DOI: 10.15514/ISPRAS-2026-38(3)-33

Ускоренная кроссплатформенная реализация дифференцируемого рендеринга функций расстояний со знаком

А.С. Будак, ORCID: 0009-0005-6819-4184 <alexey.budak@graphics.cs.msu.ru>
А.Р. Гарифуллин, ORCID: 0000-0003-3802-1774 <albert.garifullin@graphics.cs.msu.ru>
В.А. Галактионов, ORCID: 0000-0001-6460-7539 <vlgal@gin.keldysh.ru>
В.А. Фролов, ORCID: 0000-0001-8829-9884 <frolov@gin.keldysh.ru>
А.Г. Волобой, ORCID: 0000-0003-1252-8294 <voloboy@gin.keldysh.ru>

Институт прикладной математики им. М.В. Келдыша РАН,
Россия, 125047, г. Москва, Миусская пл., д. 4.

Аннотация. Восстановление параметров сцены по изображениям (обратный рендеринг) является востребованной задачей и важным направлением в компьютерной графике и зрении. В настоящее время для решения этой задачи все чаще применяется дифференцируемый рендеринг, основанный на градиентных методах оптимизации. В данной работе представлены усовершенствования метода дифференцируемого рендеринга функций расстояния со знаком, предложенного в 2024 году, а также его кроссплатформенная реализация, поддерживающая выполнение на различных типах графических ускорителей. Таким образом, обеспечивается независимость от оборудования конкретного производителя и расширяется применимость метода в разнородных аппаратных конфигурациях. В нашей работе предлагаются две ключевые модификации. Во-первых, стандартный метод трассировки луча мы заменяем методом Ньютона и аналитическим методом, адаптированными к задачам дифференцируемого рендеринга. Кроме этого, мы разбиваем вычисление производных по текстурным и геометрическим параметрам сцены на две части, соответствующие внутреннему и граничному интегралам. Такое разбиение уменьшает число выборок метода Монте-Карло, необходимых для оценки градиентов по текстуре, и позволяет распределить вычисления между двумя шейдерами. В результате разработанная реализация дифференцируемого рендеринга достигает ускорения до трех раз по сравнению с базовой реализацией, сохраняя при этом исходную точность.

Ключевые слова: компьютерная графика; дифференцируемый рендеринг; трассировка лучей; кроссплатформенность.

Для цитирования: Будак А.С., Гарифуллин А.Р., Галактионов В.А., Фролов В.А., Волобой А.Г. Ускоренная кроссплатформенная реализация дифференцируемого рендеринга функций расстояний со знаком. Труды ИСП РАН, том 38, вып. 3, часть 3, 2026 г., стр. 27–38. DOI: 10.15514/ISPRAS–2026–38(3)–33.

Благодарности: Исследование выполнено за счет гранта Российского научного фонда № 25-11-00054.

Accelerated Cross-platform Implementation of Differentiable Rendering of Signed Distance Function

A.S. Budak, ORCID: 0009-0005-6819-4184 <alexey.budak@graphics.cs.msu.ru>
A.R. Garifullin, ORCID: 0000-0003-3802-1774 <albert.garifullin@graphics.cs.msu.ru>
V.A. Galaktionov, ORCID: 0000-0001-6460-7539 <vlgal@gin.keldysh.ru>
V.A. Frolov, ORCID: 0000-0001-8829-9884 <frolov@gin.keldysh.ru>
A.G. Voloboy, ORCID: 0000-0003-1252-8294 <voloboy@gin.keldysh.ru>

Keldysh Institute of Applied Mathematics of the Russian Academy of Sciences, 4, Miusskaya sq., Moscow, 125047, Russia.

Abstract. Reconstructing scene parameters from images (inverse rendering) is a popular problem and an important area in computer graphics and vision. Differentiable rendering, based on gradient optimization methods, is currently being increasingly applied to this task. This paper presents improvements to the method for differentiable rendering of signed distance functions, proposed in 2024, as well as a cross-platform implementation that supports execution on various types of graphics accelerators. This ensures independence from specific hardware vendors and expands the applicability of the method to heterogeneous hardware configurations. Our paper proposes two key modifications. First, we replace the standard ray tracing method with Newton's method and an analytical method adapted to differentiable rendering problems. Furthermore, we split the calculation of derivatives with respect to texture and geometric scene parameters into two parts, corresponding to the interior and boundary integrals. This partitioning reduces the number of Monte Carlo samples required to estimate texture gradients and allows the computation to be distributed between two shaders. As a result, the developed implementation of differentiable rendering is three times faster compared to the baseline implementation while maintaining the same level of accuracy.

Keywords: computer graphics; differentiable rendering; ray tracing; cross-platform.

For citation: Budak A.S., Garifullin A.R., Galaktionov V.A., Frolov V.A., Voloboy A.G. Accelerated cross-platform implementation of differentiable rendering of signed distance function. Trudy ISP RAN/Proc. ISP RAS, vol. 38, issue 3, part 3, 2026, pp. 27-38 (in Russian). DOI: 10.15514/ISPRAS-2026-38(3)-33.

Acknowledgements. This research was supported by the Russian Science Foundation, grant 25-11-00054.

1. Введение

Обратный рендеринг является широко исследуемым направлением в компьютерном зрении и графике. В последнее десятилетие, на волне активного внедрения методов машинного обучения, оно стремительно развивается в виде дифференцируемого рендеринга, применяющего градиентную оптимизацию для решения задачи обратного рендеринга. Одна группа методов направлена на корректный расчёт градиентов в условиях физически обоснованного рендеринга. Для восстановления поверхности в этих методах применяется неявное представление поверхности, называемое функцией расстояния со знаком (signed distance function, SDF) – это функция, которая для точки пространства возвращает расстояние от этой точки до поверхности представляемого объекта.

Большая часть открытых реализаций методов дифференцируемого рендеринга накладывает ограничения на поддерживаемое аппаратное обеспечение – помимо возможности проведения вычислений на центральном процессоре, что делает процесс оптимизации затратным по времени, они предлагают лишь ускорение с использованием CUDA, требующей графические карты Nvidia для своей работы. При этом нужно учесть, что метод дифференцируемого рендеринга – это итеративный процесс, и каждая его итерация в несколько раз вычислительно затратнее, чем прямой рендеринг. В связи с этим необходимо, чтобы программное решение не было привязано к определённому оборудованию, а производительность являлась не менее важным аспектом, чем точность реконструкции.

2. Обзор работ

Все методы, разработанные с момента появления понятия *дифференцируемый рендеринг* в 2014 году [1], можно разбить на две группы в зависимости от типа представления –объёмного или поверхностного. К группе объёмных представлений относятся такие методы, как нейронные поля освещённости (NeRF) [2] и Gaussian splatting [3]. Методы, использующие поверхностные представления, характеризуются двумя основными направлениями. Первый – нейронный рендеринг функций расстояния [4-6], значения которой хранятся в виде векторов признаков для нейросети. Ранние методы [7, 4] использовали одну нейронную сеть для представления всей сцены, что сказывалось на скорости оптимизации. Эта проблема решалась разными способами: разбиением одной сети на несколько, каждая из которых представляла часть сцены [5, 8]; также для них модифицировался сам алгоритм рендеринга [9]. Второе направление – аналитические методы, не использующие нейронные сети. В них оптимизируются объекты, заданные мешем или функцией расстояния со знаком.

Метод edge sampling [10] в 2018 году впервые получил аналитические производные по параметрам геометрии, в частности меша, в физически-обоснованном рендеринге. Так как в этом случае цвет пикселя вычисляется как интеграл, для дифференцируемого рендеринга требовалась его производная, не равная интегралу производной – помимо неё требовалось оценить ещё один интеграл, областью интегрирования которого являлись силуэт и границы объекта. Специфика метода требовала их явного семплирования, что для меша является нетривиальной задачей. Из-за этого отрисовка одного изображения с расчётом градиентов занимала на порядок больше времени по сравнению с прямым рендерингом. При этом сам процесс оптимизации занимает сотни итераций, на каждой из которых нужно отрисовать несколько изображений сцены. На практике это приводит к ощутимым затратам по времени. В 2019 и 2020 годах появились два связанных между собой метода, борющиеся с главной проблемой edge sampling. Первый подход, репараметризация [11], вносил дополнительные смещение и разброс. Метод 2020 года, warped area sampling (WAS) [12], строго показал, что граничный интеграл, который требуется оценить для получения корректных градиентов, возникает согласно транспортной теореме Рейнольдса, и предложил продлить область определения граничного интеграла на пространство с помощью теоремы Остроградского-Гаусса. Для этого перехода строится специальное векторное поле. Этот метод даёт несмещённую оценку, и авторами было показано, как из построенного векторного поля можно получить репараметризацию, не вносящую смещение.

Три подхода к дифференцируемому рендерингу, описанные выше, использовали меши в практической реализации. Однако меши, с точки зрения задачи реконструкции поверхности, имеют ряд фундаментальных проблем, которые делают их использование нежелательным [13]: невозможность менять топологию объекта (род поверхности), возникновение необратимых самопересечений и неоптимальное распределение полигонов, когда в местах, требующих высокой детализации, не будет достаточного числа примитивов для их представления. Модификации, представленные в работе, решают две последние проблемы частично и не гарантируют стабильную реконструкцию; это, а также оставшееся без решения ограничение на топологию, мешает свободному применению мешей для оптимизации. В дальнейшем работы этой группы методов используют функции расстояния со знаком. В 2022 году к ним впервые применили репараметризацию [14]. сравнивались два подхода – прямое применение метода WAS к функциям расстояния, а также адаптированная под SDF версия. В этой работе было предложено использовать алгоритм редистансинга. Редистансинг – задача пересчёта значений сетки, в частности для SDF в узлах вычисляются корректные расстояния до поверхности. Для решения этой задачи применяются методы, вычисляющие на сетке уравнения в частных производных [15], например уравнение Эйконала [16]. В дифференцируемом рендеринге пересчёт нужен, поскольку после шага оптимизации и

обновления расстояний в узлах сетки она, строго говоря, уже не является SDF. Наконец, в работе 2024 года [17] предлагают переход от граничного интеграла к объёмному без построения векторного поля. Метод использует свойства функций расстояния и оценивает граничный интеграл *интегралом по релаксированной границе* – тонкой линии вокруг силуэта объекта. Использование SDF позволяет находить точки, принадлежащие этой области, во время трассировки лучей: значение функции расстояния в них будет меньше порогового, определяемого гиперпараметром. Этот метод продемонстрировал лучшие результаты в сравнении с [14], поэтому был выбран в качестве базового. Авторами была опубликована реализация метода на Github [18].

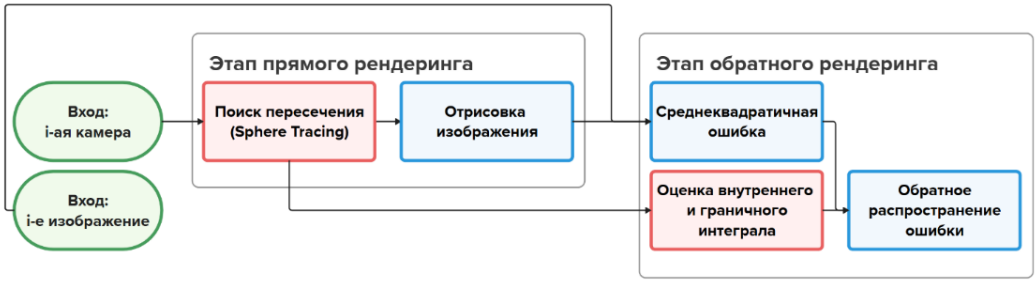
С другой стороны, на методы дифференцируемого рендеринга можно смотреть с точки зрения типа восстанавливаемых параметров. Сложной задачей является одновременное восстановление материала и поверхности объекта [19]. Рассмотренные работы, от edge sampling до базового метода, могут быть использованы для её решения, однако фокус данной работы – восстановление исключительно поверхности объекта. Тем не менее, предложенный метод не вносит никаких ограничений в сравнении с базовым ни в задачу восстановления поверхности, ни в задачу одновременного восстановления поверхности и материала.

Мотивация. В [14, 17] SDF задана на регулярной сетке, и для поиска пересечения луча и функции расстояния применяется метод sphere tracing [20]. Этот итеративный метод, предложенный в 1995 году, вычисляет значение расстояния в точке на луче, и затем делает шаг на это расстояние, пока либо не пересечёт поверхность, либо не выйдет за пределы сцены. Sphere tracing является наиболее распространённым методом для поиска пересечения луча и SDF, однако существуют альтернативы. В [21] были предложены два метода, разработанные для пересечения плотных и разреженных SDF сеток. Вместо шагания по лучу они обходят каждый воксель на его пути. По результатам сравнений, проведённых в [22], метод Ньютона и аналитический метод показали более высокую скорость в сравнении со sphere tracing в прямом рендеринге. Из этого вытекает предположение, что их использование даст прирост скорости и в дифференцируемом рендеринге. Помимо этого, в алгоритм базового метода можно внести изменения, которые дополнительно ускорят семплирование.

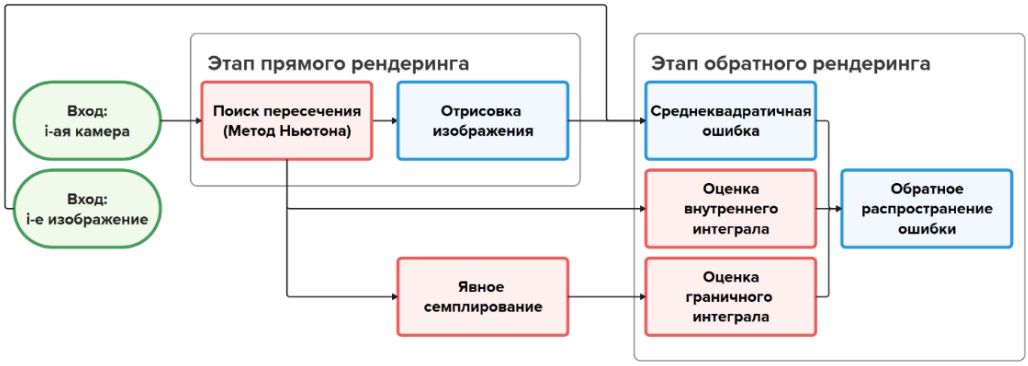
3. Предложенный метод

На рис. 1 показана схема одной итерации для базового и предлагаемого алгоритма. Для базового: сперва выбирается ракурс, с которого было сделано одно из референсных изображений, и генерируются лучи с использованием параметров соответствующей камеры. Затем, методом sphere tracing [20] отрисовывается сцена, а также проводится поиск точек релаксированной границы, после чего оцениваются внутренний и граничный интегралы. Наконец, считается функция потерь между референсным и полученным изображениями, происходит обратное распространение ошибки.

Красным цветом на рис. 1 показаны этапы алгоритма, в которые были внедрены предложенные модификации. Предлагаемый в данной статье метод, во-первых, применяет и адаптирует алгоритмы поиска пересечения луча и функции расстояния из [21] под дифференцируемый рендеринг, а во-вторых, реализует явное семплирование границы. Метод Ньютона заменяет sphere tracing в качестве метода поиска пересечения на этапе прямого рендеринга; на этапе явного семплирования идёт поиск точек для оценки граничного интеграла.



1)



2)

Рис. 1. Схема одной итерации 1) базового, 2) предложенного метода.
Зелёным цветом обозначены входные данные. Синим цветом обозначены этапы, оставшиеся без изменения. Красным цветом выделены модифицированные этапы.
Fig. 1. Diagram of a single iteration of (1) the baseline method and (2) the proposed method.
The input data are shown in green. The stages that remain unchanged are shown in blue.
The modified stages are highlighted in red.

В целях достижения кроссплатформенности, реализация выполнена с использованием C++ и Vulkan. Система Kernel Slicer [23] используется для переноса C++ кода на GPU. В отличие от базового метода, в реализации не использовалась система автоматического дифференцирования, все производные получены аналитически. Разработка началась с реализации прямого рендеринга SDF методом sphere tracing, с поддержкой цвета и простой модели освещения. Следующим шагом была реализация базового метода с аналитическим расчётом градиентов. Для обеспечения параллелизма, на каждой итерации во время отрисовки производные по параметрам записываются в отдельный массив. После расчёта функции потерь в массив добавляется её производная, затем применяется регуляризация, считается производная регуляризации, после чего дистанции SDF сетки обновляются с использованием оптимизатора Adam. В конце каждой итерации в обновлённой сетке пересчитываются значения дистанций. Регуляризация представляет собой свёртку с дискретным оператором Лапласа, выполняется параллельная версия на CPU. Для пересчета значений дистанций была реализована параллельная версия метода fast sweeping [24], выполняемая на CPU.

3.1 Явное семплирование

В физически-обоснованном рендеринге функция расчёта цвета пикселя представима интегралом следующего вида:

$$I = \int_{\Omega} f(\omega;\theta)\, d\omega, \tag{1}$$

где θ – некоторые параметры сцены. При дифференцировании по параметрам, задающим форму объекта, подынтегральная функция f терпит разрывы, чьи положения зависят от параметров дифференцирования. В [12] предлагается разбить этот интеграл на сумму интегралов по областям непрерывности D_i подынтегрального выражения, тогда область интегрирования полученных слагаемых будет зависеть от параметров сцены. Согласно обобщению формулы Лейбница для дифференцирования под знаком интеграла – транспортной теореме Рейнольдса – в этом случае производная интеграла будет равна:

$$\frac{\partial I}{\partial \theta} = \sum_i \int_{D_i} \frac{\partial f(\omega;\theta)}{\partial \theta}\, d\omega + \sum_i \oint_{\partial D_i} f(\omega;\theta)\langle \partial_\theta \omega, \hat{\boldsymbol{n}}\rangle\, d\omega \tag{2}$$

где ∂D_i – граница области D_i, $\hat{\boldsymbol{n}}$ – нормаль, направленная наружу. Первое слагаемое – это *внутренний* интеграл, второе – *граничный* интеграл. Идея базового метода [17] заключается в оценке граничного интеграла интегралом по релаксированной границе A:

$$\oint_{\partial D} f(\omega;\theta)\langle \partial_\theta \omega, \hat{\boldsymbol{n}}\rangle d\omega \;\approx\; I_A = \int_A \frac{1}{l(\omega)} f(\omega;\theta)\langle \partial_\theta \omega, \hat{\boldsymbol{n}}\rangle\, d\omega, \tag{3}$$
$$l(\omega) = \frac{\varepsilon}{\|x - x^*\|}$$

где x – точка, из которой пускаются лучи, $l(\omega)$ – ширина релаксированной границы, x^* – точка релаксированной границы на луче с направлением ω, а ε – гиперпараметр метода: пороговое значение SDF для точек релаксированной границы. В базовом методе граничный интеграл оценивается вместе со внутренним, используя те же семплы. Однако, скорость и точность реконструкции можно повысить, если разделить их оценку и выбрать разное число семплов. Идея явного семплирования заключается в переносе расчёта граничного интеграла в отдельный шейдер (рис. 2). В нём не происходит обновление параметров цвета, поэтому его логика упрощена и для каждого луча проводится меньше операций.

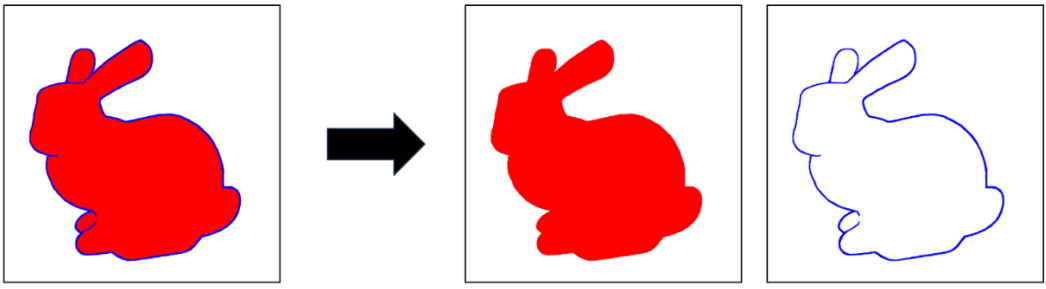


Рис. 2. Визуализация явного семплирования. Вместо одновременного оценивания внутреннего и граничного интеграла (семплами красного и синего цвета соответственно), расчёт проводится сначала для внутреннего, затем для граничного интеграла.

Fig. 2. Visualization of explicit sampling. Instead of simultaneously estimating the interior and boundary integrals (using red and blue samples, respectively), the computation is performed first for the interior integral, and then for the boundary integral.

Благодаря такому разбиению, для оценки граничного интеграла можно сделать больше семплов, требующих меньше вычислений, и уменьшить число семплов для оценки внутреннего. Если в эталонной реализации использовалось 64 семпла на пиксель, то в предложенном методе для внутреннего интеграла выбрано 16, а для граничного 128 семплов.

3.2 Альтернативные методы поиска пересечения

Для пересечения луча с SDF были реализованы метод sphere tracing, а также метод Ньютона и аналитический метод. Последние два метода обходят каждый воксел на пути луча, пока не будет найдено пересечение или луч не выйдет за пределы сцены. Для регулярной сетки, содержащей большое число «пустых» вокселей (не содержащих поверхность), эти методы были реализованы таким образом, чтобы во время поиска пересечения воксел игнорировался, если все значения дистанций в нём больше порогового значения ε – такие воксели гарантированно не внесут вклад в процесс реконструкции, поэтому алгоритм сразу же идёт дальше.

Следующим шагом была адаптация альтернативных методов под оценку граничного интеграла. Точка, лежащая на луче, принадлежит релаксированной границе, если значение функции расстояния в этой точке меньше ε, а её производная вдоль луча в точке равна нулю, то есть она является точкой локального минимума SDF вдоль луча, см. рис 3. Оба метода основаны на том, что внутри вокселя функция расстояния вдоль луча представима в виде полинома третьей степени, для которого двумя разными способами находятся корни. Таким образом, для поиска точек релаксированной границы в аналитическом методе и методе Ньютона нужно в каждом вокселе находить нули его производной (корни квадратного уравнения), а затем проверять значение функции расстояния в них. Метод Ньютона уже находит их для начального приближения, требуется только проверка значения SDF, поэтому он используется в сравнениях. При использовании трилинейной интерполяции нередки случаи, когда локальный минимум достигается на границе вокселей, этот случай обрабатывается отдельно.

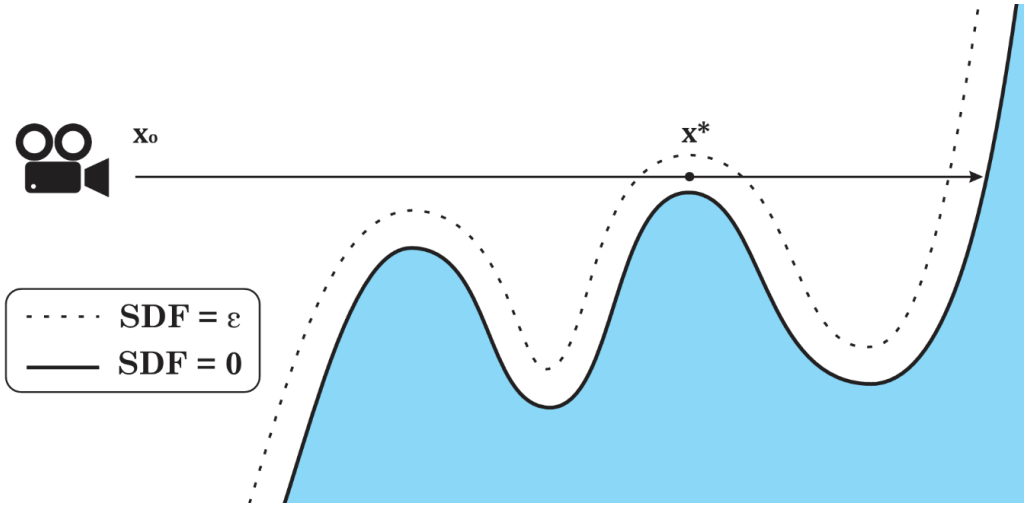


Рис. 3. Визуализация пересечения луча, выпущенного из точки $\boldsymbol{x_0}$, с SDF. Точка $\boldsymbol{x}^*$ принадлежит релаксированной границе, так как 1) SDF вдоль луча достигает в $\boldsymbol{x}^*$ локального минимума, и 2) $SDF(\boldsymbol{x}^*) < \varepsilon$.

Fig. 3. Visualization of a ray intersecting an SDF, emitted from a point $\boldsymbol{x_0}$. The point $\boldsymbol{x}^*$ belongs to the relaxed boundary because (1) SDF along the ray reaches a local minimum at $\boldsymbol{x}^*$ and (2) $SDF(\boldsymbol{x}^*) < \varepsilon$.

4. Результаты экспериментов

Проводились сравнения скорости работы и точности реконструкции для базового и предложенного метода. Так как эталонная реализация написана с использованием CUDA, в сравнениях с ней использовалась NVidia GeForce RTX 4090. Эксперименты проводились для

эталона и предложенных методов на идентичных сценах и ракурсах камер. Использовалось 6 моделей: Armadillo, Stanford Bunny, Asian Dragon, Happy Buddha – модели из репозитория 3D сканов Стэнфорда [25], Utah Teapot – модель чайника за авторством Мартина Ньюэлла; Nefertiti Scan – скан бюста Нефертити [26]. Такой выбор моделей продемонстрирует работу метода в сравнении с эталоном для объектов с разной степенью детализированности поверхности.

Эталон и предложенный метод используют одинаковую конфигурацию для экспериментов. Применяется модель освещения Ламберта с двумя направленными источниками света. Значение гиперпараметра ε взято из оригинальной работы и равно 10^{-4}. Для восстановления используются изображения размером 512x512 пикселей, полученные отрисовкой модели-референса с 16 ракурсов. На каждой итерации расчёт производных проводился для 4 ракурсов из 16. Оптимизация занимала 1000 итераций; каждые 200 итераций разрешение сетки увеличивалось в два раза по каждой оси, размер финальной сетки 512^3.

В проводимых сравнениях скорости как для предложенного метода, так и для эталонной реализации замерялось время выполнения шейдеров, так как в обоих случаях ключевые этапы алгоритма – трассировка сцены, поиск точек релаксированной границы и оценка интегралов – выполняются полностью на GPU. Для тестов, использующих явное семплирование границы, время означает среднее суммарное время работы шейдеров для внутреннего и граничного интегралов; в остальных случаях указано среднее время выполнения шейдера для внутреннего интеграла. В табл. 1 приведено среднее время выполнения одной итерации для рассматриваемых представлений. Тестировались эталонная реализация, собственная реализация базового метода, две предложенные модификации и их комбинация. ЯС в таблице означает явное семплирование. Реконструкция также проводилась на интегрированных графических ускорителях для CPU AMD Ryzen 9 7950X и Intel Core i5-11300H. Эталонная реализация не может быть включена в эти замеры, поскольку GPU этих производителей не поддерживают CUDA. Во всех случаях замерялось среднее время выполнения одной итерации на графическом процессоре для 6 моделей. Из полученных результатов можно сделать следующие выводы:

1. Собственная реализация базового метода сопоставима с эталонной по скорости.
2. Каждая из предложенных модификаций ускоряет базовый метод.
3. Метод со всеми улучшениями показывает лучший результат, ускоряя базовый метод в три раза на Nvidia GeForce RTX 4090.

Табл. 1. Среднее время одной итерации (GPU ядро) в миллисекундах на Nvidia GeForce RTX 4090, включая эталонную реализацию на CUDA, а также AMD Radeon Graphics и Intel Iris Xe Graphics.
Table 1. The average time per iteration (GPU kernel), in milliseconds, on NVidia GeForce RTX 4090, including baseline CUDA implementation, and AMD Radeon Graphics and Intel Iris Xe Graphics.

GPU	Базовый	Базовый (свой)	Ньютон	ЯС	Ньютон+ЯС
NVidia GeForce RTX 4090	70.1	74.7	26.7	28.4	**24.0**
Intel Iris Xe Graphics	–	1809.7	1554.1	523.8	**388.4**
AMD Radeon Graphics	–	1996.4	877.5	676.6	**241.0**

Для валидации было проведено сравнение результатов работы базового и итогового метода на рассматриваемых моделях с исходным мешем. После окончания процесса оптимизации сравнивалось среднее значение PSNR между восстановленной моделью и референсом на

ракурсах, использованных в оптимизации. На рис. 4 приведено сравнение качества для эталона, собственной реализации базового метода и метода с комбинацией всех предложенных улучшений. Из него следует, что качество реализованных методов сопоставимо с эталонной реализацией. Предложенный метод показывает в среднем лучшую точность в сравнении с базовым, так как он делает больше семплов для оценки граничного интеграла, получая более точные градиенты. На рис. 5 приведены примеры реконструкции рассмотренных моделей.

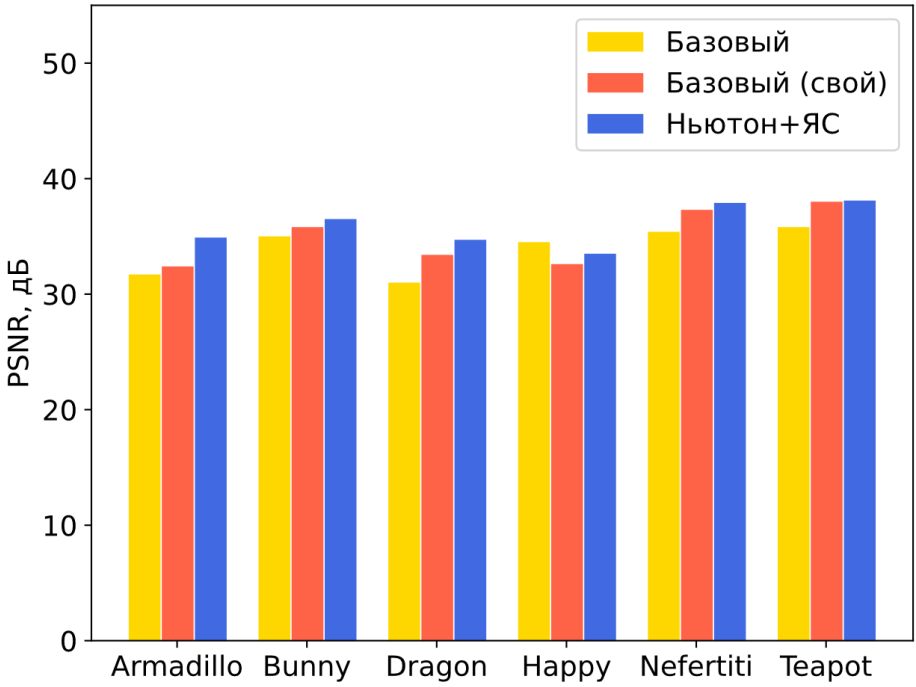


Рис. 4. Среднее по ракурсам значение PSNR для восстановленных моделей.
Fig. 4. Average PSNR across viewpoints for reconstructed models.

5. Заключение

В данной работе представлена кроссплатформенная система дифференцируемого рендеринга SDF, основанная на методе релаксированной границы и разработанная с использованием C++ и Vulkan. Предложены алгоритмические улучшения, в несколько раз ускоряющие процесс реконструкции при сравнимой точности результата. Реализованный метод не вводит новых ограничений. Однако он унаследовал ограничение, связанное с использованием регулярной SDF сетки: для восстановления объектов с мелкими деталями требуется повышать разрешение всей сетки, что значительно увеличивает размер модели. При этом подавляющая часть вокселей в этой сетке представляет пустое пространство. С точки зрения представления поверхности значения дистанций, хранимые в этих вокселях, не влияют на качество модели, и их можно было бы отбросить. Дальнейшая работа будет посвящена адаптации реализованного метода под разреженные представления.

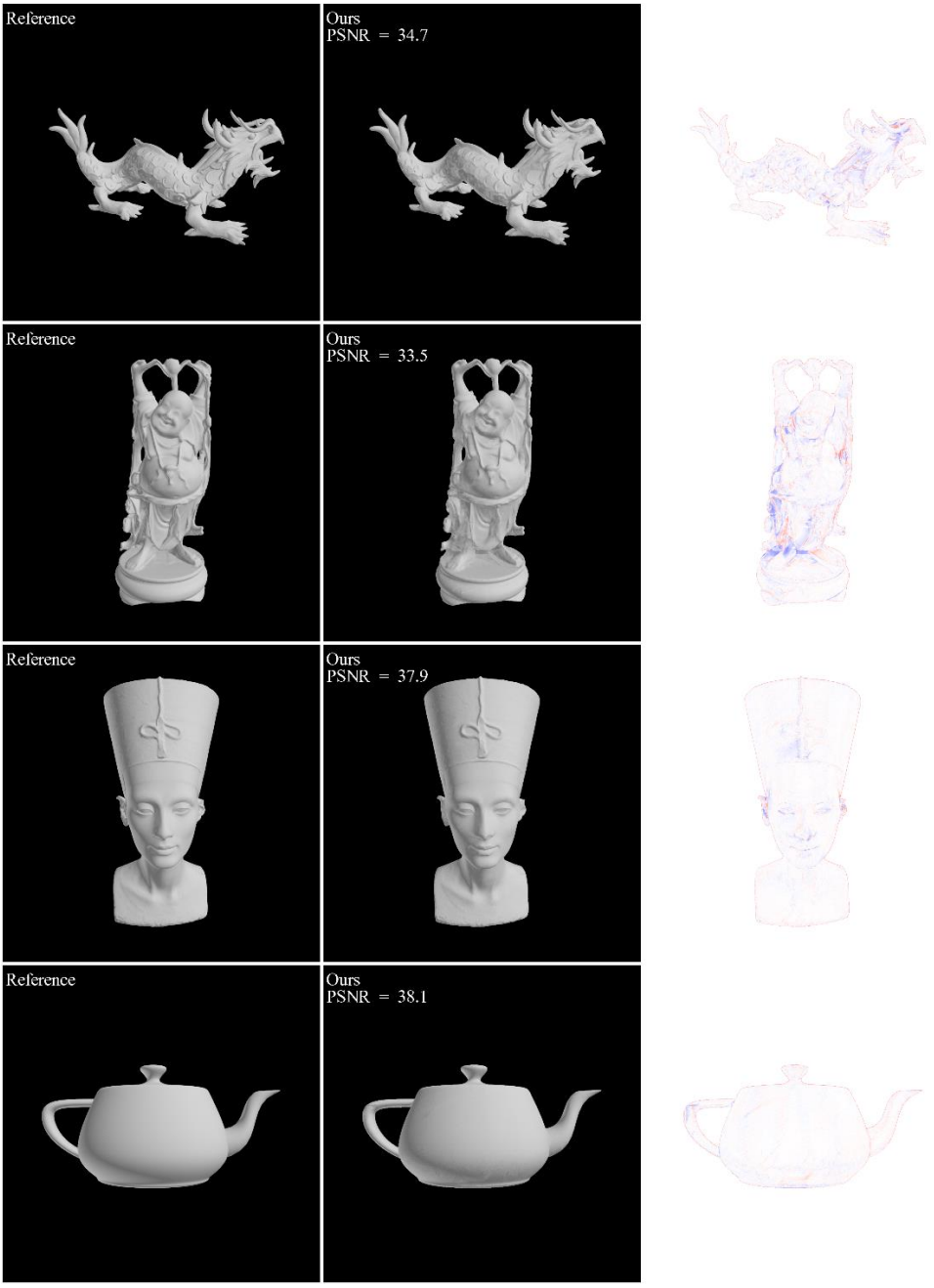


Рис. 5. Примеры реконструкции. Левый столбец: референс, средний: реконструкция, в правом показана знаковая разница между изображениями реконструкции и референса: красным и синим цветом показана положительная и отрицательная разница соответственно. Контраст увеличен для наглядности.

Fig. 5. Reconstruction examples. References are in the left column, reconstructions are in the middle, right column shows signed difference between reconstruction and reference images. Red and blue colors show positive and negative parts respectively. Contrast has been increased for clarity.

Список литературы / References

[1]. Loper M.M., Black M.J. OpenDR: An Approximate Differentiable Renderer, in Computer Vision. ECCV 2014, Zurich, Switzerland, 2014.

[2]. Mildenhall B., Srinivasan P.P., Tancik M., Barron J.T., Ramamoorthi R., Ng R. NeRF: representing scenes as neural radiance fields for view synthesis, Commun. ACM, 2021, vol. 65, no. 1, pp. 99–106.

[3]. Kerbl B., Kopanas G., Leimkuehler T., Drettakis G. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. ACM Trans. Graph., 2023, vol. 42, no. 4, pp. 1-14.

[4]. Wang P., Liu L., Liu Y., Theobalt C., Komura T., Wang W. NeuS: Learning Neural Implicit Surfaces by Volume Rendering for Multi-view Reconstruction. In Advances in Neural Information Processing Systems, 2021.

[5]. Müller T., Evans A., Schied C., Keller A. Instant neural graphics primitives with a multiresolution hash encoding. ACM Trans. Graph., 2022, vol. 41, no. 4, pp. 1-15.

[6]. Li Z., Müller T., Evans A., Taylor R. H., Unberath M., Liu M.-Y., Lin C.-H. Neuralangelo: High-Fidelity Neural Surface Reconstruction. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, 2023.

[7]. Park J.J., Florence P., Straub J., Newcombe R., Lovegrove S. DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation. In IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, California, 2019.

[8]. Wang Y., Han Q., Habermann M., Daniilidis K., Theobalt C., Liu L. NeuS2: Fast Learning of Neural Implicit Surfaces for Multi-view Reconstruction. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Paris, 2023.

[9]. Dogaru A., Ardelean A.T., Ignatyev S., Zakharov E., Burnaev E. Sphere-Guided Training of Neural Implicit Surfaces. In IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Los Alamitos, California, 2023.

[10]. Li T.-M., Aittala M., Durand F., Lehtinen J. Differentiable Monte Carlo ray tracing through edge sampling. ACM Trans. Graph., 2018, vol. 37, no. 6, pp. 1-11.

[11]. Loubet G., Holzschuch N., Jakob W. Reparameterizing discontinuous integrands for differentiable rendering. ACM Trans. Graph., 2019, vol. 38, no. 6, pp. 1-14.

[12]. Bangaru S.P., Li T.-M., Durand F. Unbiased warped-area sampling for differentiable rendering. ACM Trans. Graph., 2020, vol. 39, no. 6, pp. 1-18.

[13]. Nicolet B., Jacobson A., Jakob W. Large steps in inverse rendering of geometry. ACM Trans. Graph, 2021., vol. 40, no. 6, pp. 1-13.

[14]. Vicini D., Speierer S., Jakob W. Differentiable signed distance function rendering. ACM Trans. Graph., 2022, vol. 41, no. 4, p. 18.

[15]. Sethian J.A. A Fast Marching Level Set Method for Monotonically Advancing Fronts. In National Academy of Sciences of the United States of America, Washington, 1996.

[16]. Zhao H. A fast sweeping method for Eikonal equations. Math. Comp., 2005, vol. 74, pp. 603-627.

[17]. Wang Z., Deng X., Zhang Z., Jakob W., Marschner S. A Simple Approach to Differentiable Rendering of SDFs. In SIGGRAPH Asia 2024 Conference Papers, Tokyo, 2024.

[18]. Wang Z. The official implementation for 'A Simple Approach to Differentiable Rendering of SDFs'. Available at: https://github.com/zichenwang01/relaxed-boundary, accessed: 03. 02. 2026.

[19]. Luan F., Zhao S., Bala K., Dong Z. Unified Shape and SVBRDF Recovery using Differentiable Monte Carlo Rendering. Computer Graphics Forum, 2021, vol. 40, no. 4, pp. 101-113.

[20]. Hart J. Sphere Tracing: A Geometric Method for the Antialiased Ray Tracing of Implicit Surfaces. The Visual Computer, 1996, vol. 12, pp. 527–545.

[21]. Söderlund H.H., Evans A., Akenine-Möller T. Ray Tracing of Signed Distance Function Grids. Journal of Computer Graphics Techniques (JCGT), 2022, vol. 11, no. 3, pp. 94-113.

[22]. Budak A.S., Garifullin A.R., Frolov V.A., Galaktionov V.A. Study of Surface Representation Methods Based on Signed Distance Functions. Programming and Computer Software, 2025, vol. 51, pp. 131-139.

[23]. Frolov V.A., Sanzharov V.V., Galaktionov V.A. Kernel_slicer: high-level approach on top of GPU programming API. In Ivannikov Ispras Open Conference (ISPRAS OPEN), Moscow, 2022.

[24]. Detrixhe M., Gibou F., Min C. A parallel fast sweeping method for the Eikonal equation. Journal of Computational Physics, 2013, vol. 237, pp. 46-55.

[25]. Stanford Computer Graphics Laboratory. The Stanford 3D Scanning Repository. Available at: https://graphics.stanford.edu/data/3Dscanrep/, accessed: 03.02.2026.

[26]. Nelles J.N., Al-Badri N. Nefertiti scan, 2015. Available at: https://nefertitihack.alloversky.com/, accessed. 03.02.2026.

Информация об авторах / Information about authors

Алексей Сергеевич БУДАК – студент кафедры Интеллектуальных информационных технологий факультета ВМК МГУ, программист Института прикладной математики им. М.В. Келдыша РАН. Сфера научных интересов: реалистичная компьютерная графика, рендеринг в реальном времени, дифференцируемый рендеринг.

Alexey BUDAK – student at the Department of Intelligent Information Technologies, Faculty of Computational Mathematics and Cybernetics, Moscow State University, programmer at the Keldysh Institute of Applied Mathematics RAS. Research interests: realistic computer graphics, real-time rendering, differentiable rendering.

Альберт Рустемович ГАРИФУЛЛИН – аспирант и младший научный сотрудник Института прикладной математики им. М.В. Келдыша РАН. Сфера научных интересов: реалистичная компьютерная графика, дифференцируемый рендеринг, спектральный и нейронный рендеринг.

Albert GARIFULLIN – postgraduate student and junior researcher at the Keldysh Institute of Applied Mathematics RAS. Research interests: realistic computer graphics, differentiable rendering, spectral and neural rendering.

Владимир Александрович ГАЛАКТИОНОВ – доктор физико-математических наук, профессор, главный научный сотрудник отдела компьютерной графики и вычислительной оптики Института прикладной математики им. М.В. Келдыша РАН. Сфера научных интересов: компьютерная графика, оптическое моделирование, компьютерная лингвистика, научная визуализация.

Vladimir GALAKTIONOV – Dr Sci. (Phys.-Math.), Prof., Chief researcher at the Keldysh Institute of Applied Mathematics RAS. Research interests: computer graphics, optical simulation, computer linguistics, scientific visualization.

Владимир Александрович ФРОЛОВ – кандидат физико-математических наук, старший научный сотрудник Института прикладной математики им. М.В. Келдыша РАН, научный сотрудник факультета ВМК МГУ. Сфера научных интересов: реалистичная компьютерная графика, моделирование освещённости, разработка программных систем оптического моделирования, параллельные и распределённые вычисления.

Vladimir FROLOV – Cand. Sci. (Phys.-Math.), senior researcher at the Keldysh Institute of Applied Mathematics RAS and researcher in computer graphics at Moscow State University. Research interests: realistic computer graphics, light transport simulation, elaboration of optical simulation software systems, GPU computing.

Алексей Геннадьевич ВОЛОБОЙ – доктор физико-математических наук, ведущий научный сотрудник ИПМ им. М.В. Келдыша РАН. Область научных интересов: компьютерная графика, вычислительная оптика, трассировка лучей, моделирование освещенности.

Alexey VOLOBOY – Dr Sci. (Phys.-Math.), leading researcher at the Keldysh Institute of Applied Mathematics of RAS. Research interests: realistic computer graphics, computational optics, ray tracing techniques, lighting simulation.