Формирование эффективной пространственной структуры фотонных карт для ускорения процесса рендеринга

Р.Р. Халимов [1], Д.Д. Жданов [1], А.Д. Жданов [1]

[1] *Университет ИТМО, Кронверкский проспект д.49, литера А, Санкт-Петербург, 197101, Россия*

Аннотация

Проведен анализ методов реалистичного рендеринга с точки зрения эффективности расчета вторичного и каустического освещений. Предложен метод двунаправленной стохастической трассировки лучей с использованием фотонных карт, обеспечивающий высокую эффективность расчета каустического освещения. Рассмотрены основные подходы использования метода фотонных карт: построение фотонных карт на трассах прямых лучей и на трассах обратных лучей. Выявлены преимущества и недостатки данных методов. В качестве базового решения выбран метод двунаправленной стохастической трассировки лучей с использованием обратных фотонных карт. На базе программного комплекса Lumicept проведено профилирование данного решения. Выявлены основные проблемы, связанные с замедлением процесса рендеринга. Для большинства сцен половина времени рендеринга тратилась на обработку запросов к фотонной карте: поиск близлежащих фотонов в пределах ячейки пространственной структуры и поиск пересечения луча с интегрирующими сферами фотонов. Был проанализирован ряд решений для пространственного разбиения фотонных карт: регулярная решетка, регулярная решетка с хэш таблицей и бинарное дерево. Выявлены преимущества и недостатки рассматриваемых решений и предложено комбинированное решение, объединяющее адаптивность разбиения бинарного дерева и эффективность доступа, обеспечиваемая хэш таблицами. В программном комплексе Lumicept было реализовано комбинированное решение, что позволило повысить общую эффективность рендеринга на 30%.

Ключевые слова

Реалистичный рендеринг, двунаправленная трассировка лучей, бинарное дерево, хэш таблица, фотонные карты.

Creation of an Effective Spatial Structure of Photon Maps to Speed up the Rendering Process

R.R. Khalimov [1], D.D. Zhdanov [1], A.D. Zhdanov [1]

[1] *ITMO University, Kronversky prospect 49, liter A, Saint-Petersburg, 197101, Russia*

Abstract

The analysis of realistic rendering methods from the point of view of the efficiency of calculation of secondary and caustic illumination is carried out. A method of bidirectional stochastic ray tracing using photonic maps is proposed, which ensures high efficiency of calculation of caustic illumination. The main approaches of using the photon maps method are considered: the construction of photon maps on the traces of forward rays and on the traces of backward rays. The advantages and disadvantages of these methods are revealed. The method of bidirectional stochastic ray tracing using backward photon maps is chosen as the basic solution. Profiling of this solution was carried out based on the Lumicept software package. The main problems associated with slowing down the rendering process have been identified. For most scenes, half of the rendering time was spent processing requests to the photon map: searching for nearby photons within the cell

ГрафиКон 2022: 32-я Международная конференция по компьютерной графике и машинному зрению, 19-22 сентября 2022 г., Рязанский государственный радиотехнический университет им. В.Ф. Уткина, Рязань, Россия
EMAIL: khalimov.ruslan@mail.ru (Р.Р. Халимов); ddzhdanov@mail.ru (Д.Д. Жданов); andrew.gtx@gmail.com (А.Д. Жданов)
ORCID: 0000-0003-1923-4360 (Р.Р. Халимов); 0000-0001-7346-8155 (Д.Д. Жданов); 0000-0002-2569-1982 (А.Д. Жданов)

of the spatial structure and searching for the intersection of the ray with the integrating photon spheres. Several solutions for spatial partitioning of photonic maps were analyzed: a regular mesh, a regular mesh with a hash table and a binary tree. The advantages and disadvantages of the solutions under consideration are revealed and a combined solution is proposed that combines the adaptability of splitting a binary tree and the access efficiency provided by hash tables. A combined solution has been implemented in the Lumicept software package, which will increase the overall rendering efficiency by 30%.

Keywords
Realistic rendering, bidirectional ray tracing, binary tree, hash table, photonic maps.

1. Введение

В современном мире можно видеть всё больше изображений, полученных в результате моделирования различных объектов и сцен. Задачи реалистичного рендеринга широко востребованы для компьютерного моделирования оптически сложных сцен; создания обучающих датасетов для систем компьютерного зрения; технологий дополненной, виртуальной и смешанной реальностей; кинематографа; компьютерных игр и т. д. При этом со временем растут и требования к реалистичности генерируемых изображений.

На сегодняшний день существует большое количество различных методов реалистичной визуализации трехмерных сцен, позволяющих за конечное время вычислить корректные значения яркости, основываясь на физически корректных законах распространения света. Каждый метод обладает своими преимуществами и недостатками, и невозможно определить универсальный для всех случаев метод [1]. Например, для расчета каустического освещения наиболее эффективными являются методы фотонных карт.

Некоторые области применения реалистичной визуализации определяют жесткие требования как к реалистичности, так и к скорости рендеринга. Часто в таких случаях приходится отказываться от визуализации сложных эффектов в пользу более быстрого рендеринга изображений. Поэтому ускорение вычислений означает не только уменьшение времени рендеринга, но и получение изображений с более сложными эффектами за то же время. Также ускорение рендеринга предоставляет возможность получать более качественные изображения на устройствах с меньшими вычислительными ресурсами. В данном исследовании основной упор делался на методы фотонных карт, которые несмотря на смещенность результата могут обеспечить физическую корректность (при грамотном выборе точки и радиуса сферы интегрирования) и эффективность вычислений.

В работе [2] был предложен алгоритм вычисления глобальной освещенности с помощью метода прямых фотонных карт. Алгоритм состоял из трех шагов:

1. Трассировка фотонов.
2. Построение фотонных карт.
3. Сбор освещенности.

Алгоритм трассировки фотонов аналогичен алгоритму трассировки путей. При попадании на поверхность сцены событие, происходящее с фотоном (зеркальное или диффузное отражение, преломление или поглощение) определяется в зависимости от свойств материала при помощи метода русской рулетки [3].

Когда фотон попадает на диффузную поверхность, информация о нем сохраняется в фотонной карте. Для получения хорошего качества изображения необходима трассировка и хранение большого количества фотонов. Поэтому для хранения фотонных карт используют специальные ускоряющие структуры пространственного разбиения, которые позволяют оптимизировать поиск фотонов, попадающих в сферу с центром в произвольной точке пространства. Этап построения фотонных карт является вспомогательным для следующего шага.

На последнем этапе выполняется трассировка путей. Как показано на рисунке 1 при первом попадании луча на диффузную поверхность производится сбор освещенности [4]. Видимая яркость i-ой точки экрана $L_i(\vec{x}, \vec{v}_r)$ в окрестности сферы интегрирования радиуса r рассчитывается по следующей формуле:

$$L_i(\vec{x},\vec{v}_r) = \frac{1}{\pi^2 r^2}\sum_{p=1}^{k} BRDF(\vec{x},\vec{v}_r,\vec{v}_{r,p})\Delta F(\vec{x},\vec{v}_{r,p}), \quad (1)$$

где $BRDF(\vec{x},\vec{v}_r,\vec{v}_{r,p})$ – значение ДФР в точке $\vec{x}$ для направления падения света p-го фотона $\vec{v}_{r,p}$ и направления наблюдения $\vec{v}_r$, $\Delta F(\vec{x},\vec{v}_{r,p})$ – поток, переносимый одним фотоном в точку $\vec{x}$ в направлении $\vec{v}_{r,p}$.

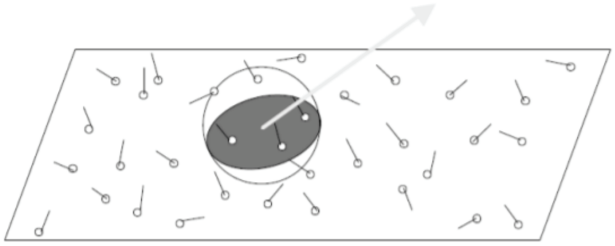

Рисунок 1 – Сбор освещенности с фотонной карты по сфере интегрирования

Поиск фотонов может выполняться одной из двух стратегий:

- Поиск k ближайших фотонов (динамический радиус сбора).
- Поиск всех фотонов в заданном радиусе (фиксированный радиус сбора).

При использовании фиксированного радиуса его значение выбирается пропорционально расстоянию от наблюдателя до точки поверхности. Такой подход упрощает процесс сбора, однако не обеспечивает постоянное количество фотонов в сфере интегрирования. Из-за этого он хуже работает при большом разбросе распределения фотонов по сцене, так как дает плохие оценки в областях с небольшим количеством фотонов и размытые оценки в областях с высокой плотностью фотонов.

Метод прямых фотонных карт имеет ряд существенных недостатков. Он может создавать распределение фотонов вне области наблюдения, может создавать большую плотность фотонов вблизи источника света и не имеет четкого критерия по числу фотонов, которые надо выпустить источникам света. Эти недостатки отсутствуют в методе обратных фотонных карт.

В методе обратных фотонных карт [5] фотоны испускаются из камеры, формируя область виденья, то есть область, которая формирует яркость для конкретного положения наблюдателя. При попадании на диффузную поверхность в фотонной карте сохраняется не только информация о фотоне, но и радиус сферы интегрирования. Это возможно благодаря тому, что на этапе трассировки фотонов известно расстояние от наблюдателя до точки наблюдения.

Принципиальное отличие между методами прямых и обратных фотонных карт заключается в методе сбора освещенности. Метод сбора проиллюстрирован на рисунке 2.

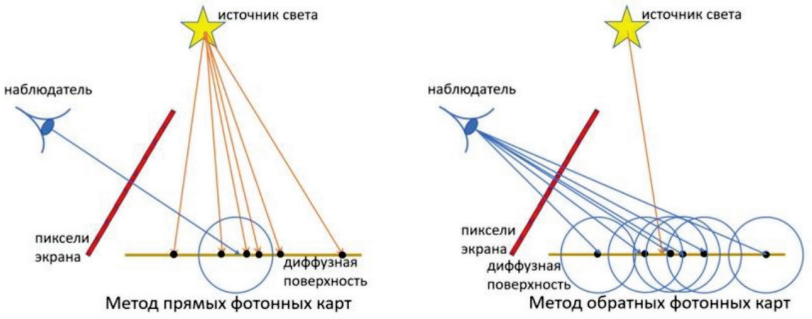

Рисунок 2 – Сбор освещенности с фотонной карты по сфере интегрирования

На этапе сбора освещенности трассируются прямые лучи, испускаемые источником света. При попадании луча на поверхность по обратной фотонной карте производится поиск всех сфер интегрирования, пересекаемых данным лучом. Энергия фотона передается всем направлениям наблюдения, соответствующим найденным сферам интегрирования. Таким образом i-й фотон формирует вклад в яркость нескольких пикселей изображения:

$$L_1(\vec{x}_1, \vec{v}_{1,r}) = \frac{1}{\pi^2 r_1^2} BRDF(\vec{x}_1, \vec{v}_{1,r}, \vec{v}_{p,i}) \Delta F(\vec{x}_1, \vec{v}_{p,i}) \quad (2)$$

$$\dots$$

$$L_k(\vec{x}_k, \vec{v}_{k,r}) = \frac{1}{\pi^2 r_k^2} BRDF(\vec{x}_k, \vec{v}_{k,r}, \vec{v}_{p,i}) \Delta F(\vec{x}_k, \vec{v}_{p,i}).$$

Если осуществить суммирование по всем прямым лучам, то результат, получаемый методами прямых и обратных фотонных карт, будет одинаков. Вопрос только в эффективности вычислений. Для повышения эффективности вычислений и снижения ошибки, вызванной конечным значением радиуса сферы интегрирования, метод прямых фотонных карт использует так называемый финальный сбор [6]. Аналогичные решения применяются и в методе обратных фотонных карт, например, выбор оптимальной стратегии интегрирования яркости вторичного освещения (выбор глубины интегрирования на диффузной трассе луча [7] и весовых коэффициентов для яркости вторичного освещения с различных точек интегрирования [8], выбор оптимального числа обратных фотонов [9]).

Для получения изображения высокого качества может потребоваться трассировка и хранение миллиардов фотонов. Такой объем данных является существенным недостатком всех методов фотонных карт. Одним из способов решения данной проблемы является применение метода стохастических прогрессивных фотонных карт [10, 11, 12]. Процесс формирования фотонных карт разбивается на фазы, каждая из которых является законченным расчетом яркости всех компонент освещения сцены. При этом радиус сфер интегрирования уменьшается в процессе перехода от одной фазы к другой.

$$\frac{r_{i+1}^2}{r_i^2} = \frac{N_i + \alpha M_i}{N_i + M_i}, \quad (3)$$

где N_i – число уже собранных до i -ой итерации фотонов, M_i – число трассируемых на i-ой итерации фотонов, α – заданный в алгоритме параметр со значением от 0 до 1.

Такое решение позволяет уменьшить объем памяти для хранения фотонных карт и ускорить процесс их создания и обработки. Однако даже методы стохастических прогрессивных фотонных карт (прямых и обратных) имеют существенные недостатки, связанные со скоростью обработки запросов к данным фотонных парт при поиске пересечения лучей с фотонами карт. Для повышения производительности обработки запросов к фотонным картам используются ускоряющие структуры, оптимальному формированию которых посвящено данное исследование.

2. Структуры пространственного разбиения

Как уже было сказано, для эффективного использования методов фотонных карт необходимо хранить большое количество фотонов, а также обеспечить быстрый поиск всех фотонов, попадающих в сферу интегрирования. Для решения этих задач используются различные ускоряющие структуры пространственного разбиения. Основными методами являются – регулярное разбиение пространства и структуры на основе бинарных деревьев.

2.1. Регулярное разбиение пространства

Одной из самых простых структур является разбиение всей области пространства на ячейки одинакового размера [13]. При построении данной структуры заранее выбирается детальность разбиения. Область пространства, ограничивающая все фотоны, разбивается на n1*n2*n3 вокселей. В каждой ячейке хранится список фотонов, попавших в неё. Пример разбиения двумерного пространства представлен на рисунке 3.

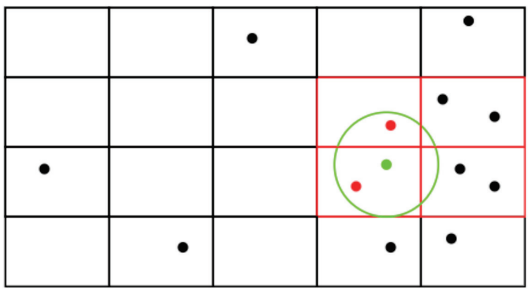

Рисунок 3 – Разбиение пространства регулярной сеткой

При поиске всех фотонов, попадающих в сферу интегрирования, сначала необходимо найти ячейку, в которой содержится точка пересечения луча с поверхностью. Далее составляется список всех ячеек, которые пересекаются со сферой. Все фотоны, хранящиеся в этих ячейках, потенциально могут принадлежать сфере интегрирования. Поэтому необходимо проверить расстояние от центра сферы до каждого такого фотона. Таким образом, структура регулярного разбиения пространства позволяет значительно сократить список проверяемых фотонов: от всех фотонов сцены до фотонов в некоторой локальной области.

Так как размеры всех ячеек равны, а также известно положение регулярной сетки в пространстве, то по координатам точки легко определить, какой ячейке она принадлежит. Благодаря этому для хранения данной структуры можно использовать хеш-таблицу. Ключом будет являться тройка из порядковых номеров ячейки по каждой оси. При этом достаточно хранить только те ячейки, которые содержат хотя бы один фотон, что сокращает объем используемой памяти. Использование хеш-таблицы позволяет быстро находить нужный воксель, поэтому одним из преимуществ данного метода является скорость работы.

Стоит отметить, что период пространственной структуры фиксирован и выбирается заранее. Выбор слишком большого периода приводит к тому, что в ячейки будет попадать большое количество фотонов, а значит на этапе поиска необходимо проводить больше вычислений и сравнений расстояния от центра сферы интегрирования до фотона. В то же время слишком мелкое разбиение требует больший объем памяти для хранения хеш-таблицы, а значит замедление запросов к ней. Таким образом, к недостаткам данного подхода можно отнести плохую адаптивность разбиения.

2.2. Бинарные деревья

Kd-дерево [14] – одно из наиболее популярных пространственных структур для эффективного поиска ближайших фотонов. Оно представляет собой бинарное дерево, каждый узел которого соответствует одному вокселю пространства. При этом ячейка родительского узла является объединением двух дочерних. Пример разбиения фотонов в двумерном пространстве проиллюстрирован на рисунке 4.

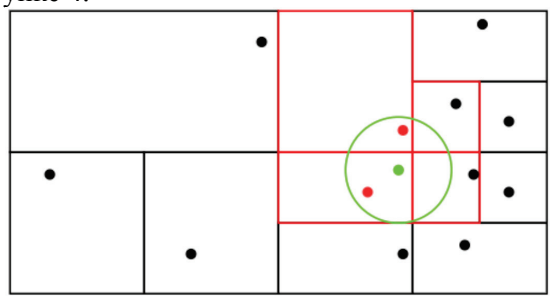

Рисунок 4 – Разбиение пространства с помощью kd-дерева

Построение kd-дерева начинается с ограничения всех фотонов сцены параллелепипедом, которому будет соответствовать корневой узел дерева. Далее для каждого узла дерева повторяются следующие шаги:

1. Выбирается секущая плоскость. Для этого необходимо выбрать ось, по которой будет разделен параллелепипед, и точку, через которую будет проведена секущая плоскость.
2. Через выбранную точку проводится секущая плоскость, перпендикулярная выбранной оси координат. Тогда все фотоны, которые оказались слева от плоскости, относятся к левому поддереву текущего узла. Остальные фотоны относятся к правому поддереву.
3. Если текущий узел содержит некоторое заранее выбранное минимальное количество фотонов, то дальнейшее разбиение прекращается.
4. Для каждого поддерева рекурсивно выполняются шаги 1, 2 и 3.

Существует ряд способов разбиения пространства: разбиение по середине самого большого по размеру вокселя, выбор плоскости разбиения таким образом, чтобы в левом и правом поддеревьях оказалось одинаковое число фотонов [15], более сложные алгоритмы, использующие оценки времени доступа к элементу дерева. Кроме того, бинарное дерево может строиться как для центров фотонов, так и для сфер интегрирования. В последнем случае доступ к фотонам ячейки дерева будет выполняться в разы быстрее, поскольку будет отсутствовать операция перехода от ячейки к ячейке, что очень дорого для бинарного дерева.

Главным преимуществом использования kd-дерева для хранения фотонных карт является хорошая адаптивность разбиения. Благодаря этому учитывается неравномерность распределения фотонов по сцене. Однако при больших размерах дерева поиск необходимых ячеек занимает больше времени по сравнению со структурой регулярного разбиения. К тому же практическая реализация бинарного дерева сталкивается с проблемой медленного доступа к памяти вычислительного устройства. При каждом переходе из очередного узла в дочерний требуется длительное полноценное обращение к оперативной памяти, а процессорный кэш только замедляет эту операцию, так как дерево хранится в памяти неравномерно и соседние ячейки не попадают в одну кэш-линию. Особенно эта проблема проявляется в многопоточном режиме, когда большое число потоков одновременно запрашивают данные фотонной карты. Таким образом, при большом размере фотонной карты переходы по дереву могут занимать значительную часть процесса поиска в дереве.

3. Анализ скорости доступа к фотонной карте при различном способе хранения фотонов

В данном исследовании разбиение пространства происходило для сфер интегрирования. Такое разбиение обеспечивает более быстрый доступ к данным фотонов (особенно в случае бинарного дерева), поскольку при попадании луча в ячейку отсутствует необходимость поиска и опроса данных соседних ячеек, которые могут быть затронуты сферой интегрирования.

При реализации структуры регулярного разбиения использовалась одинаковая детальность разбиения по всем осям. Все трассированные фотоны ограничиваются прямоугольным параллелепипедом минимального размера с гранями, параллельными осям координат. Затем он делится на $N * N * N$ вокселей. Для хранения получившейся структуры используется хеш-таблица, ключом в которой является тройка $< i, j, k >$ – порядковые номера ячейки по каждой из осей.

Алгоритм построения регулярной структуры пространственного разбиения следующий:

1. Генерация ограничивающей все фотоны ячейки $boundingBox$.
2. Расчет размера одной ячейки разбиения (детальность разбиения, которая задается как параметр структуры).
3. Создание $N * N * N$ ячеек размера $voxelSize$, которые в совокупности покрывают ограничивающий воксель. Все созданные ячейки сохраняются в хеш-таблицу.
4. Для каждого фотона производится поиск ячеек, пересекающихся с соответствующей фотону сферой. Данные о фотоне сохраняются в каждую такую ячейку.

Для поиска фотонов, попадающих в сферу интегрирования, применяется следующий алгоритм:

1. По координатам запрашиваемой точки вычисляются порядковые номера – $< i,j, k >$, по которым определяется ячейка, содержащая в себе заданную точку.

2. Для каждого фотона, содержащегося в списке найденной ячейки, проверяется расстояние до заданной точки.
3. Фотоны, находящиеся не дальше радиуса сферы, добавляются в результирующий список.

Генерация фотонов проводилась двумя способами. Первый – равномерное распределение координат от -500 до 500. Второй – нормальное распределение координат с математическим ожиданием 0 и среднеквадратичным отклонением 20. Координаты при нормальном распределении масштабировались так, чтобы минимальное и максимальное значение получалось таким же, как и при равномерном распределении. Радиус сферы фотона выбирался случайно в диапазоне от 2 до 5.

В тестировании участвовали структуры регулярного разбиения с детальностями разбиения 25, 50, 75, 100 и 125. К каждой из них было произведено 10 млн случайных запросов. Тестирование производилось на двух диапазонах количества фотонов – от 10000 до 100000 и от 100000 до 1000000. Результаты измерения скорости обработки запросов для равномерного распределения фотонов представлены на рисунке 5, для нормального распределения фотонов представлены на рисунке 6.

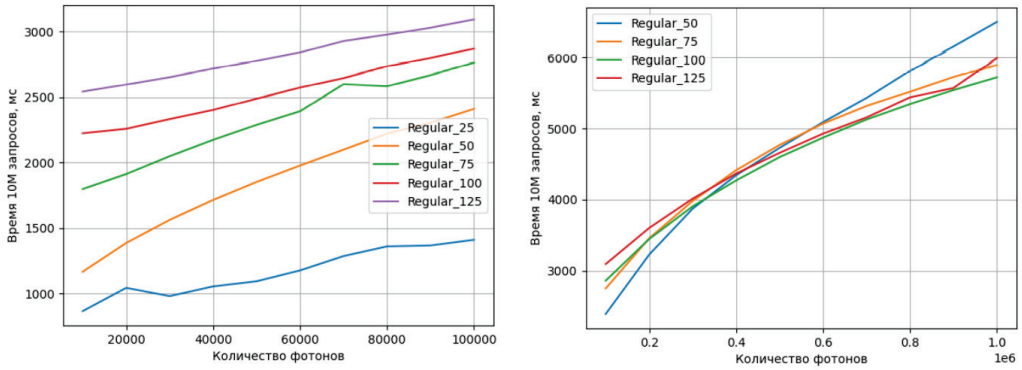

Рисунок 5 – Время выполнения запросов к структуре регулярного разбиения при равномерном распределении фотонов

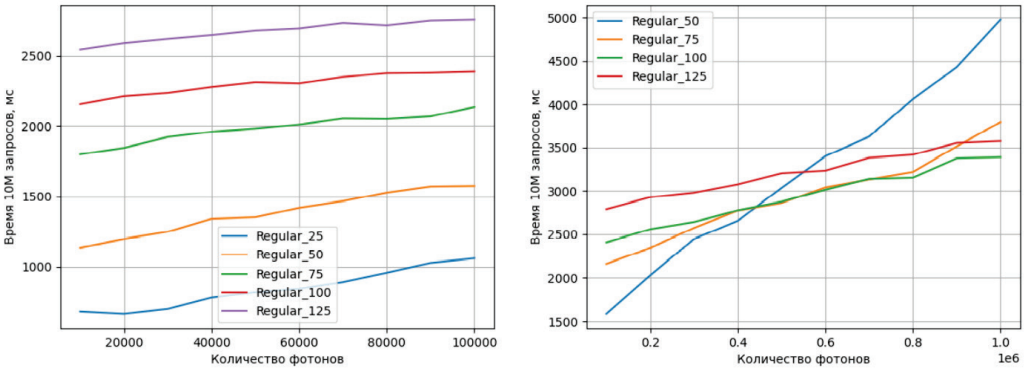

Рисунок 6 – Время выполнения запросов к структуре регулярного разбиения при нормальном распределении фотонов

По результатам измерений видно, что на небольшом количестве фотонов лучше работает более крупное разбиение пространства. А на большом количестве фотонов требуется более мелкое разбиение. При этом при слишком большом увеличении частоты разбиения (125 и более) структура начинает работать медленнее.

Для построения Kd дерева использовался алгоритм разбиения пространства на две равные части. Плоскость разбиения проходила поперек большей стороны ячейки и разбиение осуществлялось до тех пор, пока число фотонов в одной ячейке не достигало 10. Параметры тестирования были аналогичны равномерному разбиению фотонов. Результаты измерения скорости обработки запросов для равномерного распределения фотонов представлены на рисунке 7, для нормального распределения фотонов представлены на рисунке 8.

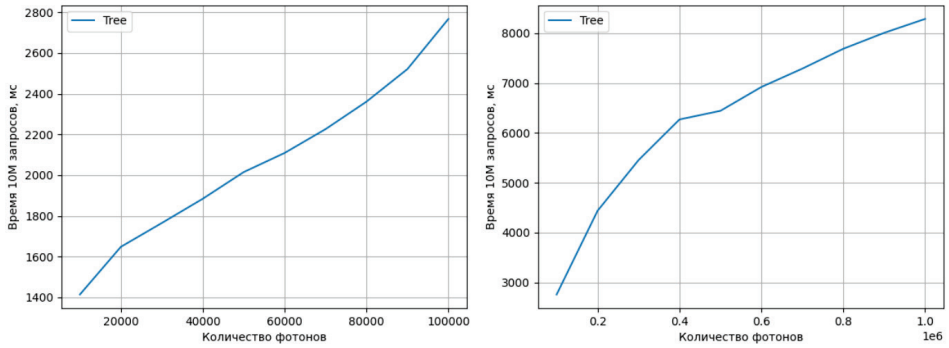

Рисунок 7 – Время выполнения запросов к структуре регулярного разбиения при равномерном распределении фотонов

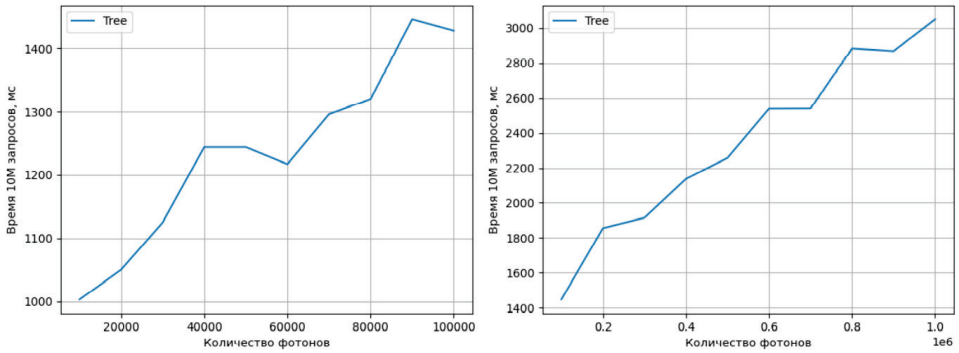

Рисунок 8 – Время выполнения запросов к структуре регулярного разбиения при нормальном распределении фотонов

Увеличение времени запроса при увеличении количества фотонов обусловлено как увеличивающимся количеством фотонов в одной терминальной ячейке, так и увеличением глубины дерева.

Оба метода пространственного разбиения имеют как преимущества, так и недостатки. В рамках исследования был предложен комбинированный метод, объединяющий преимущества обоих методов. Предложенный алгоритм пространственного разбиения с помощью kd дерева создает псевдо-регулярное разбиение пространства, в котором не все ячейки разбиты до нижнего уровня. Поэтому регулярная решетка может быть наложена на определенный уровень дерева, а для всех последующих уровней разбиения формируются небольшие «остаточные» kd деревья. При этом на уровне регулярного разбиения работает хэш таблица, что обеспечивает компактное хранение данных и быстрый доступ к «остаточному» дереву карты. Пример построения данной структуры при наложении на наиболее глубокий полный уровень kd дерева представлен на рисунке 9.

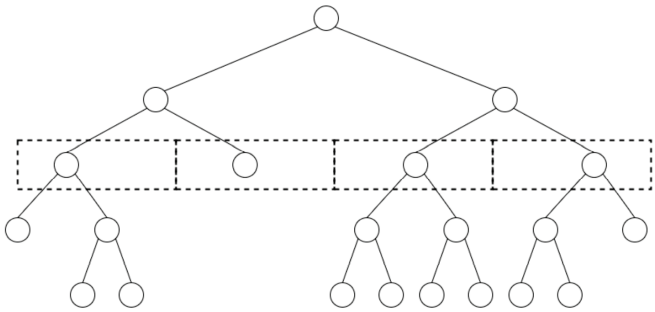

Рисунок 9 – Наложение хеш-таблицы на наиболее глубокий полный уровень kd дерева

Если хеш-таблица накладывается на заранее выбранный произвольный уровень дерева, то некоторые вершины дерева могут отсутствовать. Вместо таких вершин в хеш-таблице сохраняется ближайшая существующая родительская вершина. На рисунке 10 представлен пример, в котором хеш-таблица накладывается на последний уровень дерева.

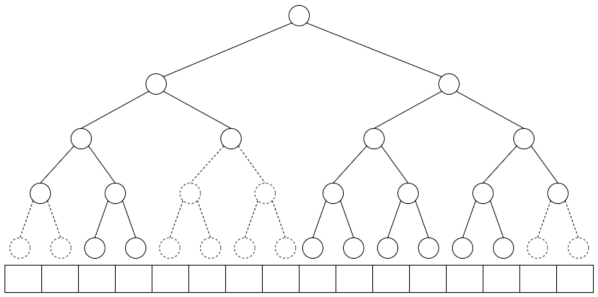

Рисунок 10 – Наложение хеш-таблицы на фиксированный уровень дерева

При использовании данного подхода важным является оптимальный уровень наложения хеш-таблицы. С одной стороны, чем глубже находится регулярное разбиение, тем больше вершин пропускается на этапе поиска фотонов и тем меньше вычислительных ресурсов занимает запрос к структуре. С другой стороны, растет объем используемой памяти. Например, при хранении в хеш-таблице указателей на вершины дерева и использовании 64-разрядной системы каждый из них будет занимать 8 байт. При наложении на 30 уровень дерева необходимо хранение 2^{30} таких указателей. Соответственно по сравнению с обычным kd-деревом в сумме требуется 8 гигабайт дополнительной памяти. Необходимость заранее выбирать оптимальный уровень наложения хеш-таблицы является одним из недостатков такого подхода.

Для более адаптивного выбора глубины дерева был реализован третий подход – адаптивное определение уровня наложения хэш таблицы:

1. Определение средневзвешенной глубины терминальных вершин.
2. Определение медианного значения глубины. Для этого в дереве сохраняется список всех терминальных вершин. После построения дерева список сортируется по глубине вершин. В качестве глубины наложения хеш-таблицы выбирается глубина вершины из середины списка. В результате тестирования на разном количестве фотонов от 10000 до 1000000 медианная глубина принимала значения в диапазоне от 11 до 21.

Для тестирования использовался аналогичный набор данных для следующих вариантов выбора уровня наложения хеш-таблицы:

- Наложение хеш-таблицы на наиболее глубоком полном уровне дерева (MinCombined).
- Наложение хеш-таблицы на глубине 10 (FixedCombined_10).
- Наложение хеш-таблицы на глубине 15 (FixedCombined_15).
- Наложение хеш-таблицы на глубине 20 (FixedCombined_20).
- Наложение хеш-таблицы на средневзвешенной глубине дерева (AvgCombined).
- Наложение хеш-таблицы на медианной глубине дерева (MedianCombined).

Тестирование производилось на двух диапазонах чисел фотонов – от 10000 до 100000 и от 100000 до 1000000. Результаты измерения скорости обработки запросов для равномерного распределения фотонов представлены на рисунке 11, для нормального распределения фотонов представлены на рисунке 12.

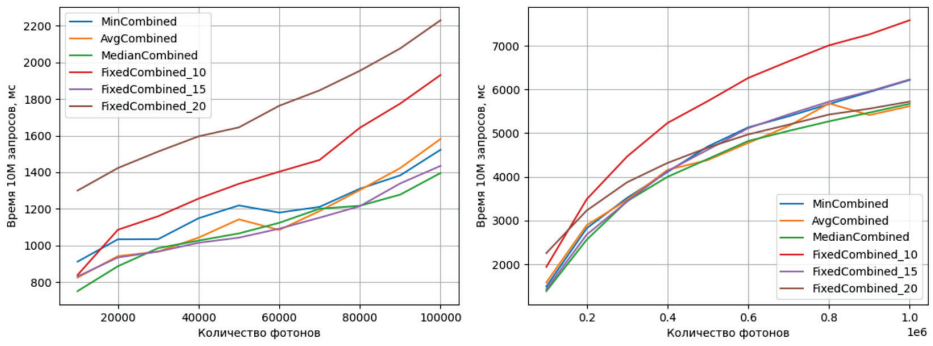

Рисунок 11 – Время выполнения запросов к структуре регулярного разбиения при равномерном распределении фотонов

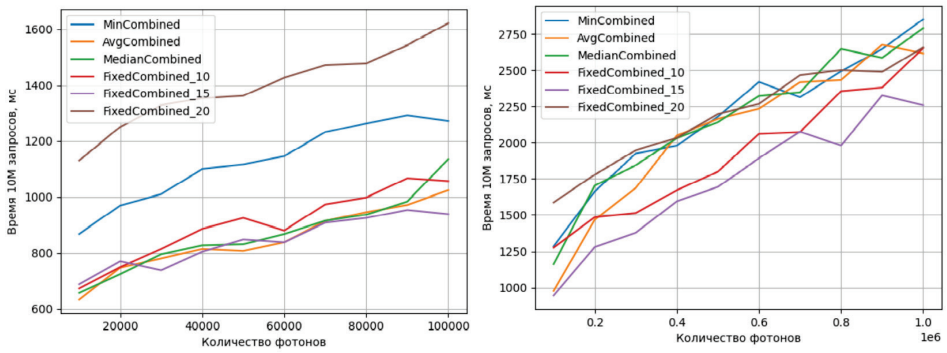

Рисунок 12 – Время выполнения запросов к комбинированной структуре при нормальном распределении фотонов

На всех вариациях числа и закона распределения фотонов можно выделить вариант наложения хеш-таблицы на глубине дерева 15 как оптимальный.

Далее было проведено исследование зависимости скорости обработки запросов от способа построения пространственной структуры. Для малого числа фотонов при равномерном и нормальном их распределении было выбрано регулярное разбиение с разрешением 25 и метод наложения хеш-таблицы на бинарное дерево на глубине 15. Для большого числа фотонов при равномерном и нормальном их распределении было выбрано регулярное разбиение пространства с разрешением 100 и метод наложения хеш-таблицы на бинарное дерево на глубине 15. Результаты измерения скорости обработки запросов для равномерного распределения фотонов представлены на рисунке 13, для нормального распределения фотонов представлены на рисунке 14.

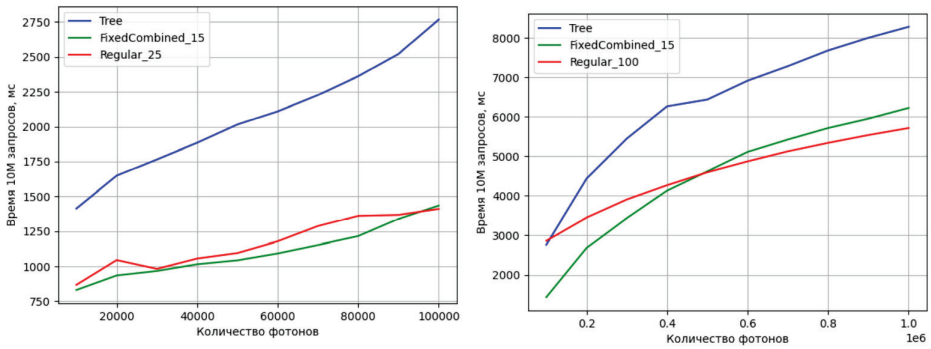

Рисунок 13 – Сравнение времени выполнения запросов к структурам пространственного разбиения при равномерном распределении фотонов

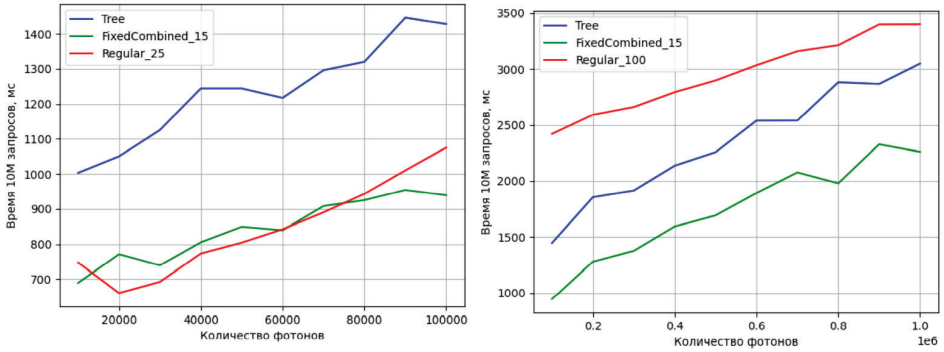

Рисунок 14 – Сравнение времени выполнения запросов к структурам пространственного разбиения при нормальном распределении фотонов

По результатам сравнения различных методов можно отметить, что в трех из четырех вариантов тестирования лучшую и примерно одинаковую производительность показывают регулярное разбиение пространства и комбинированный метод. Однако при неравномерном распределении большого числа фотонов регулярное разбиение показывает заметно худшие

результаты. Это можно объяснить его плохой адаптивностью. При большом скоплении фотонов в небольшой области пространства, в ячейке формируется длинный список фотонов для проверки попадания точки в сферу интегрирования, что сильно замедляет время работы структуры.

Также стоит отметить, что во всех случаях комбинированный подход показывает лучшие результаты по сравнению с обычным бинарным деревом. Следовательно, наложение хеш-таблицы на практике действительно оптимизирует работу структуры за счет сокращения количества переходов по дереву.

4. Результат тестирования комбинированного подхода в программном комплексе Lumicept

Lumicept [16] – это комплекс программ моделирования 3D-сцен, который для реализации физически корректного рендеринга использует метод двунаправленной стохастической трассировки лучей с использованием обратных фотонных карт. Использование графических процессоров в данном программном комплексе не предусмотрено. Однако поддерживаются распределенные вычисления за счет использования объединения ресурсов нескольких процессоров или компьютеров, подключенных по сети. В качестве оптимизации вычислений для каждого объекта сцены строится отдельная фотонная карта. В исследованиях было показано, что использование N фотонных карт с M фотонами в каждой эффективнее, чем использование одной фотонной карты с $N * M$ фотонами [17]. Для построения фотонной карты используется алгоритм kd дерева с разбиением большей стороны ячейки на две равные части.

Разработанный комбинированный подход был реализован в программном комплексе Lumicept и было произведено сравнение ускорения процесса рендеринга после замены текущей реализации на комбинированный подход. Сравнение происходило для трех типов сцен: классической Cornel Box, интерьерной сцены Room2 и сцены со светопроводящими элементами Light Guide. Изображения сцен представлены на рисунке 15.

Рисунок 15 – Тестовые сцены: Cornel Box слева, Room2 справа сверху, Light Guide справа снизу

Результаты профилирования данных сцен представлены в таблице 1.

Таблица 1 – Результаты профилирования процесса рендеринга сцен в системе Lumicept

Этап рендеринга	Занимаемое процессорное время		
	Cornel Box	Room2	Light Guide
Поиск фотонов	**24%**	**59%**	**28%**
Трассировка фотонов	11%	6%	15%
Построение kd-дерева	2%	1%	4%
Поиск пересечения с геометрией	8%	11%	22%
Расчет яркости	16%	5%	18%

На всех рассмотренных сценах результаты, приведенные в таблице 1, показывают, что наибольшее процессорное время рендеринга занимает операция поиска фотонов. Это подтверждает описанный раннее факт, что одним из ключевых факторов, снижающих скорость методов фотонных карт, является медленная работа на этапе поиска фотонов для сбора освещенности. Таким образом, оптимизация структур пространственного разбиения поможет существенно ускорить рендеринг.

Для сравнения скорости предложенного решения с существующим был произведен ряд тестовых вычислений на программном комплексе Lumicept. Анализировалась скорость работы системы рендеринга с существующим алгоритмом пространственного разбиения фотонных карт в виде kd дерева и предложенным комбинированным методом. Анализ выполнялся для трех сцен (классической Cornel Box, интерьерной сцены Room2 и сцены со светопроводящими элементами Light Guide), представленных выше.

Для проведения экспериментов использовался компьютер со следующими характеристиками:

- Процессор Intel Core i5 11400, 2.6 ГГц.
- Оперативная память Kingmax Zeus Dragon, DDR4, 32 ГБ, 2666 МГц.
- Использовано 6 ядер с поддержкой 12 потоков.

При каждом проведении эксперимента проводился рендеринг изображения в течение 30 минут. В качестве показателя скорости использовалось среднее количество выполненных фаз рендеринга. Результаты сравнения для Cornel Box, Room2 и Light Guides представлены в таблицах 2, 3 и 4 соответственно.

Таблица 2 – Сравнение скорости рендеринга сцены Cornel Box при помощи kd-дерева и комбинированным методом

Используемый метод	Общее время рендеринга, сек.	Количество фаз	Время на одну фазу, сек.	Прирост скорости
kd-three	1819	729	2.6	-
Combined_10	1817	809	2.25	11.1%
Combined_15	1801	785	2.29	8.8%

Таблица 3 – Сравнение скорости рендеринга сцены Room2 при помощи kd-дерева и комбинированным методом

Используемый метод	Общее время рендеринга, сек.	Количество фаз	Время на одну фазу, сек.	Прирост скорости
kd-three	1960	63	31.11	-
Combined_10	1902	74	25.79	21.1%
Combined_15	1914	81	23.62	31.7%

Таблица 4 – Сравнение скорости рендеринга сцены Light Guide при помощи kd-дерева и комбинированным методом

Используемый метод	Общее время рендеринга, сек.	Количество фаз	Время на одну фазу, сек.	Прирост скорости
kd-three	1802	305	5.91	-
Combined_10	1839	345	5.33	10.9%
Combined_15	1833	345	5.31	11.3%

Анализ результатов сравнения демонстрирует, что реализованная в данной работе структура комбинированного пространственного разбиения фотонных карт обеспечивает ускорение процесса рендеринга на 11–31% для большинства типовых сцен.

5. Заключение

Был произведен анализ существующих методов построения ускоряющих структур пространственного разбиения фотонных карт. Было показано, что использование kd деревьев обеспечивает высокую адаптивность пространственного разбиения фотонных карт, однако время доступа к фотонам данной карты является критической операцией, особенно при выполнении параллельных расчетов на многоядерных компьютерах. Наиболее подходящим решением с точки зрения времени доступа к данным фотонных карт обеспечивает регулярная решетка, организованная в виде хэш-таблицы, однако ее основной недостаток – отсутствие адаптивности разбиения, что при хорошем качестве пространственного разбиения требует значительного объема памяти. Предложенное решение – комбинация kd дерева на регулярной решетке и хэш-таблицы на ее верхнем уровне обеспечивает как высокую адаптивность пространственного разбиения, так и высокую скорость доступа к данным фотонной карты. Кроме того, данное решение позволяет снизить объем памяти, необходимой для хранения фотонных карт. Как возможное продолжение данного исследования будет рассмотрена возможность организации многоуровневых хэш-таблиц, построенных на базе kd дерева и обеспечивающих быстрый доступ к данным хэш-таблицы за счет использования SIMD инструкций.

6. Благодарности

Работа была выполнена при финансовой поддержке Российского Научного Фонда, грант № 22-11-00145.

7. Список источников

[1] Фролов В.А., Волобой А.Г., Ершов С.В., Галактионов В.А. Современное состояние методов расчёта глобальной освещенности в задачах реалистичной компьютерной графики. Фролов В.А., and. // Труды Института системного программирования РАН – 2021. vol. 33. no. 2, P. 7-48.

[2] H.W. Jensen. Global illumination using photon maps. // Eurographics workshop on Rendering techniques. Springer. Vienna. 1996. P. 21-30.

[3] J. Danskin, P. Hanrahan. Fast algorithms for volume ray tracing. // Proceedings of the 1992 workshop on Volume visualization. 1992. P. 91- 98.

[4] H.W. Jensen, P. Christensen. High quality rendering using ray tracing and photon mapping. // ACM SIGGRAPH 2007 courses. 2007. P. 116.

[5] V. Havran, R. Herzog, H.P. Seidel. Fast final gathering via reverse photon mapping. // Computer Graphics Forum. – Oxford. UK and Boston. USA: Blackwell Publishing Inc. 2005. Vol. 24. No. 3. P. 323-332.

[6] Christensen, P. Photon mapping tricks. // Siggraph 2002. Course Notes No. 43. A Practical Guide to Global Illumination using Photon Mapping organized by Jensen. H.W. 2002. P. 93-121.

[7] S.V. Ershov, D.D. Zhdanov, A.G. Voloboy, N.B. Deryabin. The method of quasi-specular elements to reduce stochastic noise in illumination simulation. // Light & Engineering. 2020. No. 5. P. 39–47.

[8] S. V. Ershov, A. G. Voloboy. Calculation of MIS weights for bidirectional path tracing with photon maps in presence of direct illumination. // Mathematica Montisnigri. 2020/ Volume 48. P. 86-102.

[9] S.V. Ershov, E.D. Birukov, A.G. Voloboy and V.A. Galaktionov. Noise Dependence on the Number of Rays in Bidirectional Stochastic Ray Tracing with Photon Maps. // Programming and Computer Software. 2021. Vol. 47. No. 3. P. 194–200.

[10] A.S. Kaplanyan, C. Dachsbacher. Adaptive progressive photon mapping. // ACM Transactions on Graphics (TOG). 2013. Vol. 32. No. 2. P. 1-13.

[11] C. Knaus, M. Zwicker. Progressive photon mapping: A probabilistic approach. // ACM Transactions on Graphics (TOG). 2011. Vol. 30. No. 3. P. 1-13.

[12] T. Hachisuka, H.W. Jensen. Stochastic progressive photon mapping. // ACM SIGGRAPH Asia 2009 papers. 2009. P. 1-8.
[13] J.L. Bentley, J.H. Friedman. Data structures for range searching. // ACM Computing Surveys (CSUR). 1979. Vol. 11. No. 4. P. 397-409.
[14] Q.P. Chen, B. Xue, J.I. Siepmann. Using the k-d tree data structure to accelerate Monte Carlo simulations. // Journal of Chemical Theory and Computation. 2017. Vol. 13. No. 4. P. 1556-1565.
[15] W. Hunt, W.R. Mark, G. Stoll. Fast kd-tree construction with an adaptive error-bounded heuristic. // 2006 IEEE Symposium on Interactive Ray Tracing. IEEE. 2006. P. 81-88.
[16] Lumicept Integra [Электронный ресурс]. Режим доступа – URL: https://integra.jp/en/products/lumicept.
[17] Zhdanov A.D., Zhdanov D.D. The two-level semi-synchronous parallelization method for the caustic and indirect luminance calculation in realistic rendering // CEUR Workshop Proceedings 2020. Vol. 2744. P. 1-12.