ИПМ им.М.В.Келдыша РАН • Электронная библиотека

Препринты ИПМ • Препринт № 41 за 2025 г.

ISSN 2071-2898 (Print)
ISSN 2071-2901 (Online)

С.И. Куприянов, Д.Д. Жданов, И.В. Валиев

Метод восстановления текстуры равноярких объектов сцены

Рекомендуемая форма библиографической ссылки: Куприянов С.И., Жданов Д.Д., Валиев И.В. Метод восстановления текстуры равноярких объектов сцены // Препринты ИПМ им. М.В.Келдыша. 2025. № 41. 28 с. EDN: STBFF
https://library.keldysh.ru/preprint.asp?id=2025-41

О р д е н а Л е н и н а
ИНСТИТУТ ПРИКЛАДНОЙ МАТЕМАТИКИ
имени М.В. Келдыша
Р о с с и й с к о й а к а д е м и и н а у к

С.И. Куприянов, Д.Д. Жданов, И.В. Валиев

Метод восстановления текстуры равноярких объектов сцены

Москва — 2025

Куприянов С.И., Жданов Д.Д., Валиев И.В.

Метод восстановления текстуры равноярких объектов сцены

В статье представлен метод восстановления текстуры равноярких (диффузных) поверхностей объектов сцены. Данный метод предполагается использовать как элемент предобработки данных в задаче восстановления параметров модели оптических свойств средствами дифференцируемого рендеринга. Восстановление текстуры диффузных поверхностей сцены предполагает создание uv-развертки геометрии на плоскость, формирование текстурной карты яркостей с помощью проекции камер на плоскость текстуры и выделение компоненты диффузной модели оптических свойств по яркости и освещенности сцены. Созданная карта яркостей и оригинальные изображения в сочетании с параметрами геометрии и оптических свойств сцены позволяют рассчитать первичное освещение сцены и подобрать вторичное. На основе этих данных можно рассчитать цвет поверхности. Оптимизационный процесс запускает расчет вторичного освещения методом прямой трассировки лучей с сохранением освещенности и яркости на текстурных объектах. Вторичное освещение и яркость глобального освещения, полученные в процессе итерационных вычислений, сравниваются с исходной яркостью изображения, и представленный алгоритм корректирует цвет восстанавливаемой поверхности, переоценивая новое распределение вторичного освещения. Процесс повторяется до тех пор, пока отклонение яркости полученного изображения от яркости исходного изображения не окажется в заданных пределах. Предложенный подход требует наличия заранее определенной геометрии сцены, параметров источников света, а также оригинальных изображений для заданных параметров камеры. Восстановленный цвет текстуры объектов сцены позволяет использовать его в виде начальных данных параметров сцены в системе дифференцируемого рендеринга. Кроме того, найденная цветность объектов сцены позволяет упростить процесс дифференцируемого рендеринга за счет устранения сильной вариации яркости и уменьшения числа точек изображения сцены, передаваемых дифференцируемому рендерингу.

Ключевые слова: дифференцируемый рендеринг, оптические свойства, предобработка изображений, прямое освещение, вторичное освещение анализ геометрии, яркость, цвет, развертка геометрии, текстурирование, оптимизация

Kupriyanov S.I., Zhdanov D.D., Valiev I.V.

Eliminating brightness variation for the task of differentiable rendering

The article presents a method for restoring the texture of equiluminous (diffuse) surfaces of scene objects. This method is supposed to be used as an element of data preprocessing in the task of restoring the parameters of the optical properties model by means of differentiable rendering. Restoring the texture of diffuse surfaces of a scene involves creating a UV scan of the geometry onto a plane, forming a texture brightness map using the projection of cameras onto the texture plane, and selecting the components of a diffuse optical properties model based on the brightness and illumination of the scene. The created brightness map and original images, combined with the parameters of the geometry and optical properties of the scene, make it possible to calculate the primary lighting of the scene and select the secondary one. Based on this data, you can calculate the color of the surface. The optimization process starts calculating secondary illumination using direct ray tracing while maintaining illumination and brightness on textured objects. The secondary illumination and global illumination brightness obtained during iterative calculations are compared with the initial brightness of the image, and the presented algorithm corrects the color of the restored surface and re-evaluates the new distribution of secondary illumination. The process is repeated until the brightness deviation of the received image from the brightness of the original image is within the specified limits. The proposed approach requires a pre-defined geometry of the scene, the parameters of the light sources, as well as the original images for the specified camera parameters. The restored color of the texture of the scene objects allows it to be used as the initial data of the scene parameters in the differentiable rendering system. In addition, the found chromaticity of scene objects makes it possible to simplify the process of differentiable rendering by eliminating strong brightness variations and reducing the number of points in the scene image transmitted to differentiable rendering.

Keywords: differentiable rendering, optical properties, image preprocessing, direct illumination, secondary illumination geometry analysis, brightness, color, geometry scanning, texturing, optimization

Оглавление

Введение

Дифференцируемый рендеринг представляет собой метод, обеспечивающий математически формализованное вычисление градиентов целевых функций по параметрам трехмерной сцены, включая ее геометрическое представление, оптические характеристики материалов и конфигурацию источников освещения. В отличие от классического подхода к синтезу изображений, трактуемого как детерминированный процесс с дискретным отображением пространственных данных в точку изображения, данный метод устанавливает дифференцируемое соответствие между параметрами сцены и цветовыми значениями результирующего изображения.

С практической точки зрения дифференцируемый рендеринг позволяет рассчитывать градиенты по параметрам сцены и исходным изображениям, что создает предпосылки для применения методов оптимизации на основе градиентного спуска в задачах автоматизированной настройки свойств сцены по целевым метрикам. Такой подход помогает, например, восстанавливать геометрию объектов, а также модель оптических свойств, приближенных к исходным данным сцены.

Для восстановления параметров модели оптических свойств сцены дифференцируемому рендерингу необходимо получить дополнительные данные о сцене. К ним относятся: геометрия объектов сцены, изображения сцены, информация о камерах, для которых были получены эти изображения, данные источников света, и цвета объектов сцены. Необходимо отметить, что все дополнительные данные могут быть восстановлены средствами дифференцируемого рендеринга.

В основе текущей работы лежит идея о возможности уменьшения объема данных для дифференцируемого рендеринга. В работе [1] было показано, что если целью восстановления являются коэффициенты модели оптических свойств, то можно ограничиться небольшим набором лучей, направленных в наиболее информативные точки сцены. Такой подход сокращает набор данных для расчета и тем самым ускоряет процесс определения параметров модели оптических свойств. Однако при таком подходе цвет поверхности объекта сцены должен быть постоянным. В общем случае при восстановлении цвета объекта необходимо производить расчет по всей его поверхности из-за возможной высокой вариации цвета на неоднородных поверхностях. Поэтому для того чтобы получить возможность выполнять дифференцируемый рендеринг на ограниченном наборе точек неоднородного (по цвету) объекта, необходимо сперва восстановить его цветовую текстуру. Данная работа посвящена построению цветовой карты (текстуры) объектов сцены, цветность которой может быть использована для устранения значительной вариации цвета на исходных изображениях сцены и формированию ее «одноцветных» изображений.

Обзор литературы

Для устранения вариации цвета необходимо выполнить предварительный анализ геометрии и оригинальных изображений сцены. В результате анализа геометрии формируется ее uv-развертка на плоскость. Данная развертка необходима для построения карты яркостей, расчета цвета объектов сцены, а также для инициализации цвета объектов при рендеринге сцены.

UV-развертка представляет собой процесс проекции трехмерной поверхности на плоскость для построения текстуры. Данный подход является одним из фундаментальных подходов компьютерной графики, поскольку позволяет установить соответствие между вершинами трехмерной модели и координатами текстуры (u, v) [2]. Математически uv-развертка может быть описана следующим образом: каждая точка $p \in M$ (где M — многообразие, представляющее 3D-объект) отображается в точку $(u, v) \in R^2$, сохраняя локальную геометрическую информацию [3].

Параметризация поверхности предполагает нахождение биективного отображения $\phi : M \to \Omega \subset R^2$, минимизирующего искажения. При биективном отображении каждому элементу одного множества соответствует ровно один элемент другого множества, при этом определено обратное отображение, которое обладает тем же свойством. Для треугольной сетки задача сводится к определению таких координат (u_i, v_i) для каждой вершины i, чтобы деформация у развертки была минимальной. Одним из ключевых подходов является изометрическая параметризация, сохраняющая углы и длины, однако в общем случае полная изометрия невозможна для поверхностей с ненулевой гауссовой кривизной [4]. Также используют метод наименьших квадратов (Least Squares Conformal Maps, LSCM), который решает задачу минимизации конформного искажения через оптимизацию функционала. Альтернативный подход — Spectral Conformal Parameterization (SCP) — использует спектральный анализ лапласиана сетки для построения глобально оптимального отображения [6].

Классические алгоритмы uv-развертки включают:

1. Методы на основе граничных условий

Методы, использующие фиксацию границ на выпуклом многоугольнике. Это упрощают параметризацию за счет снижения вычислительной сложности [3]. Например, привязка граничных вершин к выпуклому контуру гарантирует однозначность отображения и предотвращает самопересечения [2]. Однако такие методы могут вызывать искажения внутри области из-за неравномерного распределения деформаций, особенно для поверхностей со значительной кривизной [4].

2. Глобальные методы: Angle-Based Flattening (ABF)

Метод ABF минимизирует угловые деформации через решение системы нелинейных уравнений, сохраняя локальные углы между треугольниками [5].

3. Многосеточные подходы

Многосеточные методы адаптируют разрешение, комбинируя грубые и детальные сетки [7]. Например, на уровне низкого разрешения решается задача с аппроксимированной геометрией, после чего результаты уточняются на высоко детализированных уровнях [8]. Это позволяет снизить сложность алгоритмов с $O(N^3)$ до $O(N \log N)$, где N — число вершин. Такие методы эффективны для сложных поверхностей, таких как органические формы или анатомические структуры [9].

4. Современные методы: машинное обучение

Нейросетевые архитектуры, такие как MeshVAE, обучаются на датасетах 3D-моделей для предсказания uv-координат [10]. Эти модели минимизируют зависимость от ручной настройки, автоматизируя развертку для неструктурированных сеток. Например, генеративно-состязательные сети (GAN) используются для создания текстурных карт, учитывающих геометрические особенности объектов.

Для построения развертки существуют готовые библиотеки, например xatlas. Библиотека проста в реализации и применении. Алгоритм работает в два этапа:

Сегментация поверхности — разделение сетки на связные области (чарты) с минимизацией метрики угловых искажений. Для этого используются методы спектрального анализа, дополненные условиями сохранения локальной геометрической согласованности.

Оптимизация uv-размещения — упаковка чартов в текстурное пространство с применением итеративного алгоритма, снижающего перекрытия и растяжения за счет решения систем уравнений, учитывающих тензорные деформации на проекции.

Основным преимуществом xatlas является способность генерировать uv-развертки для высокодетализированных сеток с сохранением топологии, включая модели с нерегулярной триангуляцией и геометрическими аномалиями. Библиотека демонстрирует высокую устойчивость к артефактам, что критично для последующего запекания карт яркостей в HDR-представлении. Открытая архитектура и поддержка кроссплатформенности упрощают интеграцию в конвейеры обработки данных.

К ограничениям относится склонность к фрагментации чартов на моделях с множеством тонких элементов (например, ветвистые или волокнистые структуры), что увеличивает сложность постобработки текстур. Кроме того, отсутствие параллельных вычислений ограничивает производительность при работе с экстремально плотными сетками (более 1 млн полигонов).

После построения развертки геометрии необходимо извлечь цвет (или текстуру) по набору изображений. Выделение цветовых характеристик и текстурных признаков из изображений является фундаментальной задачей в компьютерном зрении. Эти методы лежат в основе сегментации объектов, классификации изображений и анализа сцен.

При наличии известной геометрической развертки (uv-развертки) объекта задача построения текстурной карты сводится к проецированию информации с множества изображений, снятых под разными углами, на соответствующую параметризованную поверхность. Этот процесс требует решения задач согласования данных, коррекции освещения и устранения артефактов.

Классический метод предполагает проецирование точек каждого из изображений на uv-карту с использованием матриц камеры и данных о геометрии. Для пикселя (u, v) на текстуре значение определяется как средневзвешенное по всем изображениям, которые он видит:

$$T(u,v) = \frac{1}{N}\sum_{i=1}^{N} w_i \cdot I_i(x_i, y_i),$$

где $I_i(x_i, y_i)$ — яркость пикселя в i-м изображении, w_i — веса, учитывающие угол между нормалью поверхности и лучом камеры [11]. Для устранения шума применяются фильтры (например, медианный), а для HDR-текстур используется линейное сочетание экспозиций [12].

Однако изображения содержат элементы освещения сцены и для дальнейшего использования текстур необходимо убрать из яркости текстур «влияние» источников света.

Глубокое обучение позволяет автоматизировать синтез текстур. Архитектуры типа TextureGAN [13] генерируют детализированные uv-карты, обучаясь на парных данных (изображения и развертки). Трехмерные сверточные сети (3D-CNN) агрегируют информацию из множества ракурсов, минимизируя рассогласование между проекциями [14].

Помимо этого, можно отметить работу «NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis» [15], в которой показан алгоритм дифференцируемого рендеринга с использованием нейронных сетей для восстановления параметров сцены. Хотя NeRF изначально фокусируется на реконструкции сцен, его расширения, такие как NeRF-Tex [16], позволяют извлекать текстуры через неявное представление поверхности. Нейросеть предсказывает цвет и плотность в 3D-точках, а затем проецирует эти данные на uv-карту. Обучение на PyTorch с использованием позиционного кодирования обеспечивает сохранение высокочастотных деталей, но метод требует значительных вычислительных ресурсов.

В заключение можно сказать, что существует множество способов выполнения развертки геометрии на плоскость и проецирования текстур на эту развертку. Однако данные алгоритмы можно расширить новыми решениями, которые анализируют освещение сцены и с его помощью выделяют истинные значения цвета точек текстуры поверхности. Такой подход позволит получить текстуры, идентичные используемым при создании изображений на рассматриваемой сцене.

Результаты

В задачах дифференцируемого рендеринга, направленных на восстановление параметров оптических свойств материалов, важным этапом предобработки является определение цветовых характеристик поверхностей. Эти характеристики могут задаваться как глобальными параметрами для всего объекта, так и посредством текстурных карт, кодирующих не только цвет, но и локальные оптические свойства (например, коэффициент диффузного отражения kd). В рамках текущего исследования рассматривается подход, предполагающий реконструкцию текстур для диффузных поверхностей на этапе предобработки сцены. Для восстановления цветовых значений предлагается построить развертку геометрических объектов сцены на плоскость и собрать карты яркостей с HDR изображений. Далее можно использовать развертку для составления всех текстурных данных сцены. Идея исследования заключается в том, что карта яркостей и развертка позволяют составить карту первичного и вторичного освещения, что дает возможность оценить цветовую компоненту объекта сцены.

Развертка геометрии

Для генерации текстурных карт яркостей требуется построить uv-развертку трехмерной геометрии, обеспечивающую однозначное соответствие между точками поверхности и координатами текстуры. Несмотря на то, что геометрия объектов известна, автоматизация этого процесса для сложных моделей остается нетривиальной задачей. Для построения развертки использовалась библиотека xatlas.

Помимо развертки необходимо выбрать разрешение текстуры. Разрешение можно задать вручную, но его также можно сделать зависимым от положения и разрешения камер. Такой подход позволяет сохранить максимум деталей текстуры при формировании яркости по изображениям. В данном исследовании реализован адаптивный метод выбора разрешения.

Размер текстуры адаптивно рассчитывается на основе площади поверхности 3D-объекта и максимальной плотности точек камер. Сначала вычисляется площадь треугольников и определяется центр ограничивающего его объема. Для каждой камеры определяется расстояние до этого центра, и на основе его (с учетом поля зрения, разрешения и дистанции) рассчитывается размер проекции одной точки на объект. Плотность точек (количество на единицу площади) обратно пропорциональна квадрату этого размера.

Итоговое разрешение текстуры определяется как квадратный корень из произведения площади поверхности на максимальную плотность, что дает возможность сохранить максимальную детализацию для всех ракурсов. Для баланса между объемом данных и производительностью вычислений разрешение ограничивается диапазоном 256–4096 точек. Такой подход обеспечивает адаптацию текстуры к разрешению камер и минимизирует

вычислительные затраты. Пример развертки геометрии и ее развертки представлен на рис. 1.

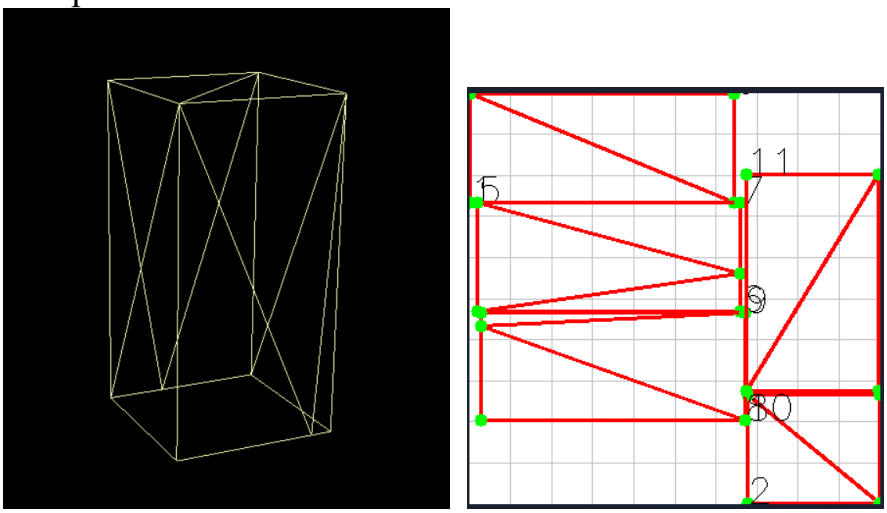


Рис. 1. Пример геометрии (слева) и ее развертка (справа)

Видно, что xatlas «разобрала» куб на отдельные плоскости и спроецировала их на текстурную карту.

Заполнение карты яркостей

После получения развертки начинается процесс заполнения текстурной карты. Сперва формируются карты яркостей, которые необходимы для восстановления цветовых текстур. На первом этапе карты яркости заполняются по оригинальным изображениям. Для каждой точки текстуры вычисляются ее мировые координаты и формируются лучи из этих точек на камеру. Если между точкой и камерой нет другого объекта сцены и точка находится в области видимости камеры, то цвет текстуры заполняется значением соответствующей точки изображения. Процесс повторяется для всех камер с усреднением цвета текстуры от каждой камеры. Таким образом формируется цветовая карта яркостей текстур объектов сцены.

Предполагается, что изображения для рассматриваемого объекта сцены созданы со всех возможных ракурсов камеры и в них нет пропущенных участков. Однако в случае затенения объектов или недостаточного числа ракурсов на текстуре могут быть незаполненные области. Эти области должны быть заполнены, чтобы в процессе последующего рендеринга не получить неопределенную яркость. Как правило, для восстановления текстур в незаполненных областях применяют нейронные сети. В предложенном методе реализован более простой способ заполнения этих областей, основанный на использовании библиотеки OpenCV.

Следует отметить, что для заполнения пропусков на текстуре необходимо учитывать особые ситуации, связанные, например, с антиалиасингом, выполняющем усреднение по области пикселя. На левой части рис. 2 изображена карта яркостей потолка сцены cornel box.

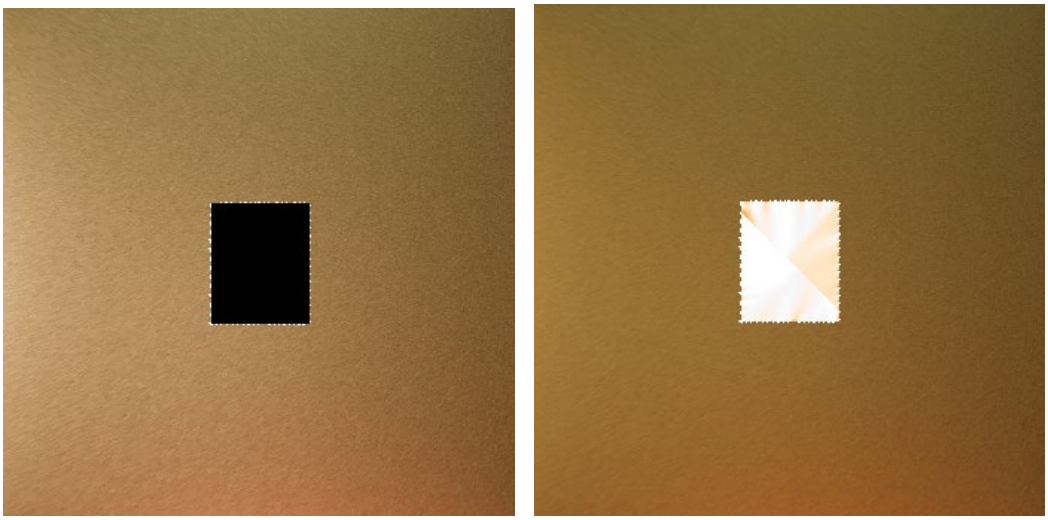


Рис. 2. Карта яркостей потолка сцены (слева)
и результат интерполяции (справа)

На данном примере в центре потолка располагался прямоугольный источник света. Вследствие антиалиасинга исходного изображения граничная область вблизи источника света обладает высокой яркостью. Интерполяция, используемая в OpenCV, формирует завышенную яркость данной области, что приводит к результату, показанному на правой части рис. 2.

Видно, что яркость незаполненной области оказалась переоцененной из-за яркой рамки, оставшейся от источника света. Для того чтобы устранить ложное заполнение для объектов вблизи сверх ярких источников света, можно использовать медианный фильтр. Поскольку медианный фильтр может испортить общую яркостью объекта сцены, он применялся только для объектов сцены, расположенных близи ярких источников света.

Пример использования медианного фильтра и заполнения яркости пропущенной области объекта после работы медианного фильтра представлен на рис. 3.

Рис. 3. Карта яркостей после применения медианного фильтра (слева) и восстановленный пропуск на карте яркостей после обработки медианным фильтром (справа)

Приведенный выше алгоритм используется для формирования карты яркостей и восстановления пропущенных областей, которые не видит ни одна из камер. После заполнения всех пропущенных участков карта яркостей и развертка геометрии передаются алгоритму восстановления цвета текстурной карты объектов сцены.

Выделение цвета текстурной карты объекта

Анализируя изображения и освещение в каждой точке текстуры, можно выделить значения ее цвета и сформировать текстурную карту. Алгоритм определения цвета текстуры диффузной поверхности математически можно вывести через определение яркости. Яркость в точке p, которая имеет компоненту цвета c, вычисляется как

$$L(p,c) = \frac{1}{\pi} Kd(p,c) \cdot \left(\sum Ed_j(p,c) + Ei(p,c)\right),$$

где $Kd(p,c)$ — коэффициент яркости в точке p для компоненты цвета c (текстура), $\sum Ed_j(p,c)$ — суммарная прямая освещенность в точке p от всех источников света, имеющих компоненту цвета c, $Ei(p,c)$ — вторичная освещенность в точке p, имеющая компоненту цвета c, $\cdot$ — операция покомпонентного умножения цвета. Далее можно вывести выражения для цвета текстуры как

$$Kd(p,c) = \pi \frac{L(p,c)}{\sum Ed_j(p,c) + Ei(p,c)}.$$

Если прямого освещения нет, то выражение принимает следующий вид:

$$Kd(p,c) = \pi \frac{L(p,c)}{Ei(p,c)}.$$

$\sum Ed_j(p,c)$ может быть вычислена, так как в соответствие с условиями работы дифференцируемого рендеринга имеется информация обо всех источниках света и геометрии сцены, следовательно, в каждой точке текстуры можно вычислить прямое освещение. Таким образом, для вычисления цвета текстуры доступны все параметры за исключением вторичного освещения $Ei(p,c)$. Кроме того, известно, что коэффициент диффузного отражения должен находиться в пределах от 0 до 1:

$$0 \le Kd(p,c) \le 1.$$

Таким образом, для получения значений цвета объекта сцены необходимо подобрать вторичное освещение так, чтобы в результате яркость рендеринга совпала с яркостью исходных изображений. Исходя из этого можно оценить значение $Kd(p,c)$ в каждой точке изображения (текстуре). Для правильной оценки текстурных значений был построен алгоритм восстановления вторичного освещения.

Общий алгоритм восстановления цвета текстуры имеет следующий вид. Сперва создается развертка и заполняется карта яркостей объектов сцены. Далее для каждой точки текстуры рассчитывается прямое освещение и

происходит приближенная оценка вторичного освещения. На основе этих данных рассчитывается текстурная карта цвета поверхности. Эти данные служат основой для процесса восстановления. По рассчитанному цвету производится рендеринг, который уточняет вторичное освещение на реальной сцене. Также в процессе рендеринга собирается яркость и сравнивается с яркостью, собранной ранее по изображениям. Впоследствии данная оценка используется для коррекции вторичного освещения. Процесс итеративный и останавливается по достижении заданного числа итераций или порога ошибки. Схема алгоритма представлена на рис. 4.

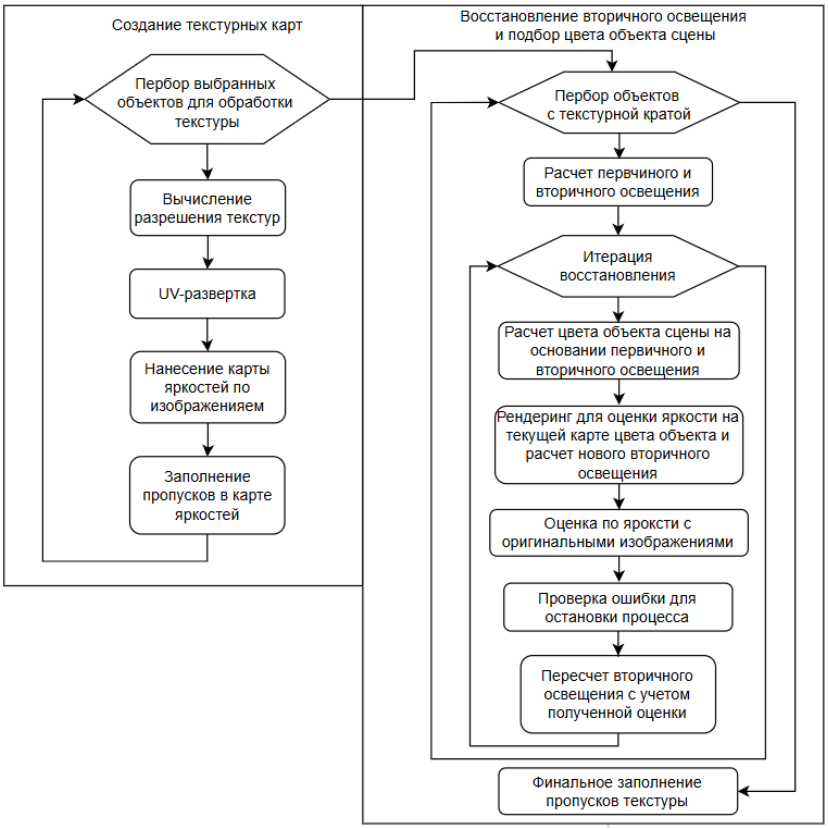


Рис. 4. Алгоритм восстановления цвета объектов сцены

Более детально алгоритм можно описать следующим образом. На первом этапе формируется текстурная карта цвета объекта, по которой будет производиться рендеринг. Для этого на uv-развертке вычисляется прямое освещение с помощью механизма выборки точек на источнике света. Цвет объектов сцены рассчитывается на основе яркости, прямого и вторичного освещений. Однако на первом шаге нельзя корректно рассчитать вторичное освещения без примерной оценки цвета поверхности. Поэтому в качестве первичной оценки вторичного освещения выбирается определенный процент от первичного освещения, например 50%. Таким образом можно получить все данные для предварительного расчета цвета объекта по карте яркости и освещения. Кроме того, необходимо убедиться, что на карте цвета нет пропусков из-за отсутствия данных о вторичном и первичном освещении.

Если пропуски имеются, то они восстанавливаются средствами билинейной интерполяции по соседним точкам.

При вычислении цвета объекта проверяется условие, что ни одна из компонент цвета не вышла за границы от 0 до 1. Если условие не выполняется, то увеличивается значение вторичной освещенности до допустимых значений компонент цвета.

Таким образом создаются начальные условия для восстановления цвета объектов сцены. На следующем шаге происходит пересчет цвета по результатам расчета вторичного освещения, полученного на начальном шаге и скорректированного по ошибке между полученной яркостью и исходной яркостью изображения. Для расчета вторичного освещения был применен метод трассировки световых лучей с накоплением освещенности на текстурных картах. Данный алгоритм реализует гибридный подход, сочетающий стохастический метод трассировки световых лучей для моделирования вторичного освещения с текстурным картографированием для пространственного накопления и представления результатов на параметризованных uv-координатах текстуры геометрии сцены. Для расчета прямой освещенности используется стандартный алгоритм выборки по значимости точек на источниках света в соответствии с их светимостью.

Предложенный подход дает правильную оценку вторичного освещения, поскольку он выполнялся на реальной сцене для заданных параметров цвета поверхности. Однако результирующая яркость может отличаться от исходной, и ее необходимо скорректировать в соответствии со значением этого различия. В проводимом исследовании в качестве оценки отличия используется разница яркости исходных изображений и полученной в результате рендеринга: $score = L_{orig} - L_{render}$. Ошибки вычисляются для всех пикселей текстуры и заносятся в текстурную карту.

Для устранения выбросов оценки яркости, возникающих, например, на границах ярких объектов происходит обработка карты яркостей с помощью квартилей, которая позволяет убрать эти выбросы. Далее полученная ошибка используется для коррекции вторичного освещения.

Используемый алгоритм коррекции заключается в расчете градиента яркости. Как уже упоминалось ранее, яркость для равноярких поверхностей рассчитывается по формуле $L_{compute} = \frac{C_{text}}{\pi} \cdot (E_d + E_i)$, где E_d — первичное освещение, E_i — вторичное, а c_{text} — значение цвета в точке рассчитанной текстурной карты поверхности. Ошибка яркости вычисляется как разность между наблюдаемой и вычисленной яркостями: $error = L_{orig} - L_{compute}$. Поскольку цель — скорректировать E_i так, чтобы минимизировать error, то можно определить соответствующий градиент.

Рассмотрим ошибку для одного канала RGB ($j \in \{R, G, B\}$):

$$error_j = L_{orig_j} - L_{compute_j} = L_{orig_j} - \frac{C_{text_j}}{\pi} \cdot \left(E_{d_j} + E_{i_j}\right).$$

Вычислим частную производную $error_j$ по E_i:

$$\frac{\partial error_j}{\partial E_{i_j}} = \frac{\partial}{\partial E_{i_j}}\left(L_{orig_j} - \frac{C_{text_j}}{\pi} \cdot \left(E_{d_j} + E_{i_j}\right)\right).$$

Поскольку C_{text}, L_{orig_j} и E_{d_j} не зависят от E_{i_j}, производная затрагивает только второй член:

$$\frac{\partial}{\partial E_{i_j}}\left(-\frac{C_{text_j}}{\pi} \cdot \left(E_{d_j} + E_{i_j}\right)\right) = -\frac{C_{text_j}}{\pi} \cdot \frac{\partial}{\partial E_{i_j}}\left(E_{d_j} + E_{i_j}\right).$$

Так как $\frac{\partial}{\partial E_{i_j}}\left(E_{d_j} + E_{i_j}\right) = 1$, получаем $\frac{\partial error_j}{\partial E_{i_j}} = -\frac{C_{text_j}}{\pi}$, таким образом градиент для рассматриваемой точки текстуры можно представить следующим образом:

$$grad_{text_j} = -\frac{C_{text_j}}{\pi}.$$

Изменение вторичного освещения осуществляется следующим образом:

$$E_{i_{j_{mod}}} = \alpha \cdot grad_{text_j} \cdot error_j,$$

где α — шаг оптимизации, определяющий скорость коррекции.

Для обеспечения стабильности сходимости и предотвращения колебаний при большом числе итераций α можно сделать адаптивной. Для этого шаг выбирается с учетом номера итераций:

$$\alpha_k = \frac{\alpha_0}{1 + k},$$

где α_0 — начальный шаг. Такой подход позволяет быстро корректировать большие ошибки на ранних итерациях и точно подстраивать параметры на поздних стадиях.

После вычисления изменений $E_{i_{mod}}$ значение вторичного освещения обновляются:

$$E_{i_{j_{new}}} = E_{i_j} - E_{i_{j_{mod}}}.$$

Изменение вторичного освещения происходит индивидуально для каждого цветового канала. Таким образом, когда ошибка положительная, освещение увеличивается, а когда отрицательная — уменьшается, что соответствует логике вариации вторичного освещения. Кроме того, для обеспечения физической корректности применяется ограничение на неотрицательность вторичного освещения:

$$E_{i_{j_{new}}} = \max\left(0, E_{i_{j_{new}}}\right).$$

После расчета вторичного освещения происходит пересчет цвета текстуры с соответствующей проверкой на диапазон от 0 до 1. В случае нарушения данного условия применяется алгоритм, описанный выше.

Тестирование на однородной поверхности

Для демонстрации работы алгоритма было создано несколько вариантов сцены cornel box с разными оптическими свойствами поверхностей и для них

было проведено восстановление оптических свойств. Первый вариант сцены — это классическая сцена cornel box. На рис. 5 показаны изображения сцены, для которых производилось восстановление оптических свойств.

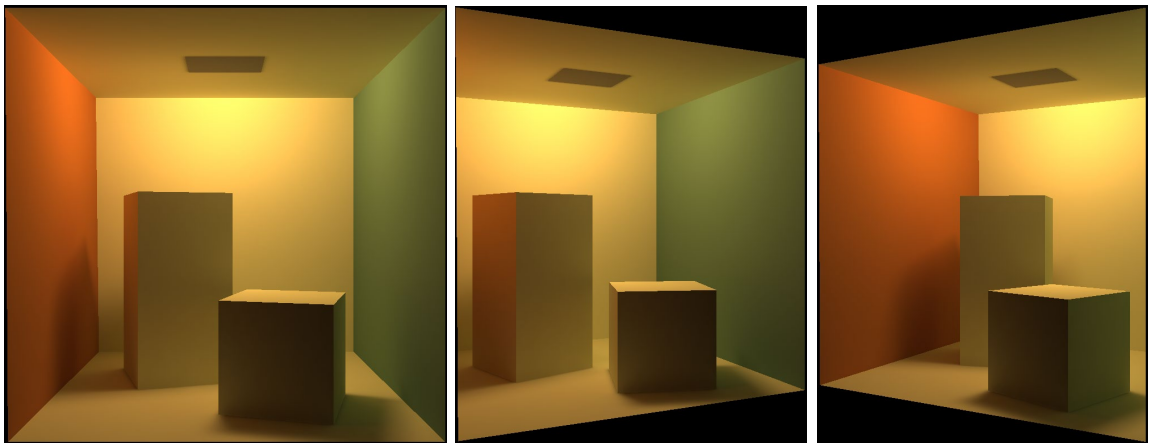


Рис. 5. Синтетическая сцена cornel box под разными ракурсами

Сцена создавалась методом прямой трассировки лучей на текстурных картах, поэтому источник света содержит только вторичное освещение, что делает его темным.

Сначала формируется развертка геометрии. На рис. 6 представлена геометрия зеленой стенки (слева), текстурная развертка этой поверхности на плоскость (по центру) и исходная карта яркостей для этой стенки (справа).

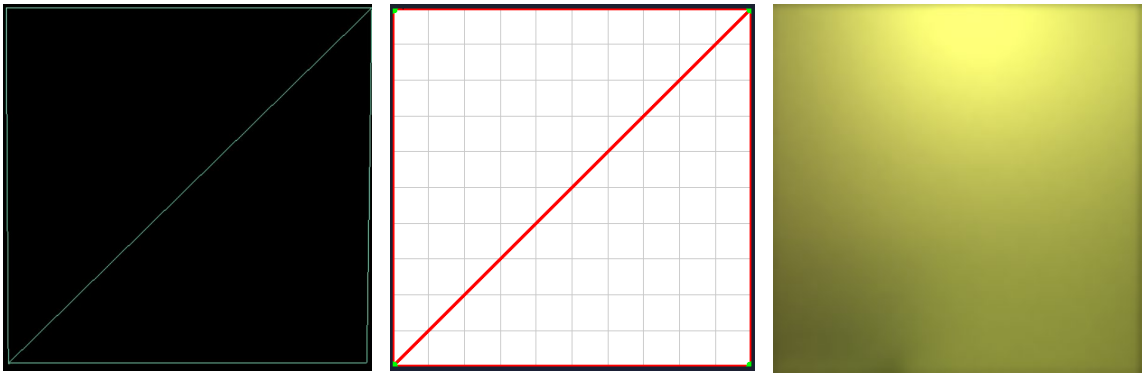


Рис. 6. Геометрия правой (зеленой) стенки сцены (слева), текстурная развертка (по центру) и исходная карта яркости (справа)

На исходной карте яркости видно явно выраженное влияние источника света. Предложенный метод восстановления цвета поверхности должен убрать вариацию яркости и сделать цвет более однородным. На рис. 7 показана карта прямого освещения этой стены (слева). После формирования карты прямого освещения рассчитывается вторичное освещение как некоторый процент от первичного освещения (центральная часть рис. 7). Далее запускается рендеринг и вычисляется яркость изображения. По вычисленной яркости изображения формируется карта ошибок, которая представляет из себя разницу оригинальной яркости и полученной в процессе рендеринга. Данная карта представлена на правой части рис. 7.

Рис. 7. Карта прямого освещения правой зеленой стенки (слева),
карта начального цвета поверхности (по центру)
и ошибка вычисления цвета (справа)

Для большей наглядности карта ошибок приводится в относительном представлении: $error_{normilized} = \frac{error}{L_{orig}} * 100$, где L_{orig} — это яркость исходных изображений. На рис. 8 представлены результаты четырех шагов процесса восстановления цвета объектов сцены. Левый верхний рисунок — результат начального выбора цвета. Он дает ошибку более 100%, однако после первой итерации максимальная ошибка уже снизилась до 3.5% (правый верхний рисунок). После третьей итерации максимальная ошибка уже стабилизировалась (третья итерация — левый нижний рисунок, четвертая итерация — правый нижний рисунок) и приблизилась к 1.8%. Данная точность является приемлемой и дальнейшие итерации уже не дают существенного изменения точности.

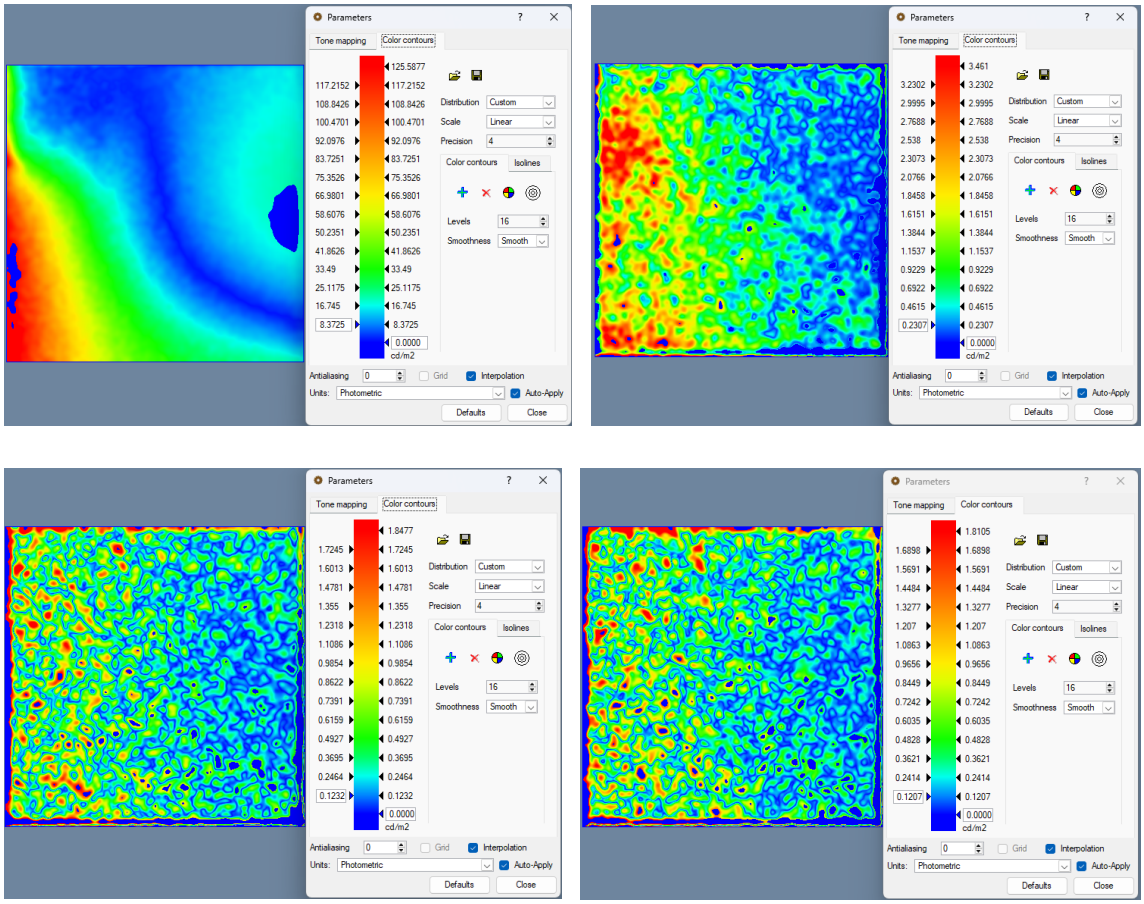


Рис. 8. Карты нормализованной ошибки: на первой итерации (верхний левый рисунок), на второй итерации (верхний правый рисунок), на третьей итерации (нижний левый рисунок), на четвертой итерации (нижний правый рисунок)

В процессе итераций ошибка уменьшилась, что подтверждает рациональность примененного алгоритма. Тестирование производилось с $\alpha_0 = 2$ в адаптивном варианте, зависящем от номера итерации. В большинстве случаев требовалось не более 5 итераций. При увеличении числа итераций точность практически не поднимается. На рис. 9 показана точность 20-й итерации (слева) и цвет поверхности (справа), полученный в результате восстановления оптических свойств данным алгоритмом.

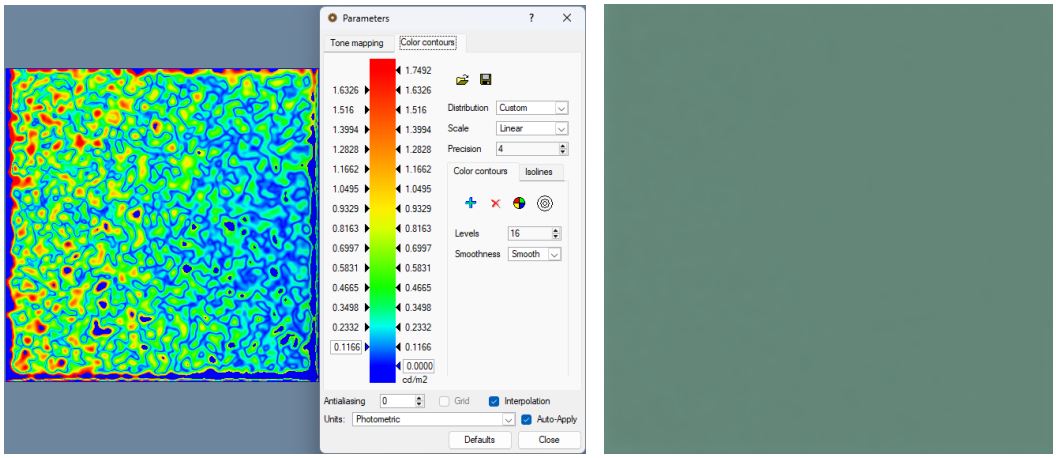


Рис. 9. Нормализованная ошибка на 20-й итерации (слева) и цвет поверхности после 20 итераций (справа)

Видно, что максимальная ошибка уменьшилась, но не стала нулевой. Причиной этого может служить недостаточное число лучей, используемых для расчета вторичного освещения. В работе использовалось 100 миллионов световых лучей, что достаточно для получения корректного среднего результата, однако может быть недостаточно для того, чтобы «осветить» края текстуры. С другой стороны, ошибка 1.8% вполне приемлема и соизмерима с точностью работы большинства измерительных устройств. Поэтому для обеспечения баланса между скоростью и точностью работы расчет вторичного освещения проводился на 100 миллионах световых лучей.

Поскольку восстановление проводилось на объекте с однородным цветом, можно оценить скорость и ошибку сходимости цвета к истинному значению. На рис. 10 показан график сходимости среднего значения цвета объекта (слева), средняя ошибка в абсолютных значениях (по центру) и относительная ошибка (справа).

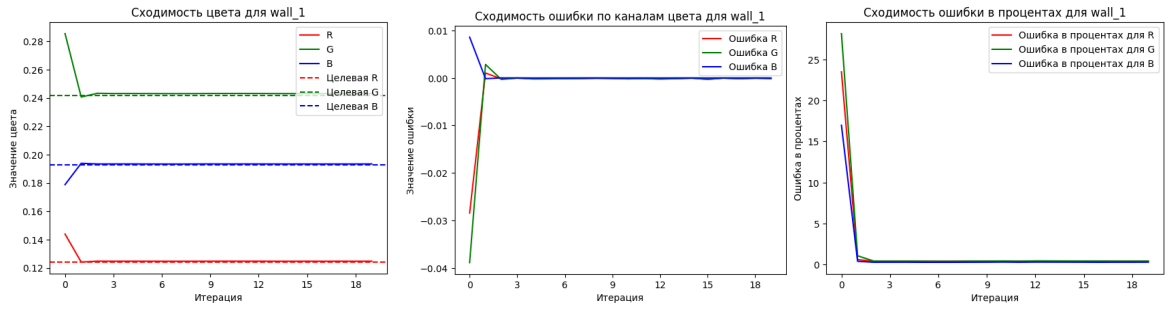


Рис. 10. График сходимости среднего значения цвета к истинному значению цвета правой зеленой стенки (слева), абсолютная ошибка (по центру) и относительная ошибка (справа)

На графиках видно, что цвет сходится к истинному значению, а ошибка стремится к 0. Это свидетельствует о корректной работе алгоритма. Видно, что для восстановления одного объекта сцены алгоритм находит приемлемый результат за 3-4 итерации. Также можно выполнить рендеринг и сравнить

изображение с оригиналом. На рис. 11 показан результат сравнения оригинального изображения с результатом рендеринга с восстановленными свойствами.

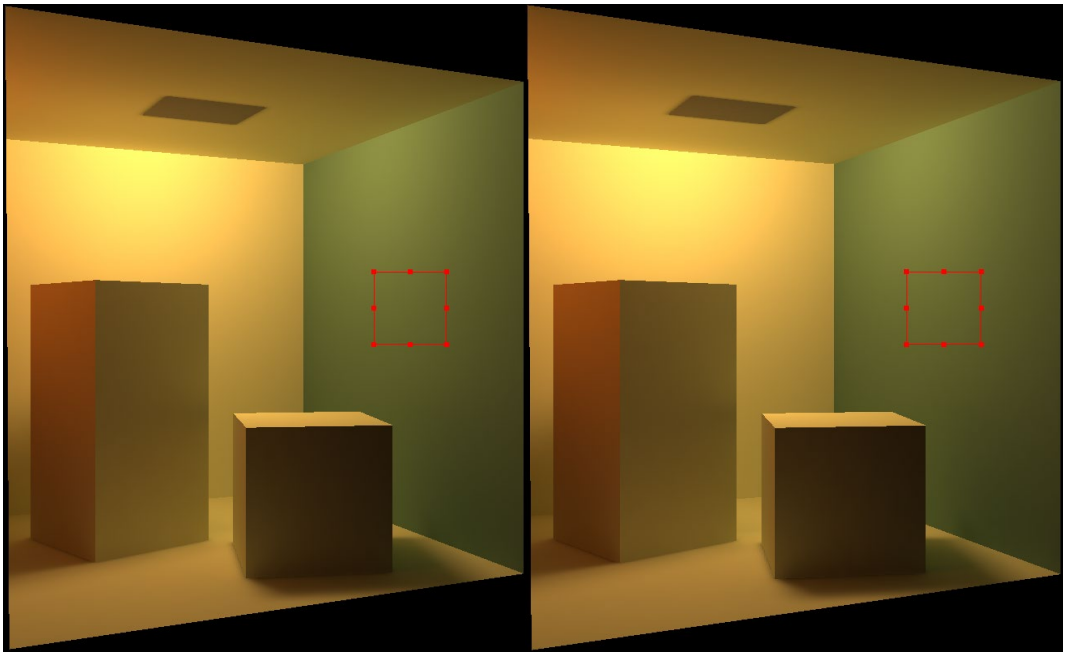
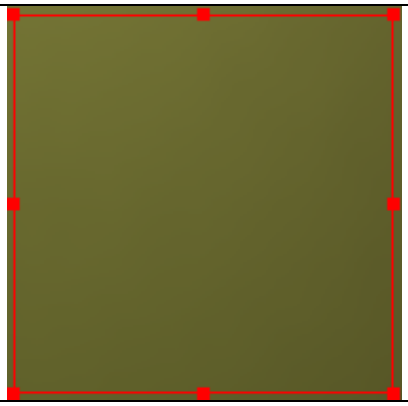
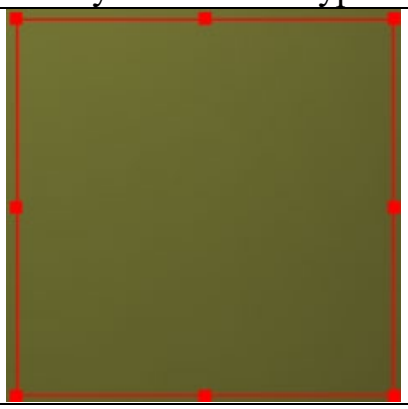


Рис. 11. Оригинальное изображения (слева), результат рендеринга, полученный после восстановления цвета (справа)

Визуально результаты практически неразличимы. Таблица 1 демонстрирует данные для выделенной области зеленой стенки.

Таблица 1. Сравнение средних значений яркости и цвета для области 90 × 90 пикселей на оригинальном изображении и изображении, полученном с помощью рендеринга с восстановленными оптическими свойствами зеленой стенки

	Оригинал	Рендеринг на полученной текстуре
Изображение		
Средняя яркость (cd/m2)	0.392047	0.391723
Среднеквадратичное отклонение яркости	0.0513821	0.0516017

(cd/m2)		
Цвет RGB (cd/m2)	0.415489, 0.417419, 0.0716505	0.415249, 0.417045, 0.0715795
Цветность xy (CIE)	0.391717, 0.460432	0.391744, 0.460422

Аналогичный тест был проведен для красной стенки сцены. На левой части рис. 12 показана карата ошибки в процентах, а на правой части — полученная цветовая текстурная карта.

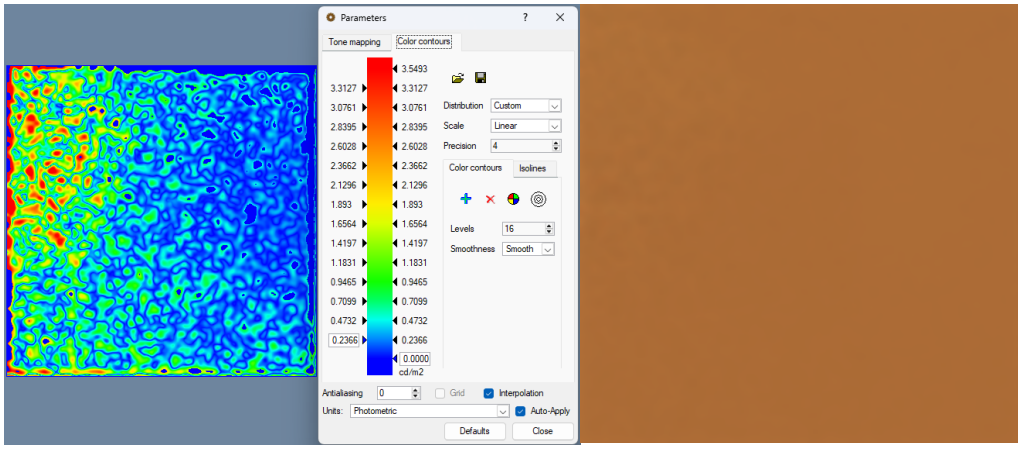


Рис. 12. Карта ошибок в процентах (слева) и восстановленный цвет поверхности (справа) для красной стенки

На рис. 13 показано сравнение оригинального изображения (слева) и результатов рендеринга для восстановленных параметров левой красной стенки (справа).

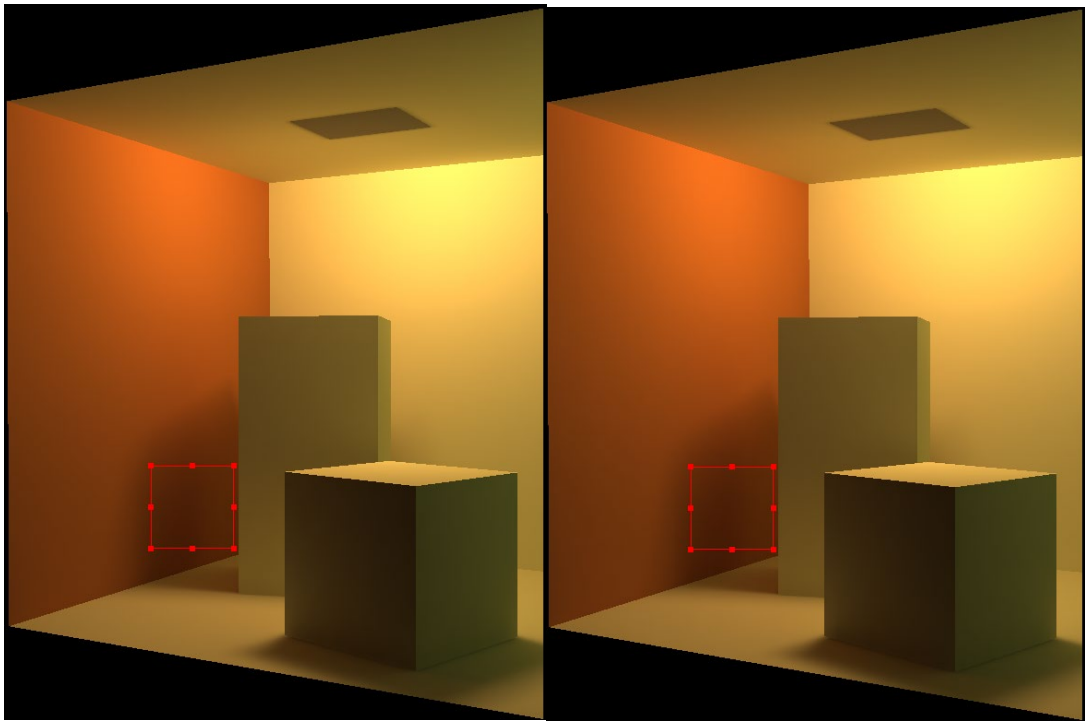
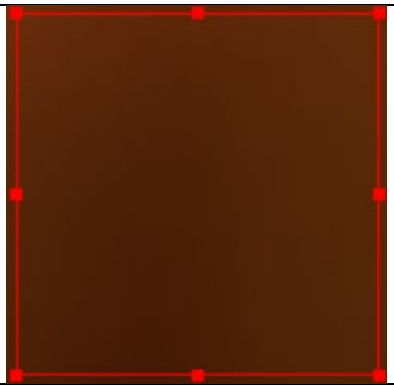
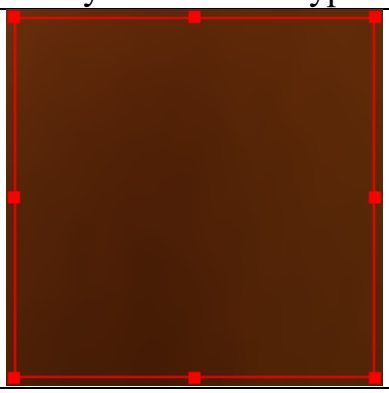


Рис. 13. Оригинальное изображение (слева) и результат рендеринга для восстановленных параметров сцены (справа)

Видно, что результаты практически неразличимы. Таблица 2 демонстрирует данные для выделенной области красной стенки.

Таблица 2. Сравнение средних значений яркости и цвета для области 90 × 90 пикселей на оригинальном изображении и изображении, полученном с помощью рендеринга с восстановленными оптическими свойствами красной стенки

	Оригинал	Рендеринг на полученной текстуре
Изображение		
Средняя яркость (cd/m2)	0.0893196	0.0892520
Среднеквадратичное отклонение яркости (cd/m2)	0.0181009	0.0179845
Цвет RGB (cd/m2)	0.276969, 0.0424213, 0.00120226	0.276751, 0.0423917, 0.001120092
Цветность xy (CIE)	0.562333, 0.38754	0.562329, 0.387545

Видно, что даже в затемненной области, где максимальная ошибка может достигать 3 процентов, средние значения идентичны оригиналу, что свидетельстве о хорошем результате восстановления.

Кроме того, можно провести восстановление на потолке сцены, где прямое освещение отсутствует. В данном случае вторичное освещение устанавливается не как процент от первичного, а как процент от среднего значения освещенности по всей сцене. Такой подход позволяет избежать ситуации, когда объект полностью закрыт от источников света. Для большей наглядности можно установить голубой цвет для потолка. На рис. 14 представлено сравнение оригинального изображения с результатом рендеринга для восстановленных параметров оптических свойств.

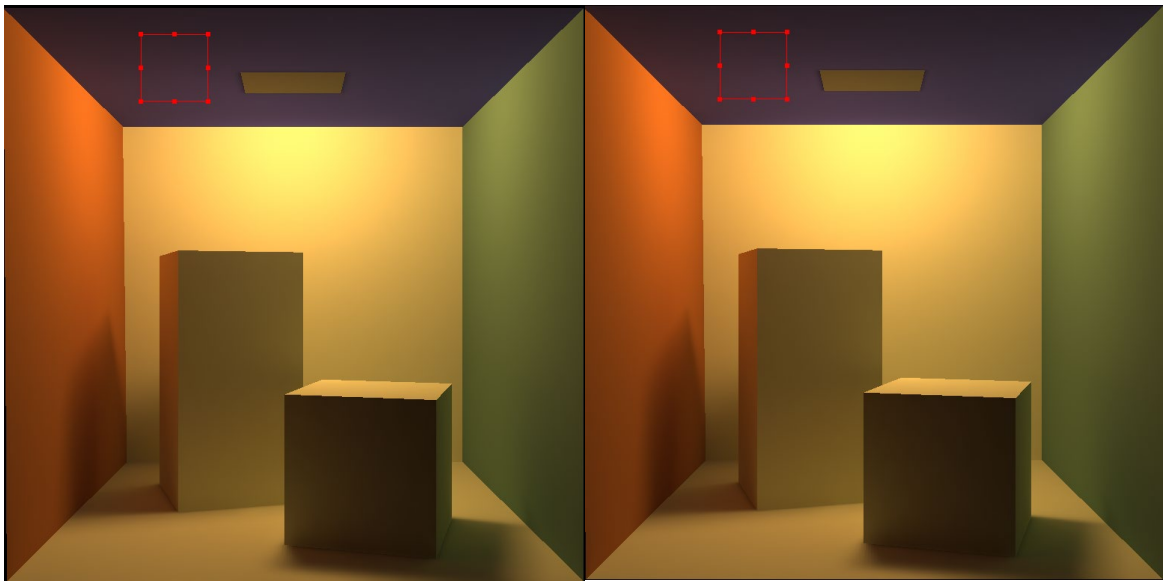
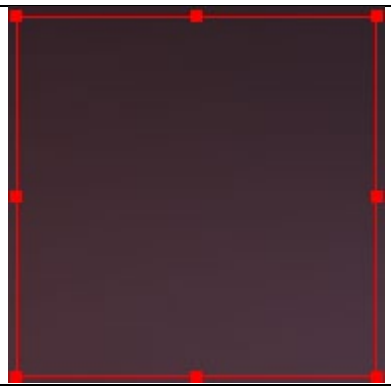


Рис. 14. Оригинальное изображение (слева) и результат рендеринга с восстановленным голубым цветом потолка (справа)

Визуально результаты практически неразличимы. Таблица 3 демонстрирует данные для выделенной области голубого потолка.

Таблица 3. Сравнение средних значений яркости и цвета для области 90 × 90 пикселей на оригинальном изображении и изображении, полученном с помощью рендеринга с восстановленными оптическими свойствами голубого потолка

	Оригинал	Рендеринг на полученной текстуре
Изображение		
Средняя яркость (cd/m2)	0.0781676	0.0781493
Среднеквадратичное отклонение яркости (cd/m2)	0.0214328	0.0215023
Цвет RGB (cd/m2)	0140054, 0.0590389, 0.085381	0.13989, 0.0590521, 0.085481
Цветность xy (CIE)	0.357999, 0.296823	0.357786, 0.296719

Тестирование на текстурированной поверхности

Приведенные выше данные показывают хороший результат для однородных поверхностей. Далее будет продемонстрирована работоспособность алгоритма на текстурированных объектах. Для этой цели на правую стенку была нанесена текстура. На рис. 15 показано изображение сцены с текстурой в виде бабочки.

Рис. 15. Изображение сцены с текстурой в виде бабочки на правой стенке

Используя предложенный алгоритм, можно извлечь карту ошибок и выполнить рендеринг на полученной текстуре. На левой части рис. 16 показана карта ошибок, на правой части представлена исходная текстура бабочки.

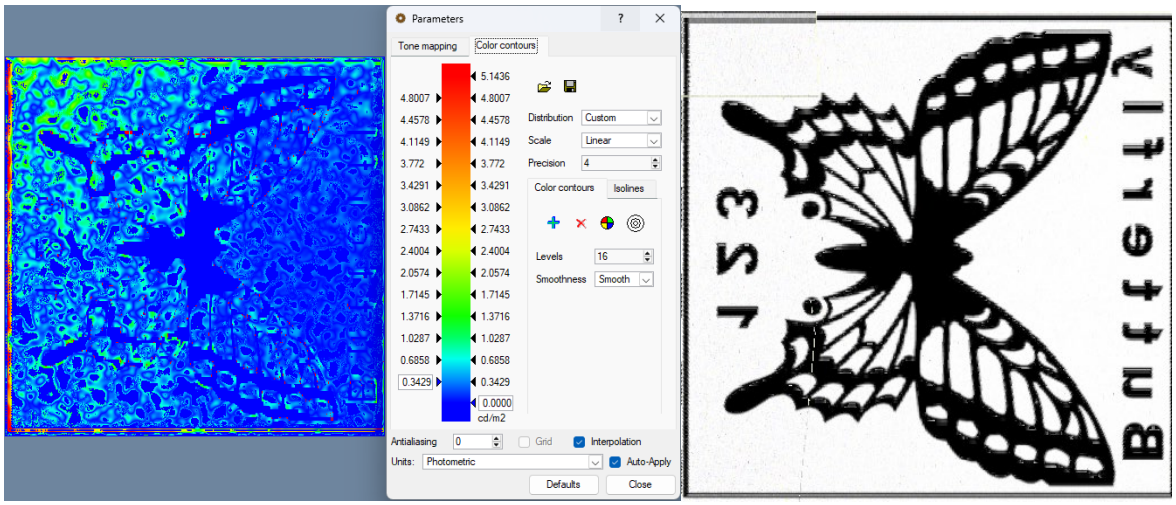


Рис. 16. Карта ошибок текстуры в виде бабочки (слева),
восстановленная текстура (справа)

После выполнения рендеринга с восстановленной текстурой можно сравнить исходные и восстановленные изображения. Результат представлен на рис. 17. На левой части рисунка представлено исходное изображение, на правой части показан результат рендеринга с восстановленной текстурой. Полученная текстура соответствует исходной и имеет те же самые артефакты, что и оригинальная текстура.

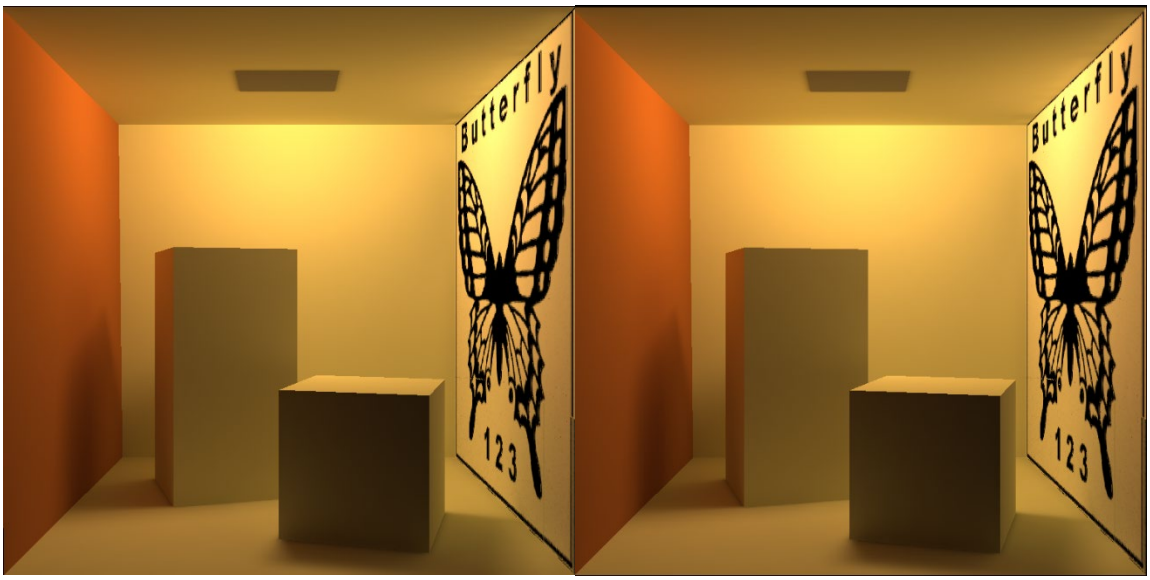


Рис. 17. Исходное изображение (слева), результат рендеринга
для восстановленной текстуре (справа)

Для изображений с текстурой нет возможности выполнить сравнение средних значений яркости и цвета. Но карта ошибок на рис. 16 и результат сравнения на рис. 17 показывают, что текстура была корректно восстановлена.

Далее демонстрируется возможность восстановления «цветной» текстуры. На рис. 18 показан пример изображения текстуры с церковью (слева), карта ошибок (по центру) и результат рендеринга с восстановленной текстурой (справа). Несмотря на сложный характер распределения цвета по поверхности текстуры максимальная ошибка ее восстановления не превышает 2.7%.

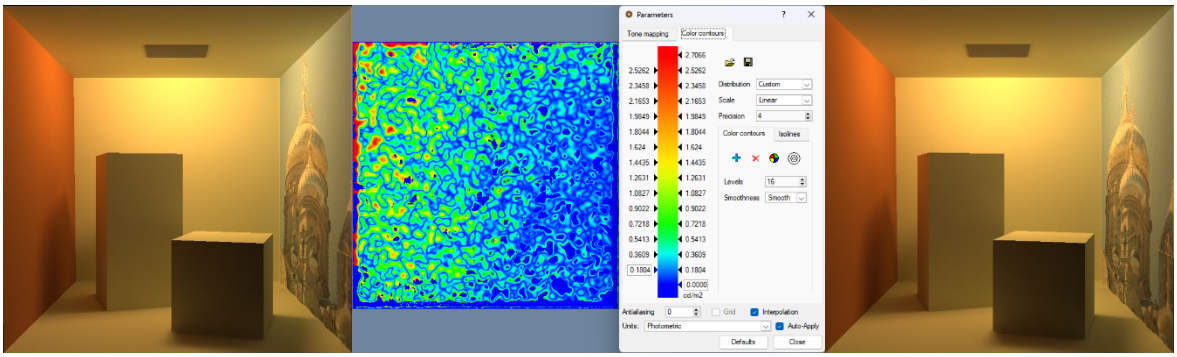


Рис. 18. Исходное изображение церкви (слева), карта ошибок (по центру) и результат рендеринга с восстановленной текстурой (справа)

Цвет текстуры на изображении 18 отличается от исходного цвета текстуры. Рис. 19 демонстрирует, как цвет исходной текстуры (центральное изображение) отличается от ее цвета после прямого и вторичного освещения цветным источником цвета (левое изображение). Разработанный алгоритм восстановления цвета позволяет корректно восстановить исходный цвет текстуры без учета прямого и вторичного освещений (правая часть рис. 19).

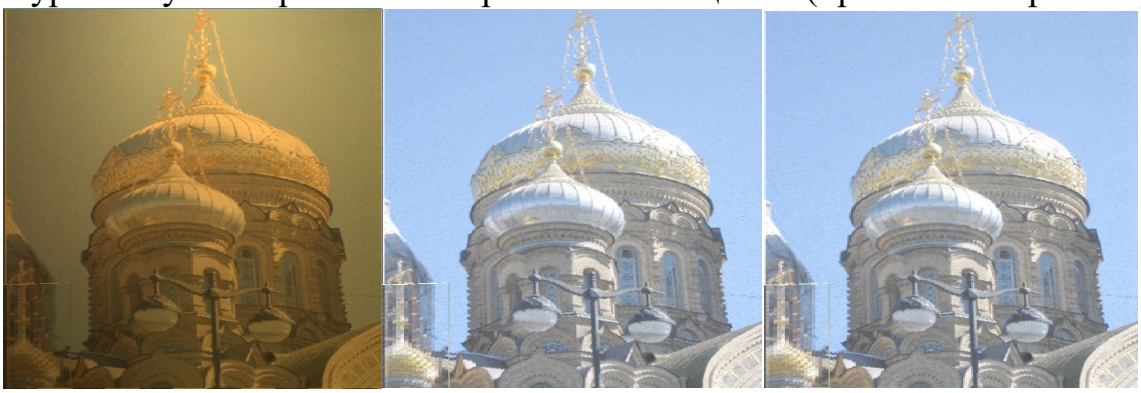


Рис. 19. Слева оригинальное изображение, по центру расположена оригинальная текстура, а справа — восстановленная

Видно, что восстановленная текстура практически совпадает с исходной текстурой, сохраняя мелкие детали, например, небольшой дефект, который присутствовал на исходной текстуре (маленькая белая линия, в нижней левой четверти изображения).

Заключение

Предложенный алгоритм позволяет предварительно построить текстурные карты, собрать яркости по оригинальным изображениям и рассчитать прямое и вторичное освещение на развертке геометрии для ускорения процесса восстановления оптических свойств модели сцены средствами дифференцируемого рендеринга. Кроме того, алгоритм использует эти данные для восстановления цвета поверхности сцены. Проведенные исследования показывают эффективность восстановления цвета поверхности на сценах как с однородными свойствами, так и

сложными цветными текстурами. Данный алгоритм позволяет убрать вариацию цвета объектов сцены и тем самым использовать предложенный ранее метод [1] для выбора наиболее информативных точек по одноцветному изображению сцены. Построенная текстура может быть передана в дифференцируемый рендеринг как вариация цвета, что позволяет снизить размерность задачи дифференцируемого рендеринга и повысить скорость и точность его работы.

Библиографический список

1. Куприянов С.И., Жданов Д.Д., Валиев И.В. Предобработка изображений для решения задач дифференцируемого рендеринга // Препринты ИПМ им. М.В. Келдыша. 2025. № 24. — С. 1-23
2. Desbrun, M., Meyer, M., & Alliez, P. (2002). Intrinsic Parameterizations of Surface Meshes. Computer Graphics Forum, 21(3), 209–218
3. Floater, M.S., & Hormann, K. (2005). Surface Parameterization: a Tutorial and Survey. In Advances in Multiresolution for Geometric Modelling (pp. 157–186). Springer.
4. Lévy, B., Petitjean, S., Ray, N., & Maillot, J. (2002). Least Squares Conformal Maps for Automatic Texture Atlas Generation. ACM Transactions on Graphics, 21(3), 362–371.
5. Sheffer, A., & de Sturler, E. (2001). Parameterization of Faceted Surfaces for Meshing Using Angle-Based Flattening. Engineering with Computers, 17(3), 326–337.
6. Mullen, P., Tong, Y., Alliez, P., & Desbrun, M. (2008). Spectral Conformal Parameterization. Computer Graphics Forum, 27(5), 1487–1494.
7. Zayer, R., Lévy, B., & Seidel, H.-P. (2007). Linear Angle Based Parameterization. In Proceedings of the Fifth Eurographics Symposium on Geometry Processing (pp. 135–141).
8. Sorkine, O., Cohen-Or, D., & Toledo, S. (2002). High-Pass Quantization for Mesh Encoding. In Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Geometry Processing (pp. 42–51).
9. Gu, X., Luo, F., Sun, J., & Yau, S.-T. (2016). Convergence of Discrete Conformal Geometry and Computation of Teichmüller Space. Foundations of Computational Mathematics, 16(3), 649–685.
10. Tatarchenko, M., Dosovitskiy, A., & Brox, T. (2016). Multi-view 3D Models from Single Images with a Convolutional Network. arXiv preprint arXiv:1605.03557.
11. Debevec, P.E., Taylor, C.J., Malik, J. Modeling and Rendering Architecture from Photographs. ACM SIGGRAPH (1996).
12. Matusik, W., et al. Image-Based 3D Photography. Eurographics Workshop on Rendering (2003).

13. Wenqi Xian, Patsorn Sangkloy, Varun Agrawal, Amit Raj, Jingwan Lu, Chen Fang, Fisher Yu, James Hays. TextureGAN: Controlling Deep Image Synthesis with Texture Patches. CVPR (2018).
14. Thies, J., et al. Deferred Neural Rendering. ACM TOG (2020).
15. Mildenhall, B., Srinivasan, P. P., Tancik, M., et al. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis // ACM Transactions on Graphics. 2020. Vol. 39, no. 4.
16. Baatz, H. NeRF-Tex: Neural Reflectance Field Textures. 2021.