. Модификация и анализ фильтров выделения контуров изображений: статья на сайте [Электронный ресурс] URL: https://cyberleninka.ru/article/n/modifikatsiya-i-analiz-filtrov-vydeleniya-konturov-izobrazheniy/viewer (дата обращения 20.07.20)

4. Эффекты постобработки: Официальная документация графического движка Unreal Engine 4 [Электронный ресурс] URL: https://docs.unrealengine.com/en-US/Engine/Rendering/PostProcessEffects/index.html (дата обращения 20.07.20)
5. Работа с постобработкой в Unreal Engine 4: статья на сайте [Электронный ресурс] URL: https://habr.com/ru/post/350172/ (дата обращения 20.07.20)
6. Материал постобработки: Официальная документация графического движка Unreal Engine 4 [Электронный ресурс] URL: https://docs.unrealengine.com/en-US/Engine/Rendering/PostProcessEffects/PostProcessMaterials/index.html (дата обращения 20.07.20)
7. Черняев Р.В. Выпускная квалификационная работа на тему: «Разработка графического визуализатора для 3D моделей с поддержкой текстур высокого разрешения и PBR материалов на Unreal Engine 4» // МГТУ им. Н.Э.Баумана, кафедра РК6

Literature

1. Operator Sobel: article on the website [Electronic resource] URL: https://en.wikipedia.org/wiki/Sobel_operator (date accessed: 20.07.20)
2. Algorithms for highlighting the contours of images: article on the site [Electronic resource] URL: https://habr.com/ru/post/114452/ (date of access 20.07.20)
3. Kirsanov M.N. Modification and analysis of filters for highlighting image contours: article on the website [Electronic resource] URL: https://cyberleninka.ru/article/n/modifikatsiya-i-analiz-filtrov-vydeleniya-konturov-izobrazheniy/viewer (date of access 20.07.20)
4. Post-processing effects: Official documentation of the graphics engine Unreal Engine 4 [Electronic resource] URL: https://docs.unrealengine.com/en-US/Engine/Rendering/PostProcessEffects/index.html (date of access 20.07.20)

5. Working with post-processing in Unreal Engine 4: article on the website [Electronic resource] URL: https://habr.com/ru/post/350172/ (date of access 20.07.20)
6. Post-processing material: Official documentation of the graphics engine Unreal Engine 4 [Electronic resource] URL: https://docs.unrealengine.com/en-US/Engine/Rendering/PostProcessEffects/PostProcessMaterials/index.html (date of access 20.07.20)
7. Chernyaev R.V. Final qualification work on the topic: "Development of a graphical visualizer for 3D models with support for high-resolution textures and PBR materials on Unreal Engine 4" // Bauman MSTU, department RK6