УДК 378:004
DOI: 10.25686/978-5-8158-2474-4-2025-53-60

Приближенный рендеринг B-сплайновых поверхностей с тримингом в реальном времени

М. С. Кунц[1], А. Р. Гарифуллин[2], И.О. Коротаев[1], В.А. Фролов[1,2]
[1]Институт перспективных исследований проблем искусственного интеллекта и интеллектуальных систем МГУ имени М.В. Ломоносова, Москва, Россия
[2] Институт прикладной математики им. М.В. Келдыша РАН, Москва, Россия

Аннотация. В данной работе предлагаются два улучшения существующего метода рендеринга поверхностей на основе трассировки лучей. Первое улучшение нацелено на борьбу с артефактами рендеринга, возникающими из-за того, что в существующих подходах на основе метода Ньютона не гарантирована сходимость к ближайшему пересечению. Предложен метод стохастического выбора начального приближения, позволяющий получить корректное изображение сцены, состоящей из NURBS поверхностей, в течение нескольких кадров за счет сохранения правильного начального приближения. Второе улучшение нацелено на ускорение процесса тримминга (или обрезки) поверхности после того, как пересечение уже найдено. Предложенный метод на основе двумерных функций расстояния ускоряет рендеринг в среднем от 1.5 до 2 раз ценой незначительного понижения точности тримминга и, кроме того, позволяет практически без накладных расходов отображать ребра объекта.
***Ключевые слова*:** трассировка лучей, NURBS, функции расстояния со знаком.

Approximate real-time rendering of trimmed B-spline surfaces

M. S. Kunts[1], A.R. Garifullin[2], I.O. Korotaev[1], V.A. Frolov[1,2]
[1]IAI Moscow State University, Moscow, Russia
[2]Keldysh Institute of Applied Mathematics, Moscow, Russia

Abstract. This paper proposes two enhancements to an existing ray tracing-based surface rendering method. The first enhancement addresses rendering artifacts that arise because existing Newton-based approaches do not guarantee convergence to the closest intersection. A stochastic initial guess selection method is proposed, which enables the generation of a correct image for a scene composed of NURBS surfaces within a few frames by preserving a valid initial approximation. The second enhancement aims to accelerate the surface trimming (or clipping) process after the intersection has been found. The proposed method, based on two-dimensional distance functions, accelerates rendering by an average factor of 1.5 to 2 at the cost of a minor reduction in trimming accuracy. Furthermore, it allows for the visualization of object edges with virtually no computational overhead.
***Keywords*:** ray tracing, NURBS, signed distance functions.

Введение

NURBS (Non Uniform Rational B-Spline) поверхности являются стандартом геометрического моделирования в современных САПР-системах благодаря своей гибкости и точности представления как элементарных, так и сложных форм. Они широко применяются в промышленном дизайне, авиа- и машиностроении, а также в других областях, требующих работы с параметрическими поверхностями с высокой точностью.

Однако рендеринг таких поверхностей является сложной и не до конца решенной в настоящий момент проблемой. Фундаментальная сложность этой задачи заключается в оптимизации трех ключевых характеристик алгоритма рендеринга: точность, скорость, потребляемая память. На практике приходится соблюдать баланс между этими характеристиками, подходящий для целевой задачи и вычислительного оборудования.

В данной работе мы предлагаем улучшение существующего метода рендеринга B-сплайновых поверхностей на основе трассировки лучей и позиционируем разработанные алгоритмы для приложений предварительного просмотра инженерных моделей, состоящих из B-сплайновых поверхностей, на ограниченном по ресурсам и возможностям графическом оборудовании (GPU), а также для рендеринга на центральном процессоре (CPU).

Обзор существующих работ

1. Растеризация

Растеризация триангулированных поверхностей в настоящий момент является основным методом рендеринга B-сплайновых моделей. Например, этот подход использует библиотека OpenCASCADE [1].

Данный подход имеет принципиальные ограничения, связанные с необходимостью преобразования параметрических поверхностей в полигональные сетки. Это сказывается на точности получаемого изображения и на потребляемой памяти. Кроме того, растеризация плохо масштабируется и показывает низкую производительность на сложных сценах, когда размер треугольника сопоставим с размером пикселя или меньше его.

Чтобы обойти указанные ограничения, был предложен метод динамической тесселяции ETER [2], адаптирующий плотность триангуляции на основе расстояния до камеры и локальной кривизны поверхности. Это программное окружение реализует алгоритмы, завязанные на такие аппаратные возможности GPU, как консервативная растеризация, тензорные ядра и меш-шейдеры. Реализация метода динамической тесселяции из работы [2] в теории возможна и на графических процессорах, не имеющих полной поддержки указанной аппаратной функциональности, но эффективность такого решения находится под вопросом.

2. Трассировка лучей

Численный метод пересечения лучей с усеченными рациональными поверхностями Безье был предложен в работе [3]. В этом подходе задача нахождения пересечений луча с поверхностью сводится к задаче построения выпуклой оболочки поверхности в специальном пространстве, которая используется для оценки диапазонов параметров, гарантированно содержащих искомые точки. Хотя метод обеспечивает высокую точность и гарантирует нахождение всех точек пересечения, он может давать сбой в некоторых особых случаях. Кроме того, метод обладает высокой вычислительной сложностью, поскольку требует выделения локальных участков поверхности и построения их выпуклых оболочек. Также он потребляет память, зависящую от параметров поверхности, что затрудняет его эффективную реализацию на видеокартах.

Для ускорения обработки сложных моделей с B-сплайновыми поверхностями был разработан подход, основанный на использовании BVH (иерархии ограничивающих объемов) в сочетании с методом Ньютона-Рафсона для поиска пересечений лучей с поверхностью [4]. Ключевая особенность метода – использование эвристики, учитывающей кривизну поверхности для выбора оптимального уровня подразделения, минимизирующего вероятность расхождения метода. Однако пересечение с B-сплайновыми поверхностями без преобразований имеет высокую вычислительную сложность, что может повлиять на производительность.

Последующие исследования были направлены на оптимизацию алгоритмов пересечения. В работе [5] предложены усовершенствования метода отсечения Безье. Комбинация отсечения Безье с методом Ньютона-Рафсона позволила увеличить скорость сходимости на 30-50 %, а введение адаптивных порогов ошибок устранило проблемы с бесконечными циклами.

Современные подходы к рендерингу усеченных B-сплайновых поверхностей демонстрируют высокую точность и качество сглаживания [6]. Ключевой особенностью является использование kd-деревьев для эффективной организации обрезающих кривых, сгруппированных в «монотонные наборы»: последовательность связных кривых одинаковой монотонности. Применение метода бисекции обеспечивает точное определение областей усечения, а ускоряющие структуры позволяют достичь высокой производительности.

Дальнейшая оптимизация была достигнута в работе [7], которая была направлена на минимизацию дорогостоящего поиска пересечения с кривыми и на расширение случаев быстрой классификации, не требующей сложных вычислений. Сохраняя точность рендеринга, данный подход в настоящее время является наиболее производительным решением для трассировки лучей на GPU, демонстрируя скорость обработки до нескольких сотен миллионов лучей в секунду для среднесложных сцен.

Работа [8] использует в качестве начального приближения пересечение с полигональной сеткой, которое может быть эффективно посчитано, используя существующие библиотеки и средства аппаратного ускорения трассировки лучей на CPU и на GPU. В этой работе использовалось пересечение с единичной поверхностью линзы, и не было реализовано усечение (тримминг). Мы считаем это перспективным направлением исследований, так как неочевидна его применимость к рендерингу объектов, состоящих из нескольких NURBS поверхностей, а также поверхностей с тримингом.

Предложенный метод

Наша работа является развитием метода, предложенного в [7]. Этот метод состоит из трех основных этапов: (1) обход BVH-дерева; (2) поиск пересечения луча и Безье-поверхности методом Ньютона–Рафсона; (3) усечение (тримминг) поверхности с использованием двумерного kd-дерева. Предложенный метод модифицирует два последних шага. В действительности этот подход требует предобработку, так как алгоритмы вычисления точек и производных для B-сплайновых поверхностей S(u, v) обладают высокой вычислительной сложностью и требуют значительных объемов памяти [9], что делает их неэффективными для реализации на GPU. Поэтому перед построением BVH-дерева работа [7], как и наша работа, использует преобразование B-сплайновых поверхностей в поверхности Безье с помощью алгоритма вставки узлов [10].

1. Метод стохастического выбора начального приближения

Метод Ньютона–Рафсона, несмотря на обеспечиваемую квадратичную скорость сходимости, критически зависит от выбора начального приближения и в общем случае не гарантирует сходимость даже при достаточно сильном разбиении геометрии BVH-деревом (хотя в этом случае вероятность появления артефактов уменьшается ценой повышения потребления памяти) [7]. Основная идея нашего метода заключается в том, чтобы варьировать начальное приближение на разных кадрах. Если хотя бы в одном кадре было получено ближайшее пересечение, дальнейшие вариации будут отброшены (рис. 1).

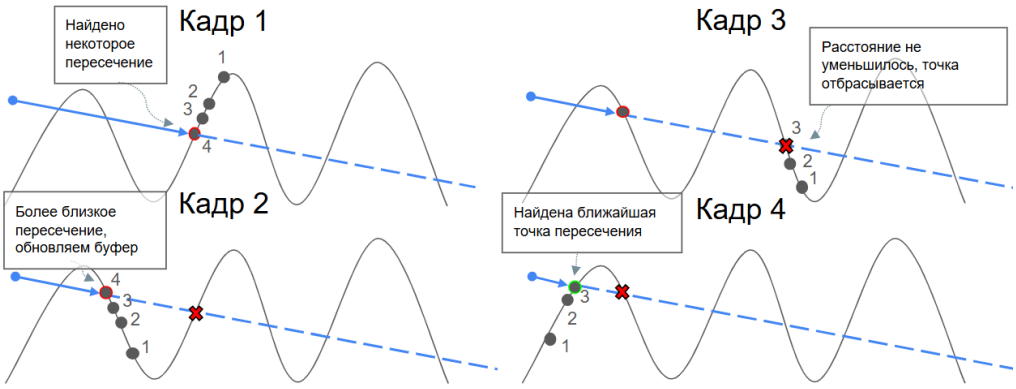


Рисунок 1. Иллюстрация идеи работы предложенного метода для нескольких кадров

Для формализации подхода введем унифицированное параметрическое пространство. Область определения каждого подучастка поверхности отобразим в единичный квадрат $[0,1]^2$. Такая нормализация позволяет единообразно задавать начальные приближения для всех участков поверхности. Например, чтобы для всех участков начальные приближения были в их параметрических центрах, можно использовать глобально заданные в унифицированном пространстве значения $(\hat{u},\hat{v}) = (\frac{1}{2},\frac{1}{2})$. В таком случае для каждого участка начальное приближение вычисляется следующим образом:

$$(u_0, v_0) = (\frac{u_{min} + u_{max}}{2}, \frac{v_{min} + v_{max}}{2}). \quad (1)$$

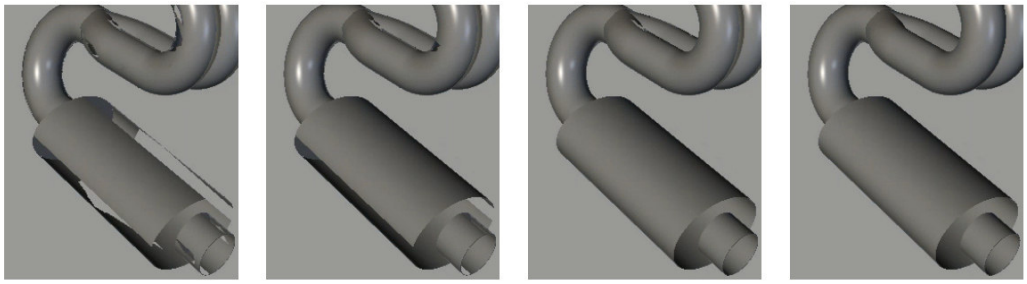
В общем случае для глобально заданного значения $(\hat{u},\hat{v}) \in [0,1]^2$ начальное приближение конкретного участка можно получить линейной интерполяцией граничных значений параметров:

$$(u_0, v_0) = \big(u_{min} + \hat{u} \cdot (u_{max} - u_{min}), v_{min} + \hat{v} \cdot (v_{max} - v_{min})\big). \quad (2)$$

Для первого кадра естественным выбором является использование параметрических центров: значения $(\hat{u}_0,\hat{v}_0) = (\frac{1}{2},\frac{1}{2})$. Таким образом формируется базовое покрытие изображения. Для каждого следующего кадра начальное приближение выбирается случайно, используя равномерное распределение:

$$(\hat{u}_i,\hat{v}_i) \sim U([0,1] \times [0,1]), i \geq 1. \quad (3)$$

Остается задействовать информацию с предыдущих кадров. Для этого используется буфер глубины, в котором записаны значения расстояния до точки пересечения в каждом пикселе изображения. Для каждого пикселя в текущем кадре цвет изменяется только в том случае, если найденная точка пересечения ближе к камере, чем предыдущий результат. В результате чего в каждом пикселе будет рассматриваться ближайшая найденная на текущий момент времени точка пересечения. Таким образом, предложенный метод при неизменной позиции камеры с течением времени позволяет получить корректное изображение (рис. 2), даже если трассировка лучей не всегда гарантирует корректный выбор начального приближения.



1-й кадр 3-й кадр 7-й кадр 20-й кадр

Рисунок 2. Демонстрация работы предложенного метода с первого по 20-й кадр

2. Метод усечения на основе двумерных SDF-текстур

SDF (англ. Signed Distance Field) представляет собой скалярное поле $\Phi(x)$: $\mathbb{R}^n \rightarrow \mathbb{R}$, значение которого в точке по модулю равно расстоянию до границы объекта и положительно или отрицательно, если точка находится снаружи или внутри объекта соответственно. В компьютерной графике наиболее часто используются двумерные (n = 2) и трехмерные (n = 3) поля расстояний. Для практического применения поле расстояния часто дискретизируют и сохраняют в виде текстуры. Такой метод используется в [11] для рендеринга текста. В предложенном методе SDF-текстуры используются в рамках усечения поверхности, где SDF может служить универсальным представлением для описания сложных границ обрезки.

Генерация текстур обрезки на основе полей расстояний со знаком (SDF) начинается с создания высокоточной бинарной маски, которая служит основой для последующих вычислений. Затем вычисляется расстояние до ближайшего пиксела с другим значением маски (1, если текущий пиксел 0, или 0, если текущий пиксел 1). На финальном этапе обработки выполняется квантование расстояний в 8-битное представление с последующей битовой упаковкой в 32-битной текстуре с целью сокращения затрат памяти. Использование предложенного метода вносит незначительные изменения в алгоритм рендеринга: вместо запроса к kd-дереву над кривыми обрезки производится выборка из SDF-текстуры с использованием билинейной интерполяции (листинг 1). Данная техника позволяет хранить информацию об усеченной области в сжатом виде без существенной потери качества (рис. 3).

```
1 def isTrimmed(surface, uv):
2   trimKdTree ←
      surface.trimKdTree
3   return trimKdTree.query(uv)
4
```

```
1 def isTrimmed(surface, uv):
2   trimSDF ← surface.trimSDF
3   distance ← trimSDF.sampleBilinear(uv)
4   return distance < 0
5
```

Листинг 1: Сопоставление алгоритмов тримминга в базовом (слева, [7]) и предложенном (справа) методах. В предложенном методе вычислительно сложная операция поиска в kd-дереве заменяется на одну текстурную выборку с билинейной интерполяцией значений.

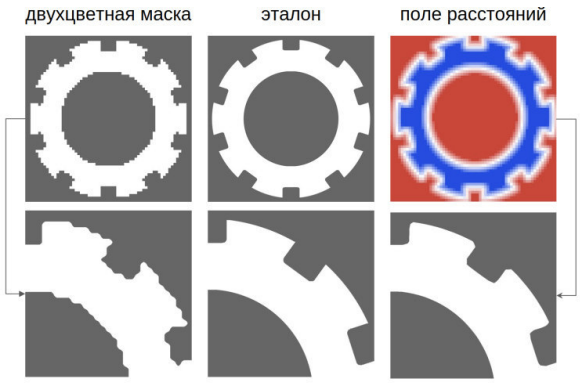


Рисунок 3. Сравнение представления трафарета через битовую маску и SDF-текстуру (64x64, 8 бит на пиксель) при одинаковом разрешении текстуры

2.1. Оптимальное распределение памяти для SDF-текстур

Наиболее важным параметром, определяющим качество усечения, является разрешение SDF-текстур. В силу ограниченности ресурсов необходимо распределить имеющуюся память под текстуры так, чтобы визуальное качество генерируемого изображения было оптимальным. Предложенный метод основывается на предположении, что большие поверхности будут занимать большую часть экранного пространства. Распределение памяти осуществляется по формуле

$$M_i = \left\lfloor M_{total} \times \frac{A_i}{\sum_{k=1}^{n} A_k} \right\rfloor, \quad (4)$$

где $i = \overline{1 \ldots n}$ – номер поверхности;

M_{total} – общее количество выделенной памяти для SDF-текстур в байтах;

A_k – оценка площади k-й поверхности;

M_i – количество выделенной памяти для SDF-текстуры i-й поверхности в байтах.

Рассматриваются квадратные SDF-текстуры $H_i \times H_i$, размер H_i которых вычисляется по формуле

$$H_i = \lfloor \sqrt{M_i} \rfloor. \quad (5)$$

2.2. Визуализация ребер и границ

Использование SDF-текстур открывает возможность отображения ребер и границ объектов без существенных накладных расходов (рис. 4). Поскольку расстояние до границы уже записано в SDF-текстуре, для определения принадлежности точки к границе используется условие

$$p \in \delta\Omega \approx \Phi\Omega(p) < \epsilon, \quad (6)$$

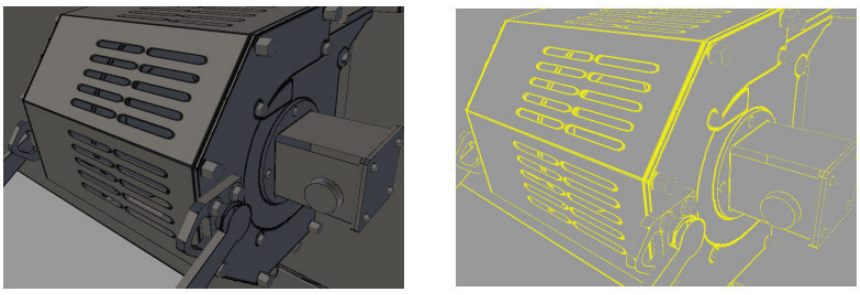
где ϵ – пороговое значение, определяющее толщину линии.

Для плавного перехода можно использовать интерполяцию:

$$color_{final} = \alpha \cdot color_{edge} + (1 - \alpha) \cdot color_{surface}, \quad (7)$$

где α – функция от Φ(p), обеспечивающая гладкий переход.

Примеры использования SDF-текстур для реализации данных режимов отображения приведены на рисунке 4.



а) *б*)

Рисунок 4. Демонстрация визуализации ребер с помощью предложенного метода: *а* – отображение с границами; *б* – только видимые границы

Результаты и сравнение

Для оценки предложенных методов были выбраны две модели из открытого набора данных ABC [12] и три модели из работы [7], которые также находились в открытом доступе. Основные параметры используемых моделей представлены в таблице 1. B-сплайновые кривые и поверхности соответствуют исходным данным модели, а поверхности и монотонные кривые Безье – полученным результатам после преобразований.

Таблица 1. **Характеристики тестовых моделей**

Модель	NURBS Поверхности	NURBS Кривые	Поверхности Безье	Кривые Безье	Потребление памяти (МБ)
ABC_687231	27306	139526	99721	460252	116.3
ABC_83870	92458	498024	238374	787783	210.0
Robot	4292	21702	5786	30862	27.8
Eiffel	14774	132926	16626	157914	38.9
Egoist	22363	113509	102233	328927	126.1

ABC_687231 ABC_83870 Robot Eiffel Egoist

1. Метод выбора начального приближения

Для метода стохастического выбора начального приближения численный эксперимент показал улучшение точности со временем (табл. 2, рис. 2). Кроме того, он показал, что в среднем 10 кадров достаточно, чтобы убрать артефакты, существенно ухудшающие качество изображения (средний PSNR на 10-м кадре >35 для всех моделей). Таким образом, в случае визуализации реального времени (более 25 кадров в секунду) данный процесс будет занимать доли секунды.

2. Метод тримминга поверхностей при помощи SDF-текстур

В качестве эталона рассматриваются изображения, получаемые использованием базового метода на основе kd-деревьев с высоким уровнем разбиения поверхностей на участки малой кривизны. Мы исследовали, как меняется точность предложенного метода при различном разрешении SDF-текстур (табл. 3).

По полученным в ходе экспериментов данным можно сказать, что SDF-текстуры демонстрируют качественный прирост производительности по сравнению с kd-деревом при сохранении приемлемого качества визуализации (PSNR >30 на всех дистанциях) даже при двукратном уменьшении объема используемой памяти. Рисунок 6 демонстрирует визуальное сопоставление.

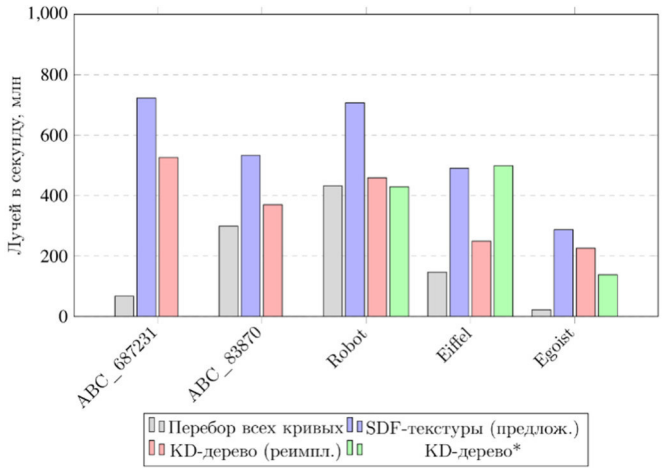


Рисунок 5. Сравнение производительности базового и предложенного методов.
Графический ускоритель аналогичен тому, что использовался в статье [7].
*Значения, продемонстрированные в статье [7]

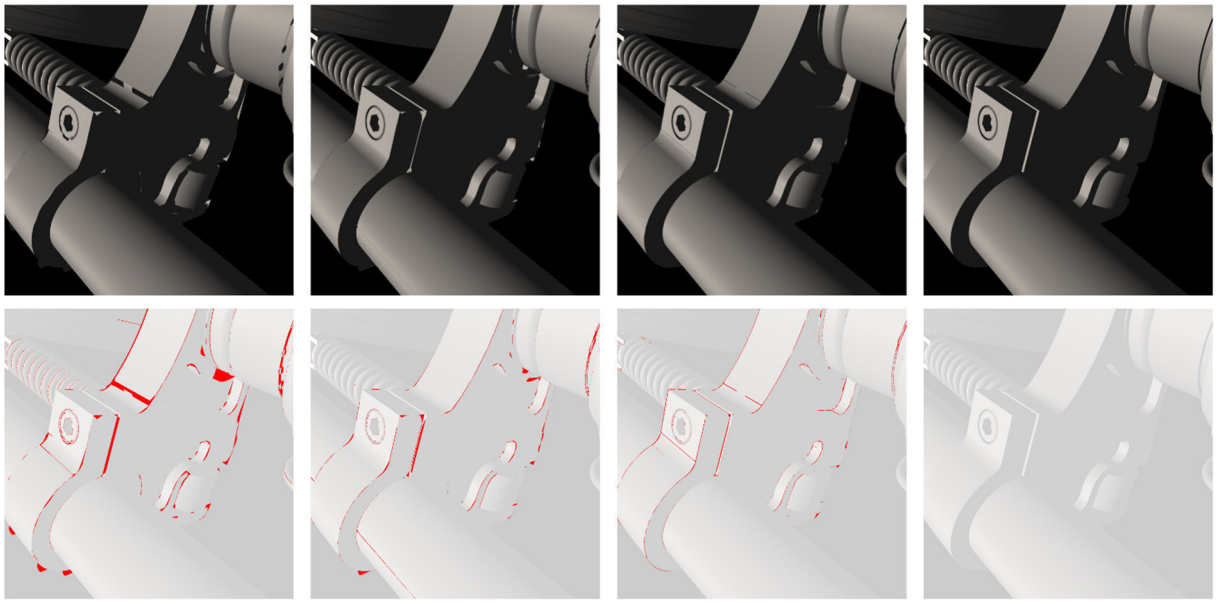


Рисунок 6. Разница изображений, генерируемых при помощи SDF-текстур, с эталоном при различном потреблении памяти.

Таблица 2. **Качество изображений на 10-м кадре с применением алгоритма подавления артефактов**

Модель	кадр 1	кадр 10
ABC_687231	20.3	40.2
ABC_83870	28.7	39.9
Robot	22.5	38.4
Eiffel	25.9	35.7
Egoist	21.1	45.5

Таблица 3. **Точность визуализации при помощи SDF-текстур по метрике PSNR при различных затратах памяти на данные для тримминга (относительно базового метода)**

Модель	50%	100%	200%	400%
ABC_687231	46.4	46.9	49.2	51.4
ABC_83870	32.2	36.5	38.1	41.7
Robot	34.2	39.4	40.7	44.8
Eiffel	32.0	32.6	37.0	37.2
Egoist	43.5	45.5	48.2	51.0

Заключение

В работе предложены два метода улучшения рендеринга B-сплайновых поверхностей. Метод стохастического выбора начального приближения эффективно устраняет артефакты алгоритма Ньютона в течение нескольких кадров при статичной камере. Метод на основе SDF-текстур демонстрирует значительное ускорение рендеринга (в 1.5-2 раза) при регулируемых уровне качества и потреблении памяти, а также дополнительно предоставляет возможность рендеринга рёбер объектов. Оба метода показывают хорошую эффективность при работе на системах с ограниченными ресурсами.

Таким образом, предложенные методы позволяют повысить точность и скорость визуализации сложных NURBS-моделей на ограниченных по ресурсам системах, что делает их перспективными для применения в системах интерактивного предпросмотра и CAD-приложениях.

Для визуализации в условиях изменяющейся камеры или движения моделей перспективным направлением представляется использование репроекции данных с предыдущих кадров для инициализации поиска пересечений в новых условиях наблюдения. Альтернативный подход может заключаться в комбинации попиксельного стохастического выбора начальных приближений с алгоритмами шумоподавления. Реализация указанных улучшений позволит расширить область применения предложенных методов.

Список литературы

1. "Opencascade3d", 2025. Available: https://dev.opencascade.org/. [Accessed 14 5 2025].

2. R. Xiong, Y. Lu, C. Chen, J. Zhu, Y. Zeng and L. Liu, "Eter: Elastic tessellation for real-time pixel-accurate rendering of large-scale nurbs models", *ACM Transactions on Graphics,* No. 2023, pp. 1-13.

3. T. Nishita, T. W. Sederberg and M. Kakimoto, "Ray tracing trimmed rational surface patches", *Proceedings of the 17th annual conference on Computer graphics and interactive techniques,* p. 337–345, 1990.

4. W. Martin, E. Cohen, R. Fish and P. Shirley, "Practical ray tracing of trimmed nurbs surfaces", *Journal of Graphics Tools,* p. 27–52, 2000.

5. A. Efremov, V. Havran and H.-P. Seidel, "Robust and numerically stable bezier clipping method for ray tracing nurbs surfaces", *Proceedings of the 21st Spring Conference on Computer Graphics.*

6. A. Schollmeyer and B. Froehlich, "Efficient and anti-aliased trimming for rendering large nurbs models", *IEEE Transactions on Visualization and Computer Graphics,* p. 1489–1498, 2019.

7. J. Sloup and V. Havran, "Optimizing ray tracing of trimmed nurbs surfaces on the GPU", *Computer Graphics Forum,* p. 161–172, 2021.

8. D. Zhdanov, I. Potemin and A. Zhdanov, "Using embree technologies for ray tracing in optical systems with free-form surfaces", *Proceedings of the International Conference on Computer Graphics and Vision "Graphicon",* vol. 33, p. 97–107, 2023.

9. K. Jankauskas, "Time-efficient nurbs curve evaluation algorithms", *Information Technologies 2010: proceedings of the 16th international conference on Information and Software Technologies,* 2010.

10. L. Piegl and B. Tiller, The NURBS Book, 2 ed., Springer, 1997.

11. C. Green, "Improved alpha-tested magnification for vector textures and special effects," *ACM SIGGRAPH 2007 courses,* 2007.

12. S. Koch, A. Matveev, Z. Jiang, F. Williams, A. Artemov, E. Burnaev, M. Alexa, D. Zorin and D. Panozzo, "ABC: A big cad model dataset for geometric deep learning", *The IEEE Conference on Computer Vision and Pattern Recognition,* 2019.