Метод излучательности с упрощенным расчетом форм-факторов. Аспекты параллельной реализации

Дебелов Виктор Алексеевич, Саттаров Максим Александрович
ИВМиМГ СО РАН, Новосибирск, Россия

Аннотация

В данной работе рассматривается оригинальный подход в методе излучательности, позволяющий ускорить процесс расчета изображения по сравнению с классическим методом расчета. Подход основан на предварительной обработке сцены – рекурсивном разбиении конечных элементов сцены до тех пор, пока проекция конечного элемента не станет меньше одного пиксела изображения. Еще одной особенностью метода является то, что расчет форм-фактора сводится к вычислению функции видимости между конечными элементами.
Предложена параллельная реализация разработанного метода, которая позволяет ускорить расчет конечного изображения. Эксперименты показывают, что достигается ускорение, которое зависит линейно от числа процессоров, правда, с коэффициентом меньшим 1.

Ключевые слова*: radiosity, form factor, parallel algorithm, subdivision of finite elements*.

1. ВВЕДЕНИЕ

Одной из проблем, возникающих при расчете фотореалистических изображений конечно-элементными методами (метод излучательности), является проблема возникновения артефактов на конечном изображении, обусловленных чрезмерно крупными по сравнению с разрешением конечного изображения или неудачно выбранными конечными элементами, из которых состоит сцена. Безусловно, используемый метод подсчета форм-факторов вносит дополнительные погрешности. Например, при расчете форм-факторов по методу полукуба [1], если центр конечного элемента заслонен каким-либо объектом от источника света, то и весь конечный элемент оказывается темным. Тогда, в случае крупных конечных элементов, тень от загораживающего объекта будет искажена.
Другая проблема в методе излучательности – это поиск быстрого и как можно более точного метода расчета форм-факторов. Для точного расчета форм-фактора необходимо вычислять функцию видимости для *нескольких* пар точек, чтобы затем воспользоваться одним из приближенных методов вычисления интеграла. Вычисление функции видимости требует перебора тысяч конечных элементов, если не используются специальные методы, позволяющие уменьшить этот перебор. Однако, так или иначе, вычисление функции видимости является весьма трудоемкой процедурой, и сокращение количества ее вызовов всегда положительно сказывается на быстродействии.
В предлагаемой работе предлагается подход к решению этих проблем. Рассматриваемый метод основан на предобработке сцены до начала вычислений. Изначально фиксируются параметры камеры, а затем слишком крупные элементы сцены итеративно разбиваются на более мелкие до тех пор, пока все элементы сцены не станут достаточно малы – пока каждый из них не станет занимать на результирующем изображении менее одного пиксела. После этого для обработанной таким образом сцены может применяться практически любой метод расчета освещенности и получения конечного изображения. Конечно, артефакты на конечном изображении могут оставаться и после такой предобработки, однако, каждый такой “шумовой” конечный элемент будет занимать не более одного пиксела, и для их удаления можно применить к конечному изображению какой-либо низкочастотный фильтр, например, гауссовский фильтр с параметром больше единицы, или медианный. Если качество такого отфильтрованного изображения неудовлетворительно из-за недостаточной четкости (низкочастотные фильтры характеризуются тем, что они, в некоторой степени, “размывают” изображение), то можно применить рендеринг на подпикселном уровне, т.е. рассчитывать изображение для более мелкой пиксельной сетки. Именно это разрешение используется как основа для критерия разбиения сцены в предлагаемом методе. После этого рассчитывается “большое” изображение, которое затем преобразуется в изображение требуемого размера с помощью усреднения значений пикселов. Для усреднения могут быть вновь применены параметры гауссовского фильтра, либо использовано обычное усреднение по какому-либо шаблону.
Впервые подход, использующий конечные элементы с размерами не превышающими размер пиксела, был предложен в работе [2]. В этой работе рассматриваются варианты получения фотореалистических изображений сцены, состоящей из полигонов. В терминологии, предлагаемой в работе [2], полигоны, которые на конечном изображении занимают один и менее пикселов, называются *микрополигонами*. Основной предмет рассмотрения в этой работе – сравнение технологий текстурирования, использующих микрополигоны. Текстуры пассивны, они не участвуют в расчете балланса освещенности в сцене, поэтому для задачи, рассматриваемой в данной работе, подход, связанный с применением текстур, неприменим.
В данной работе рассматривается один из вариантов параллельной реализации всей цепочки алгоритмов, начиная с предобработки сцены и заканчивая итеративным решением системы линейных уравнений в классическом методе излучательности. Параллельная реализация алгоритмов основывается на применении библиотеки MPI [3]. В работе анализируются результаты численных экспериментов, демонстрирующих преимущества и недостатки предлагаемых методов, а также эффективность распараллеливания.

2. МЕТОД ИЗЛУЧАТЕЛЬНОСТИ

2.1. Вывод основных уравнений

Пусть *сцена* **S –** произвольная кусочно-гладкая поверхность, а $L(x,\omega)$ - *функция яркости,* то есть мощность света, излучаемого из точки $x \in S$ в направлении ω.

Камерой будем называть упорядоченный набор параметров $(r, x, y, z, N_x, N_y, d)$, где

$r \in \mathbf{R}^3$ – центр камеры, то есть начало системы координат камеры;

(x, y, z) – ортонормированный базис системы координат камеры, $x, y, z \in \mathbf{R}^3$;

(N_x, N_y) – *разрешение* камеры, то есть пара целых чисел, обозначающих количество пикселей по горизонтали и по вертикали в изображении, получаемом с помощью этой камеры;

d – *фокусное расстояние* камеры, расстояние от центра камеры до ее фокуса F;

Изображением будем называть равномерную двумерную (N_x, N_y) решетку из *пикселов.* Пиксел – это элемент изображения, хранящий одно вещественное значение $v \in [0,1]$.

Мы будем рассматривать лишь изображения в градациях серого, поскольку в рамках нашей модели цветное изображение – это всего лишь три независимых изображения в градациях серого (красный, зеленый и голубой каналы), Другими словами в рассматриваемой модели освещенности, взаимодействие света разных частот и изменение частоты света после отражения от поверхности не учитываются.

В рамках введенных обозначений *общее уравнение визуализации* (rendering equation, [1]) записывается в виде следующего интегрального соотношения:

$$L(x,\omega) = \int_{\Omega}\int_{S} K(x,\omega,y,\omega') \cdot L(y,\omega') d_y S \, d_{\omega'}\Omega + L_e(x,\omega) \qquad (1)$$

где Ω – единичная сфера, $K(x,\omega,y,\omega')$ – функция ядра, а $L_e(x,\omega)$ – *собственное излучение* (эмиссия) сцены.

Это уравнение совпадает с одной из форм известного уравнения переноса излучения (см., например, [4]).

В *классическом методе излучательности* [1] рассматриваются диффузные сцены, т.е. сцены, поверхность которых рассеивает свет во все стороны равномерно. Поэтому в этом методе полностью пренебрегают зависимостью *функции яркости L* от направления, т.е. вместо функции $L(x,\omega)$ рассматривают функцию $L(x)$. Соответственно и функция ядра $K(x,\omega,y,\omega')$ будет зависеть лишь от двух переменных и будет иметь вид $K(x,y)$.

Отметим, что существуют модификации метода излучательности, которые работают и для зеркально-диффузных поверхностей, см., например, [5]. В этих методах обычно сначала рассчитывается баланс энергии с помощью обычного метода излучательности, а потом дополнительно рассчитываются эффекты зеркальности. При этом общее уравнение визуализации в виде (1) не используется.

Функцию $K(x,y)$ обычно вычисляют по формуле:

$$K(x,y) = \begin{cases} \dfrac{k(x)}{\pi} \cdot V(x,y) \cdot \dfrac{\left|\cos\angle(x-y,n_x) \cdot \cos\angle(x-y,n_y)\right|}{\|x-y\|^2}, \\ \text{если } x \neq y \\ 0, \text{ иначе} \end{cases},$$

где *функция видимости*

$$V(x,y) = \begin{cases} 1, \text{ если открытый отрезок (x, y)} \\ \qquad\qquad \text{не пересекается с S}, \\ 0, \text{ иначе} \end{cases}$$

и $k(x)$ – *коэффициент рассеивания* в точке x.

Такой вид функции ядра соответствует отражению по закону Ламберта (диффузному отражению) [1], при котором излучение в данной точке поверхности не зависит от направления на наблюдателя. В этих предположениях уравнение визуализации для классического метода излучательности записывается в следующем виде:

$$L(x) = \int_S K(x,y) \cdot L(y) d_y S + L_e(x) \qquad (2)$$

Существует также форма уравнения (2) для двусторонних поверхностей [6]. В двустороннем методе, в отличие от классического метода излучательности, пренебрегают зависимостью функции яркости L от направления не полностью, а лишь в следующем смысле: для каждой точки x поверхности сцены считают, что в ней для всех направлений ω заданы лишь два значения функции яркости – одно для направлений, составляющих острый угол с направлением нормали в этой точке, и другое – для составляющих тупой угол. Таким образом, в двустороннем методе излучательности вместо функции $L(x,\omega)$ рассматривается функция $L(x,i)$, где i принимает значения ± 1 в зависимости от того, с какой стороны поверхности берется значение функции яркости.

2.2. Дискретизация уравнения визуализации для классического метода излучательности

Для численного решения уравнения (2) будем использовать один из вариантов *метода конечных элементов – метод Галеркина* [7].

Пусть $L_2(S)$ – пространство суммируемых с квадратом функций, заданных на кусочно-гладкой поверхности **S.** Выберем конечный ортонормированный в $L_2(S)$ набор *базисных* функций $\{\varphi_i\}_{i=1}^N$. Будем искать приближения неизвестной функций в виде линейной комбинации этих базисных функций:

$$L(x) = \sum_{j=1}^{N} L_j \varphi_j(x) \qquad (3)$$

Подставляя представление (3) в уравнение (2) и затем скалярно (в $L_2(S)$) умножая уравнение на φ_k и деля на

площадь конечного элемента $mes(S^i)$, получим следующую систему линейных уравнений относительно неизвестных L_i:

$$L_i = \sum_{j=1}^{N} F_{ij} L_j + L_i^e \quad , \qquad (4)$$

где

$$F_{ij} = \frac{1}{mes(S^i)} \int_S \int_S K(x,y)\varphi_i(x)\varphi_j(y) d_x S d_y S \qquad (5)$$

$\left(F_{ij}\right)$ называют матрицей *форм-факторов,* или матрицей *геометрических коэффициентов формы*. В литературе достаточно часто принимается несколько иное определение:

$$F'_{ij} = \frac{1}{mes(S^i)} \int_S \int_S \frac{1}{k(x)} K(x,y)\varphi_i(x)\varphi_j(y) d_x S d_y S$$

Отличие заключается в том, что F'_{ij} зависит лишь от геометрии сцены, и не зависит от коэффициента рассеивания, который обуславливается физическими характеристиками сцены. Однако в данной работе нам будет удобнее использовать определение (5).

Физически форм-фактор F_{ij} представляет собой долю световой энергии, падающей на i-й конечный элемент, которая, будучи отражена от него, попадает на j-й конечный элемент. Отсюда, в частности, следует, что норма матрицы форм-факторов, подсчитанная как максимум равномерных норм строк, меньше единицы.

Обычно в методе излучательности используется кусочно-постоянное восполнение, то есть в качестве базисных выбираются кусочно-постоянные функции. В этом случае вся сцена **S** разбивается на *конечные элементы* S^i так, что

$$S = \bigcup_i S^i \text{ и } S^i \cap S^j = \varnothing \text{, если } i \neq j \, .$$

В качестве базисных функций берутся

$$\varphi_i(x) = \begin{cases} 1, \text{ если x} \in \text{S}^\text{i} \\ 0, \text{ если x} \notin \text{S}^\text{i} \end{cases} .$$

Введем еще одно упрощение: будем считать, что коэффициент рассеивания *k(x)* также является кусочно-постоянным, то есть

$$k(x) = k_j \; \forall x \in S_j \, .$$

В этом случае можно привести более конкретную формулу для вычисления элементов матрицы форм-факторов F_{ij}:

$$F_{ij} = k_j \frac{1}{mes(S^i)} \times$$

$$\int_{S^i} \int_{S^j} \frac{\left| < n_x, x-y >< n_y, x-y > \right|}{\|x-y\|^2} V(x,y) \, d_y S \, d_x S$$

Ясно, что самым сложным моментом в вычислении форм-факторов является численный подсчет четверного поверхностного интеграла. Для этого используется несколько основных методов:

1. Упрощенный. В этом методе приближенное значение форм-фактора оценивается по одному лучу между двумя точками на конечных элементах S^i и S^j, например, между их центрами, по следующей формуле:

$$F_{ij} = mes(S^j) k_j \frac{\left| < n_x, x-y >< n_y, x-y > \right|}{\|x-y\|^2} V(x,y) \qquad (6)$$

где x и y – точки, принадлежащие S^i и S^j, соответственно.

Недостатком этого метода является возникновение артефактов при выборе слишком крупных конечных элементов S^i. Однако имеется и серьезное преимущество – функция видимости вычисляется лишь однажды.

Замечание: единственная операция в (6), которая требует значительных вычислительных затрат, – это вычисление функции видимости. Поэтому имеет смысл заранее подсчитывать и хранить в памяти лишь значения функции видимости, а форм-факторы получать в момент решения системы линейных уравнений (4) “на лету”. Это позволяет значительно сэкономить память, ведь значение функции видимости занимает один бит, а вещественное значение двойной точности – 64 бита.

2. По нескольким точкам. Выбираем на каждом конечном элементе какое-то количество точек (в случае прямоугольных конечных элементов можно выбрать точки равномерной сетки). Пусть на конечном элементе S^i выбраны точки $\{x_k\}_{k=1}^N$, а на S^j – $\{y_l\}_{l=1}^M$. Тогда форм-фактор F_{ij} можно оценить с помощью следующего приближения:

$$F_{ij} = mes(S^j) k_j \sum_{k=1}^{N} \sum_{l=1}^{M} \frac{\left| < n_x, x_k - y_l >< n_y, x_k - y_l > \right|}{\|x_k - y_l\|^2} V(x_k, y_l)$$

Вариантом этого метода вычисления форм-факторов может быть метод Монте-Карло [8], примененный для вычисления четверного интеграла.

Нужно заметить, что несмотря на то, что рассматриваемая формула при больших числах M и N дает довольно хорошее приближение, весьма серьезным недостатком этого метода является то, что при увеличении точности сильно увеличиваются вычислительные затраты за счет многократного подсчета функции видимости.

Существуют и другие, более специальные, методы вычисления форм-факторов [1], например, *метод полукуба,* однако, в рамках данной работы они не рассматриваются.

3. МЕТОД ИЗЛУЧАТЕЛЬНОСТИ С УПРОЩЕННЫМ РАСЧЕТОМ ФОРМ-ФАКТОРОВ

3.1. Постановка задачи

Как и ранее, будем рассматривать сцену S – кусочно-гладкую поверхность с заданными на ней как функции точки коэффициентами рассеяния и эмиссии. Пусть сцена представлена в виде конечного множества гладких

непересекающихся поверхностей S_i , которые мы будем называть *конечными элементами (КЭ)*:

$$S = \bigcup_{i=1}^{N} S_i \text{, и } S_i \cap S_j = \varnothing \text{ при } i \neq j .$$

Зафиксируем описанные в п. 2.1 параметры камеры $(r, x, y, z, N_x, N_y, d)$.

Разбиением сцены будем называть множество конечных элементов $\left\{S_{ij}\right\}_{\substack{1<i<N \\ 1<j<M_i}}$, удовлетворяющее следующим условиям:

$$S_{ij} \cap S_{kl} = \varnothing \text{, если } i \neq k \text{ или } j \neq l ,$$

$$\bigcup_{j=1}^{M_i} S_{ij} = S_i \; \forall i = 1..N .$$

При этом будем говорить, что $\left\{S_{ij}\right\}_{1<j<M_i}$ (при фиксированном i) – это *разбиение конечного элемента* S_i .

Будем называть произвольную булеву (принимающую лишь два значения – 0 и 1) функцию α конечного элемента S_{ij} *критерием разбиения*. Критерий разбиения зависит как от сцены S , так и от параметров камеры, перечисленных выше.

Сформулируем основную задачу, рассматриваемую в данном разделе, в рамках введенной терминологии. Требуется найти разбиение сцены $\left\{S_{ij}\right\}_{1<i<N, 1<j<M_i}$, удовлетворяющее условию $\alpha(S_{ij}) = 1$ для любого конечного элемента S_{ij} этого разбиения. Кроме того, мы будем минимизировать количество элементов конечного разбиения, поскольку вычислительные ресурсы, требуемые для расчета конечного изображения, сильно зависят от количества конечных элементов сцены. Именно количество конечных элементов в конечном разбиении будет служить мерой эффективности для алгоритма построения разбиения.

Теперь рассмотрим один их возможных критериев разбиения. Для данного разбиения сцены $\left\{S_{ij}\right\}_{i,j}$ рассмотрим растровое "изображение", лежащее в квадрате $[-1, 1]\times[-1\ 1]$. в плоскости векторов ***a*** и ***b***, имеющее разрешение $\left(N_x, N_y\right)$, и хранящее пару (i, j) в пикселе, если луч, исходящий из фокуса камеры и проходящий через центр этого пикселя впервые пересекает сцену в конечном элементе S_{ij} , и особую пару значений, отличную от индексов всех конечных элементов разбиения (например, $(-1,-1)$), если луч не пересекает сцену.

Положим $\alpha(S_{ij}) = 1$, если рассматриваемый растр содержит один и менее пикселей, соответствующих S_{ij} . Такой критерий разбиения будем называть *основным критерием разбиения*. Основной критерий разбиения гарантирует нам то, что любой конечный элемент полученного разбиения сцены будет занимать на результирующем изображении не более одного пикселя.

3.2. Описание алгоритма

Далее мы везде будем предполагать, что $\alpha(S_{ij})$ – это основной критерий разбиения. Для этого критерия мы можем построить достаточно эффективный алгоритм решения поставленной задачи. Этот алгоритм описывается следующими строками псевдокода:

```
InitTheDivision;
While(не все конечные элементы текущего разбиения удовлетворяют критерию разбиения)
{
    RenderTheImage;
    For each S_ij
    {
        if (α(S_ij)≠1)
            Split(S_ij);
    }
}
```

Рассмотрим подробнее операторы этого псевдокода:

- **`InitTheDivision`** – процедура, создающая начальное разбиение сцены следующим образом:

$$S_{i0} = S_i .$$

- **`RenderTheImage`** – процедура построения растра, рассматриваемого выше, для облегчения вычисления критерия разбиения. После того, как растр построен, вычисление критерия сводится к выяснению факта, что конечный элемент занимает более одного пикселя на этом растре.
- **`Split`** – процедура, которая разбивает данный конечный элемент и добавляет результат в разбиение. Например, прямоугольные конечные элементы можно разбивать на две части прямой, соединяющей середины больших сторон, а треугольник – медианой, проходящей через середину большей стороны. Такой способ разбиения обеспечивает выполнение следующего *условия измельчения разбиения*, важного для дальнейшего изложения:

> ***получаемые конечные элементы имеют радиусы описанных вокруг них сфер, относящимися к радиусу описанной сферы исходного конечного элемента как 1 к r, где r < s < 1 (s фиксирован для данного типа конечных элементов).***

Рассмотрим работу приведенного алгоритма на простой сцене, состоящей из трех прямоугольников, верхний из которых является источником света, нижний экраном, а средний – загораживает источник света от экрана. На рис. 1 приводятся изображения этой сцены на различных шагах работы рассматриваемого алгоритма.

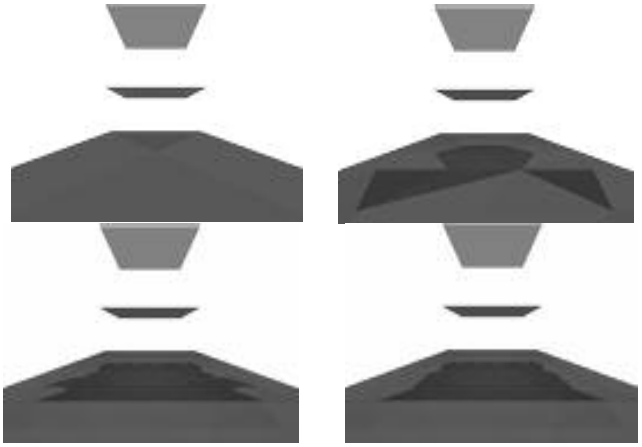


Рис. 1. Сцена на различных шагах алгоритма разбиения

Ясно, что большая часть времени работы этого алгоритма будет уходить на построение растра, то есть на пересечение лучей, проходящих через центры пикселей, с конечными элементами сцены. Это становится особенно важным если принять во внимание то, что количество конечных элементов сцены может расти с каждым следующим шагом алгоритма в геометрической прогрессии. Предлагается следующий алгоритм пересечения луча со сценой, который позволяет снизить зависимость количества операций, затрачиваемых на построение растра, от степени измельчения сцены. Будем хранить исходную сцену как массив конечных элементов. Кроме того, заведем массив списков конечных элементов, на которые разбивается исходная сцена, т.е. в i-том списке хранятся конечные элементы, на которые разбит i-ый конечный элемент исходной сцены. Когда требуется узнать, с каким конечным элементом измельченной сцены пересекается некоторый луч, мы пересекаем его с исходной сценой (в которой, вообще говоря, меньше конечных элементов, чем в измельченной), а затем ищем, с каким из конечных элементов соответствующего списка пересекается этот луч. Такие способ хранения обработанной сцены и алгоритм пересечения сцены с лучом позволяют существенно снизить сложность задачи вычисления функции видимости. Подобный прием используется для снижения времени пересечения луча со сценой в *кластерных* методах [9].

3.3. Обоснование корректности алгоритма

Теорема. Описанный в п. 3.2 алгоритм для прямоугольных и треугольных конечных элементов решает задачу нахождения разбиения сцены за конечное количество шагов

Доказательство

То, что построенный алгоритм остановится через конечное число шагов, обеспечивается тем, что каждая операция разбиения конечного элемента удовлетворяет условию измельчения разбиения.

Действительно, пусть условие измельчения разбиения выполнено. Обозначим радиус сферы, описанной около конечного элемента S_i через $R(S_i)$. Пусть $R_{\max} = \max_i R(S_i)$ – наибольший радиус описанной сферы, а q – размер пикселя. Тогда, если

$$k > k_0 = \frac{\log\left(\frac{q}{R_{\max}}\right)}{\log s} \qquad (7)$$

то на k-ом шаге радиус сферы, описанной около любого конечного элемента текущего разбиения, станет меньше размера пикселя, и, следовательно, два луча, проходящие через центры разных пикселей, не смогут пересечься с одним и тем же конечным элементом, и будет выполнено условие остановки алгоритма.

Для описанного способа разбиения прямоугольных и треугольных конечных элементов это условие может быть установлено из достаточно простых геометрических соображений.

То, что каждый конечный элемент результирующего разбиения будет удовлетворять критерию разбиения, очевидным образом следует из условия остановки алгоритма □

Замечание: Из оценки (7) в доказательстве теоремы 1 можно вывести верхнюю оценку количества действий в рассматриваемом алгоритме: $T < Ck_0 N_x N_y$, где T – количество действий в алгоритме, (N_x, N_y) – разрешение камеры, а константа C зависит от трудоемкости вычисления пересечения луча с конкретной сценой.

4. РАСПАРАЛЛЕЛИВАНИЕ АЛГОРИТМОВ

4.1. Общие замечания

Для классического метода излучательности характерна с одной стороны большая вычислительная сложность, связанная с необходимостью вычисления функции видимости, а с другой стороны – огромные затраты памяти на хранение матрицы форм-факторов (или матрицы видимости). В настоящее время расчет реальных сцен с большим количеством конечных элементов оказывается возможным лишь при использовании параллельных вычислительных систем. В данном докладе будет подробно рассмотрено распараллеливание лишь некоторых этапов алгоритма, а именно: фаз предварительной обработки и вычисления функции видимости.

4.2. Распараллеливание фазы предварительной обработки

Как в рассмотренном классическом методе излучательности с упрощенным расчетом форм-факторов, так и в варианте, основанном на методе прогрессивной излучательности, присутствует фаза предварительной подготовки сцены – фаза измельчения конечных элементов. Последовательный вариант алгоритма этой фазы вычислений рассмотрен в разделе 3. Очевидны два пути распараллеливания предобработки сцены.

Первый заключается в том, что между процессорами разделяются области экрана, и каждый процесс разбивает те конечные элементы сцены, которые занимают более одного пикселя лишь в этой области экрана. Проблема тут заключается в том, что некоторые конечные элементы могут

одновременно быть “видны” в нескольких таких областях экрана, и в этом случае их разбиением должны заниматься несколько процессов одновременно. Возникает необходимость согласования этих разбиений, что приводит к дополнительным вычислительным усилиям и повышению нагрузки на механизм пересылки данных между процессами, что может быть критичным для вычислительных систем с разделенной памятью (например, кластеров).

Другой путь распараллеливания предобработки сцены заключается в том, что части сцены разделяются между процессами, а затем каждый процесс разбивает свою подсцену. Каждый процесс имеет полноразмерный “экран”, но “отрисовывает” на нем лишь свою часть сцены. Это позволяет избежать накладных расходов, связанных с пересылкой данных между процессами во время работы: в этом алгоритме процессы выполняют совершенно независимые задачи. Недостаток такого подхода к распараллеливанию предобработки сцены заключается в том, что, вообще говоря, полученное разбиение сцены будет отличаться от полученного в результате работы исходного последовательного алгоритма – оно будет сильнее измельчено из-за того, что некоторые конечные элементы, будучи заслонены от экрана другими конечными элементами в последовательной версии алгоритма (и соответственно не разбиваясь на более мелкие части), оказываются не заслоненными, если рассматривается лишь подсцена конкретного процесса. Тем не менее, результаты, которые дает этот параллельный алгоритм, являются вполне удовлетворительными, более того, за счет большего измельчения разбиения сцены реалистичность изображения даже увеличивается, и недостаток превращается в достоинство.

4.3. Распараллеливание фазы вычисления функции видимости

Вычисление матрицы видимости – это самая трудоемкая фаза метода излучательности, поэтому требуется тщательное распределение нагрузки между процессами и постоянный контроль над ней, поскольку невозможно заранее сказать, сколько времени займет вычисление конкретной ячейки или блока матрицы видимости.

Предлагается следующий вариант параллельного алгоритма вычисления матрицы видимости. Один из процессов мы выделим среди остальных и назовем *главным*. Остальные процессы будут *подчиненными*. Главный процесс будет заниматься балансировкой нагрузки подчиненных процессов. Каждый из подчиненных процессов будет вычислять определенный набор блоков матрицы видимости, которые определяются парой конечных элементов исходной сцены, разбитых на более мелкие конечные элементы в предобработанной сцене. Таким образом, для пары (S_i, S_j) вычисляемый блок матрицы видимости состоит из значений функции видимости на парах конечных элементов (S_{ik}, S_{jl}).

Главный процесс перебирает все пары конечных элементов и по очереди рассылает номера этих пар в качестве *заданий* подчиненным процессам. Главный процесс следит за загрузкой каждого из процессов, получая от них рапорты о выполнении заданий. Только когда выполнено текущее задание (вычислен блок матрицы видимости), подчиненный процесс получает новое. Когда все задания разосланы и выполнены, главный процесс получает вычисленные блоки матрицы видимости от подчиненных процессов, собирает из них всю матрицу видимости и записывает ее в файл. Такая организация вычислений позволяет практически полностью избежать простоя рабочих процессов. Существенным недостатком является то, что главный процесс фактически исключен из реальных вычислений и занимается лишь балансировкой загрузки остальной процессов. Однако с ростом количества процессоров этот недостаток становится все менее значимым.

Замечание. Для сцен, которые изначально состоят из достаточно большого количества конечных элементов, имеет смысл в качестве заданий подчиненным процессам рассматривать вычисление сразу целой полосы матрицы видимости, определяющейся одним конечным элементом, поскольку для таких сцен накладные вычислительные затраты на пересылку заданий и рапортов могут превышать затраты на собственно вычисление блоков матрицы видимости. Это особенно важно при использовании машин с раздельной памятью и кластеров рабочих станций. Именно такой прием использовался при проведении вычислительных экспериментов, описанных в п. 5.2.

5. ВЫЧИСЛИТЕЛЬНЫЕ ЭКСПЕРИМЕНТЫ

5.1. Анализ метода излучательности с упрощенным расчетом форм-факторов

Данный раздел посвящен анализу проведенной серии вычислительных экспериментов, направленных на изучение работы описанных алгоритмов на практике. В первом параграфе мы проведем исследование метода излучательности с упрощенным расчетом форм-факторов. Все эксперименты, описанные в первом подразделе этого раздела, были проведены на следующей конфигурации вычислительной системы: Intel Celeron 800, 384 Mb RAM, Windows XP. Вычисления проводились с использованием последовательной реализации алгоритмов, описанных выше. Вычислительные эксперименты, которые анализируются в данном параграфе, проводились на сцене “Constructive Wood” (“Конструктивный лес”) [1], которая очень часто применяется при анализе алгоритмов, основанных на методе излучательности. На рис. 2 показан схематичный вид сверху на эту сцену.

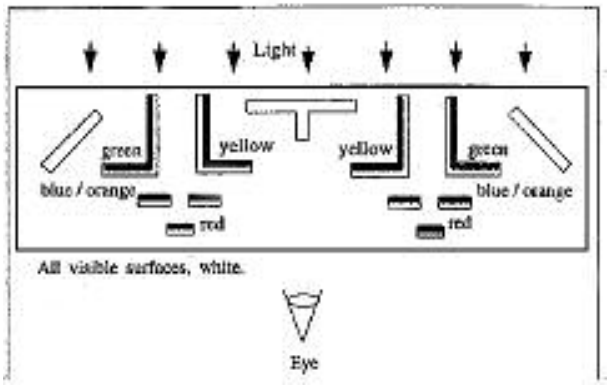


Рис. 2. Сцена “Constructive Wood”

Как видно из рис. 2, все поверхности, обращенные к глазу – это белые идеальные диффузные отражатели, однако за счет

света сзади и переотражений от цветных поверхностей, эти белые поверхности окрашиваются.

Алгоритм метода излучательности с упрощенным расчетом форм-факторов, предложенный в разделе 2, состоит из трех основных частей: предобработка сцены, вычисление функции видимости и решение СЛАУ.

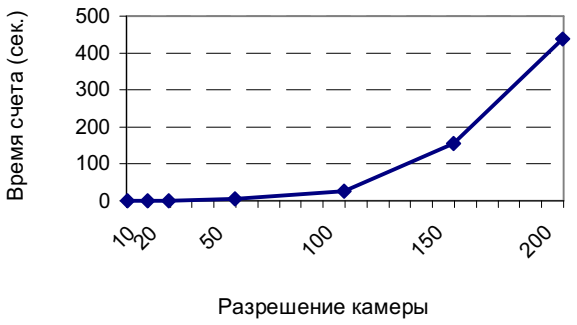


Рис. 3. Зависимость времени предобработки сцены от разрешения камеры

Здесь наиболее трудоемким оказался этап вычисления матрицы видимости. На рис. 3 показана зависимость времени расчета матрицы видимости от разрешения камеры. Однако, более интересна зависимость времени расчета матрицы видимости от количества конечных элементов в

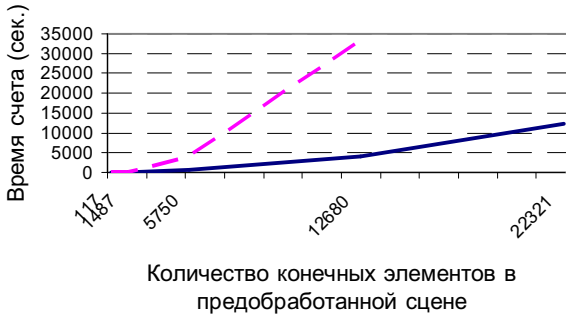


Рис. 4. Зависимость времени расчета матрицы видимости от количества элементов в предобработанной сцене

предобработанной сцене. На рис. 4 приводится эта зависимость в сравнении с классическим методом излучательности (пунктирная линия на графике). Для классического метода излучательности характерна зависимость третьей степени между временем расчета матрицы видимости и количеством элементов в сцене, в то время как в предлагаемом методе излучательности с упрощенным расчетом форм-факторов та же зависимость предположительно близка к квадратичной. Более точные теоретические оценки этой зависимости для метода излучательности с упрощенным расчетом форм-факторов вывести не удается, однако график на рис. 4, построенный по экспериментальным данным, в некоторой мере подтверждает эти соображения. Эти различия объясняются применяемым способом хранения и вычисления функции видимости (см. п. 3.2), который для сцены Constructive Wood особенно эффективен из-за малого количества конечных элементов в исходной сцене.

5.2. Исследование характеристик параллельной реализации

Цель данного параграфа – анализ параллельной реализации метода излучательности с упрощенным расчетом форм-факторов. Все эксперименты, описанные ниже проводились на сцене, изображенной на серии рисунков в заключении доклада. Сцена изначально состояла из 2391 треугольника.

Вычислительные эксперименты проводились на кластере МВС-1000 Сибирского Суперкомпьютерного Центра СО РАН (http://www2.sscc.ru). Кластер работает под управлением RedHat Linux и в данный момент состоит из пяти вычислительных модулей. Каждый вычислительный модуль имеет два процессора над общей памятью в 2 Gb. Задачи запускались в монопольном режиме.

2	3	4	5	6	7	8	9	10
6500	3026	1850	1534	1345	1123	907	580	503
1	2.15	3.51	4.2	4.83	5.79	7.17	11.21	12.92

В прелагаемой таблице строки имеют следующий смысл:

1. Число процессоров.
2. Время счета в секундах.
3. Коэффициент ускорения по сравнению с двухпроцессорным вариантом.

Основной параметр, анализ которого представляет интерес при исследовании любого параллельного алгоритма, – это так называемый коэффициент ускорения алгоритма. Коэффициентом ускорения параллельного алгоритма называют отношение времени выполнения алгоритма на 1 процессоре ко времени выполнения этого же алгоритма на N процессорах. Идеальный параллельный алгоритм должен обладать свойством линейности ускорения, то есть вдвое большее количество процессоров посчитает задачу в два раза быстрее. Результаты экспериментов, представленные в таблице, показывают, что в целом предлагаемый вариант распараллеливания метода излучательности с упрощенным расчетом форм-факторов работает вполне удовлетворительно. С увеличением числа процессоров уменьшается время счета почти линейно.

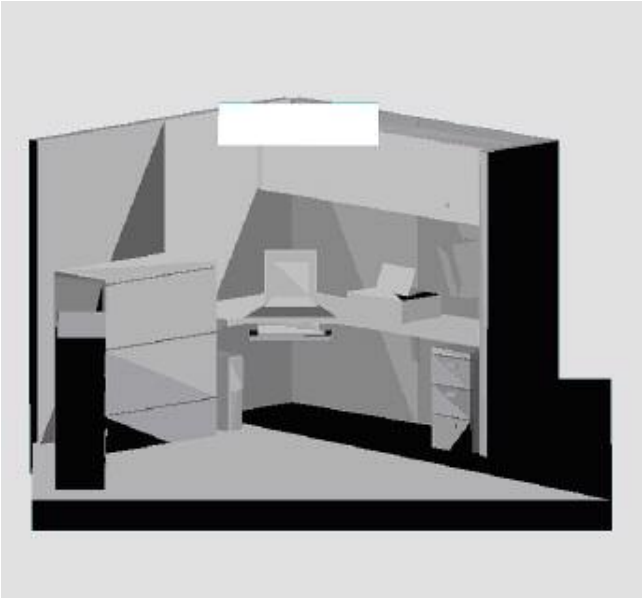
6. ЗАКЛЮЧЕНИЕ

В данной работе были представлены следующие результаты:

- Предложен метод предобработки сцены, который позволяет применять упрощенную схему расчета форм-факторов в различных вариантах метода излучательности.
- Обоснована корректность этого метода предобработки и получена оценка трудоемкости соответствующего алгоритма.
- Предложены схемы распараллеливания как самого метода предобработки сцены, так и метода излучательности с упрощенным расчетом форм-факторов.
- Проведена серия вычислительных экспериментов, направленных на анализ предлагаемых методов и эффективности предложенных вариантов их распараллеливания, в частности, исследовано ускорение параллельной реализации по сравнению с последовательной реализацией в зависимости от количества процессоров.

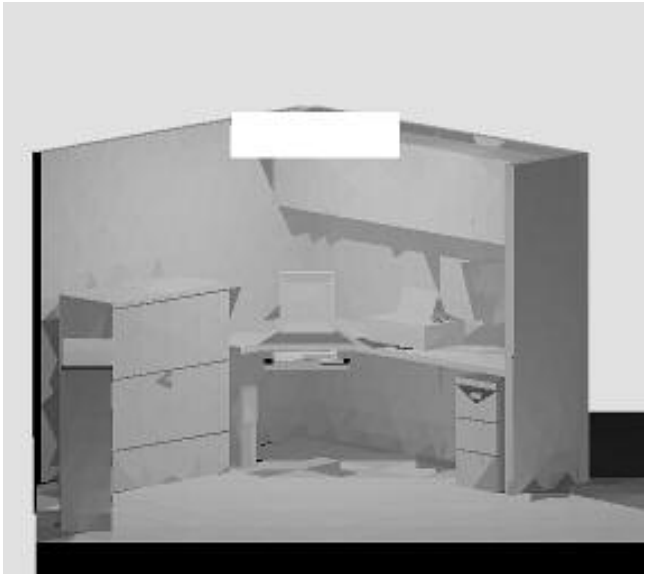
В заключении приведем серию изображений тестовой сцены, полученных с помощью описанных методов. На этих

рисунках видно, как уменьшается количество артефактов с ростом разрешения камеры.
На первом рисунке сцена изображена с помощью метода излучательности с упрощенным расчетом форм-факторов, но без предварительной обработки.



На втором рисунке приводится изображение той же сцены, однако в данном случае была применена предварительная обработка сцены исходя из разрешения камеры 50 на 50 пикселов.

Наконец, изображение на третьем рисунке получено при разрешении камеры 70 на 70 точек, причем к конечному изображению был применен гауссовский фильтр.



Благодарности

Данная работа проводилась при частичной поддержке РФФИ по грантам №№ 99-01-00577 , 99-01-90422 и 01–01–00817.

7. ЛИТЕРАТУРА

[1] Cohen, M.F., Wallace, J.R. Radiosity and Realistic Image Synthesis. Academic Press, 1993.

[2] Robert, C., Carpenter, L., Catmull, E. The Reyes Rendering Architecture, SIGGRAPH'87, 95-102.

[3] MPI Software Technology, http://www.mpich.com.

[4] Шорин С.Н. Теплопередача. М.: Высшая школа, 1964.

[5] Мальдон Д.В., Упольников С.А. Использование метода излучательности для сцен с зеркальными поверхностями, Graphicon'97, 21-24.

[6] Саттаров М.А. Метод излучательности с упрощенным расчетом форм-факторов. Аспекты параллельной реализации. Магистерская диссертация. Новосибирск: НГУ, 2002.

[7] Марчук Г.И. Методы вычислительной математики. М: Наука, 1980.

[8] Ермаков С.М., Михайлов Г.А. Статистическое моделирование. М.: Наука, 1982.

[9] Smits, B., Arvo, J., Greenberg, D. A clustering algorithm for radiosity in complex environments. In Proceedings of SIGGRAPH ’94, 435–442.

Об авторах

Лаборатория численного анализа и машинной графики Института вычислительной математики и математической геофизи СО РАН, Новосибирск, Россия.
Дебелов Виктор Алексеевич – ведущий научный сотрудник, к.ф.-м.н., с.н.с, E-mail: debelov@oapmg.sscc.ru.
Саттаров Максим Александрович – аспирант,
E-mail: sattarov@dataeast.ru.

Parallel radiosity algorithm with simplified calculation of form-factors

Victor A. Debelov, Maxim A. Sattarov
Institute of Computational Mathematics and
Mathematical Geophysics SB RAS,
Novosibirsk, Russia

Abstract

The given paper is devoted to an original approach allowing the speedup of the process of image calculations with respect to the classical matrix radiosity algorithm. It bases on the preprocessing phase when finite elements of a scene are subdivided recursively while an image of a single finite element is greater than 1 pixel. Another particularity of the method is that a calculation of form-factor between two finite elements reduces to up to a single calculation of a function of their mutual visibility.
Also a parallel implementation of the algorithm is considered, which speedups the image calculation. The experiments done show that the computational time has linear dependence of the number of processors.