УДК 004.925

Р.В. Мальчева, М. Юнис, А. Джамиль
Донецкий национальный технический университет
raisa@cs.dgtu.donetsk.ua

Исследование влияния шага трассирования лучей и коэффициента различия в цвете на время выполнения формирования изображения

Рассмотрена реализация модифицированного алгоритма трассировки лучей с применением строчной межпиксельной интерполяции на SIMD и MIMD архитектурах. Описаны условия проведения исследований влияния шага трассирования и значения коэффициента различия в цвете на время формирования изображения. Приведены результаты исследований.

Трассировка лучей, пиксель, интерполяция, коэффициент различия в цвете, исследования

Введение

Алгоритм трассировки лучей относится к последовательным алгоритмам с N-повторяющимися действиями [1]. Это значит, что для каждого пикселя камеры выполняется набор одних и тех же операций по формированию луча, поиска ближайшего объекта на пути следования луча, расчет преломлений, отражений, поиск источников света, в которые первоначальный луч попадает для вычисления теней.

При использовании метода трассировки лучей ключевым является требование к производительности системы. Как показали исследования [1], от 75% до 95% времени формирования изображения приходится на поиск пересечения луча с элементами сцены. Существуют аппаратные решения, позволяющие ускорить анализ пересечения «луч-объект», в т.ч. за счет распараллеливания этапов алгоритма и данных [2,3,4].

Для современных мониторов с высокой разрешающей способностью разница в цвете смежных пикселей трассируемого изображения уменьшается, т.е. пиксели имеют приблизительно один и тот же цвет. На практике, сцена средней загруженности имеет около 70% пиксельных сегментов (области пикселей, которые различаются по цвету менее чем на 1%) и только 8% *крайних областей* (области пикселей, которые различаются по цвету более чем на 25%). Наличие этой информация позволяет трассировать не все пиксели экрана, а только их часть с некоторым шагом. А затем получать цветовые параметры для «пропущенных» пикселей методом строчной или блочной межпиксельной интерполяции [5,6,7].

Целью данной работы является исследование влияние шага трассирования лучей и коэффициента различия в цвете на время формирования изображения модифицированным алгоритмом с применением межпиксельной интерполяции.

Модификация алгоритма трассировки лучей

Основная идея модификации алгоритма трассировки состоит в следующем:

- трассировать меньшее количество пикселей, чем разрешение экрана;
- для получения значения непрослеженных пикселей применять линейную или блочную интерполяцию;
- для получения желаемого качества отображения варьировать шаг трассировки в зависимости от значения коэффициента максимального различия в цвете [8] трассируемых пикселей.

При строчной интерполяции (рис.1) для каждой строки экрана трассируются пиксели с некоторым шагом [6]. Например, на рис.1 показано, что закрашенные пиксели трассируются процессорными элементами (возможно, виртуальными) PE1, PE2 и PE3, соответственно.

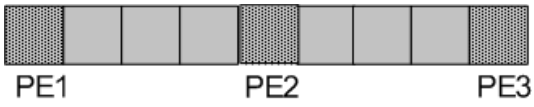

Рисунок 1 – Пример строчной интерполяции

Значения цветовых компонент для пикселей между ними (не закрашены) определяются линейной интерполяцией. В этом случае считаем, что трассировка выполняется с шагом, равным 3 пикселя. Для следующих строк алгоритм работает аналогично. Т.е. при строчной интерполяции не учитывается, что близкие по значения цветовые компоненты имеют не только пиксели, соседние по горизонтали, но и по вертикали.

При блочной интерполяции (рис.2) трассировка выполняется только для 4 пикселей, находящихся в углах блока [7]. Т.о. применяется

вертикальная, горизонтальная и диагональная интерполяция.

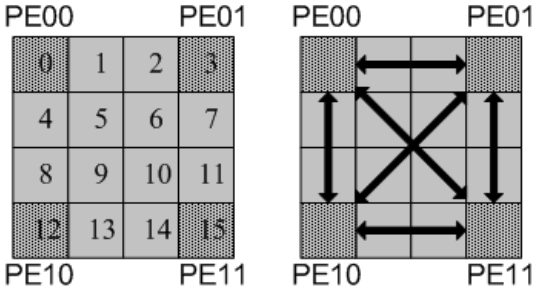

Рисунок 2 – Пример блочной интерполяции

Процессорные элементы PE00, PE01, PE10 и PE11 трассируют соответствующие пиксели экрана. Значения цветовых компонент для остальных пикселей формируются путем интерполяции:

01,02 - интерполяция (00,03) – горизонтальная;
04,08 - интерполяция (00,12) – вертикальная;
07,11 - интерполяция (03,15) – вертикальная;
13,14 - интерполяция (12,15) – горизонтальная;
05,10 - интерполяция (00,15) – диагональная;
06,09 - интерполяция (03,12) – диагональная.

В этом случае считаем, что выполняется блочная интерполяция с шагом, равным 2 пикселя.

Разработка модели алгоритма и подготовка тестовой сцены для выполнения исследований

Модель алгоритма трассировки лучей выполнена в виде Windows Forms приложения и DLL. Тестовая сцена содержит 3 примитива: 2 сферы и плоскость (рис.3). Сферы без текстур, задан только цвет материала, полупрозрачные, преломляющие свет, с возможностью отражения, поверхность бликовая. Плоскость имеет текстуру, способна к отражению.

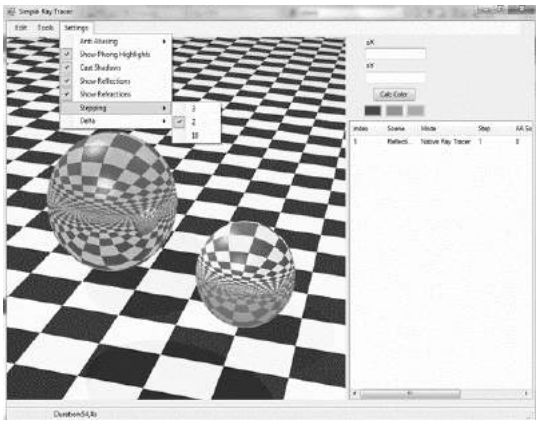

Рисунок 3 – Создание тестовой сцены

Разработанная модель поддерживает обработку изображений, с учетом отражающих, преломляющих, рассеивающих свойств материала. Поддерживает мягкие тени, сглаживание, отображение бликов.

Для проведения исследований использован программный пакет MPI.NET. MPI.NET – высокопроизводительная, легкая в использовании реализация технологии Message Passing Interface в среде Microsoft .NET.

MPI является стандартом для программной реализации параллельных алгоритмов для систем, использующих распределенную память (вычислительные кластеры). MPI.NET обеспечивает поддержку всех языков .NET платформы.

Библиотека включает в себя значительно расширенный функционал возможностей (например, автоматическую сериализацию объектов), который позволяет гораздо легче строить «параллельные» программы для выполнения на вычислительных кластерах [5].

Реализация и исследование модифицированного алгоритма трассировки лучей на системе с SIMD архитектурой

Как видно из краткого описания алгоритма, он полностью повторяет концепцию Single Instructure – Multi Data – SIMD.

SIMD алгоритм построен по классической схеме – мастер процесс раздает задания остальным процессам (Slave), и в последствии собирает результаты вычислений.

Мастер процесс в зависимости от количества Slave – процессов вычисляет, какие строки будет обрабатывать каждый Slave процесс, рассылает эти данные и начинает ожидание приема результатов вычислений от каждого Slave процесса.

Тестирование выполнялось для формирования фрагмента изображения размером 600 на 600 пикселей.

Схема выполнения и взаимосвязей процессов SIMD алгоритма изображена на рис.4.

В ниже приведенном фрагменте кода приведено деление данных между Slave процессами и их рассылка.

```
for (int i = 1;i<commSize;i++)
{
sb = new sendingBlock(master.AA,
0,0,master.Scene,master.Step);
sb.Begin=300/(commSize-1)*(i-1);
sb.End=sb.Begin+300/(commSize-1);
comm.Send<sendingBlock>(sb,i,0);
}
```

В следующем фрагменте приведен код, который обеспечивает прием данных в мастер процессе.

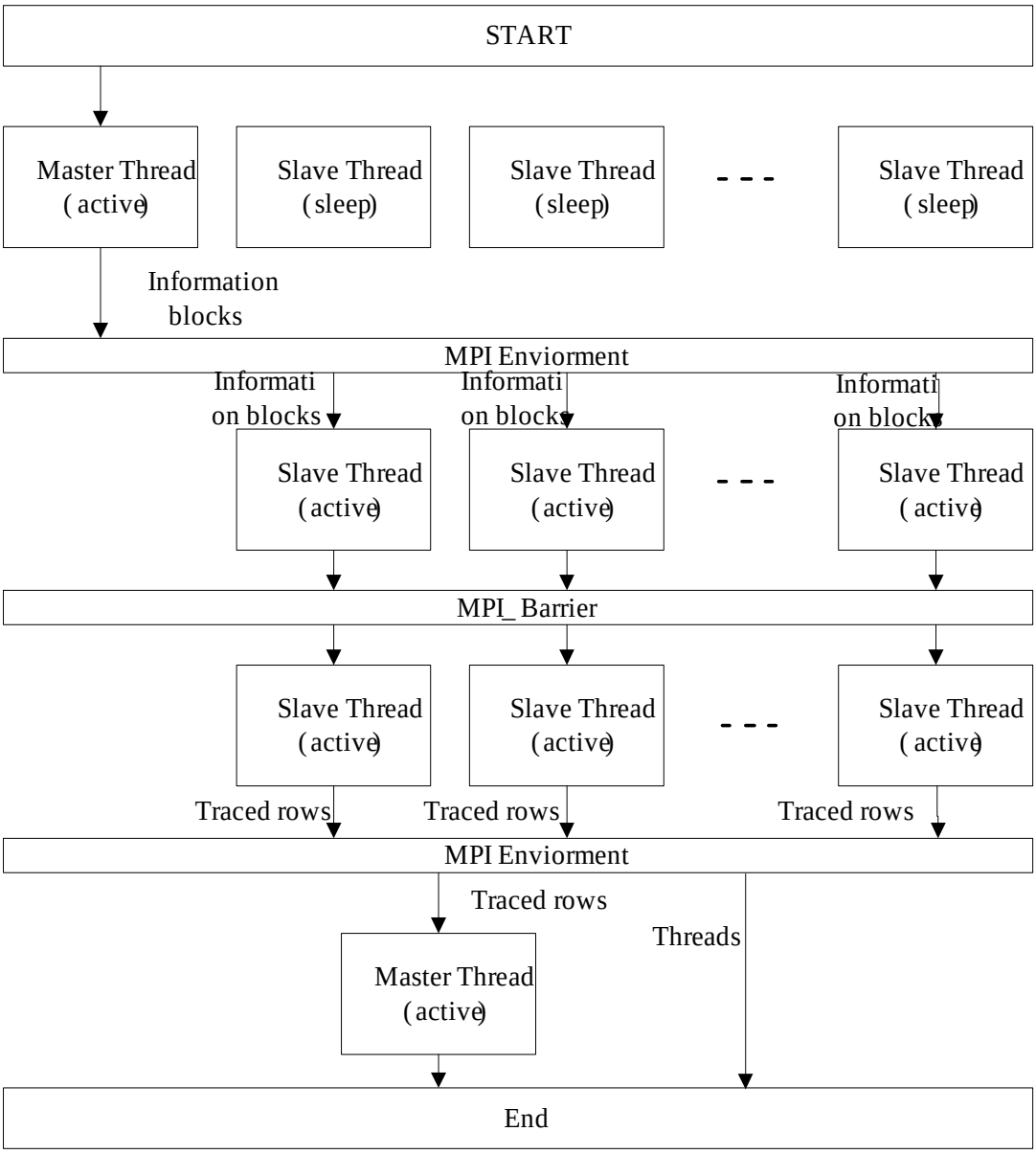

Рисунок 4 - Схема выполнения и взаимосвязей процессов SIMD алгоритма

```
  if (rank == 0)
  {int row_id = -1;
  Drawing.Brush br = null;
  int[]status_array=new int[comm.Size];
  CompletedStatus cs;
  Logging logcolor=new logging
("logcolor");
  for (UInt16 cnt_rcv = 0; cnt_rcv <
(300+(comm.Size-1)); cnt_rcv++)
  {comm.Receive<int>(Communicator.anySou
rce, TAG_INFORMATION,
  out row_id, out cs);
  comm.Receive<Color>(cs.Source,
TAG_DATA, ref color_array,out
  cs);
  f.setBitmapRows(row_id,
color_array);}}
```

Расчет времени выполнения трассировки сцены обеспечивает следующий код:

```
DateTime t1 = DateTime.Now;
…….Some RT code ……
DateTime t2 = DateTime.Now;
TimeSpan dt = t2.Subtract (t1);
```

Тестирование алгоритма проводилось на одноядерной и двуядерной системах. Параметры процессорной части двуядерной системы приведены на рис.5, параметры памяти – на рис.6.

Свойства ЦП	
Тип ЦП	Mobile DualCore Intel Core 2 Duo T5500, 1666 MHz (10 x 167)
Псевдоним ЦП	Merom-2M
Степпинг ЦП	M0
Наборы инструкций	x86, x86-64, MMX, SSE, SSE2, SSE3, SSSE3
Исходная частота	1667 МГц
Мин./макс. множитель ЦП	6x / 10x
Engineering Sample	Нет
Кэш L1 кода	32 Кб per core
Кэш L1 данных	32 Кб per core
Кэш L2	2 Мб (On-Die, ASC, Full-Speed)
Multi CPU	
CPU #1	Intel(R) Core(TM)2 Duo CPU T5450 @ 1.66GHz, 1666 МГц
CPU #2	Intel(R) Core(TM)2 Duo CPU T5450 @ 1.66GHz, 1666 МГц

Рисунок 5 – Параметры процессорной части тестовой станции

Свойства шины памяти	
Тип шины	Dual DDR2 SDRAM
Ширина шины	128 бит
Соотношение DRAM:FSB	10:5
Реальная частота	333 МГц (DDR)
Эффективная частота	666 МГц
Пропускная способность	10662 Мб/с

Рисунок 6 – Параметры памяти тестовой станции

Строчная линейная интерполяция позволяет обрабатывать сегменты пикселей любой длины. Но при этом необходимо контролировать потерю качества изображения с точки зрения восприятия человеческого зрения.

Для оценки изменения качества изображения использован коэффициент максимального расхождения в цвете, ***Eps*** - величина в цветовой системе LAB, на которую могут отличаться два цвета, чтобы человеческий глаз их воспринимал одинаково. По стандарту CIA 76 этот коэффициент равен 2,3 у.е. При проведении экспериментов использовались значения от 1-го до 5-ти, т.к. при динамическом отображении восприятие ухудшается.

Первоначальные эксперименты со значением шага трассирования, равным 10 пикселей, показали, что для 20-25% пиксельных сегментов необходимо выполнять уменьшение шага интерполяции в 2 (и больше) раз на основе полученных значений коэффициента максимального расхождения в цвете [5].

Поэтому при исследования использовалась длина сегмента в 3 и 4 пикселя, т.е. шаг трассирования равен 2 и 3 пикселя, соответственно.

Результаты исследований приведены на рис.7.

Как видно из диаграмм, модифицированный алгоритм трассировки с шагом, равным 2 пикселя, и интерполяцией дает прирост быстродействия в 12,5%. Использование шага трассировки в 3 пикселя увеличивает этот прирост до 15%.

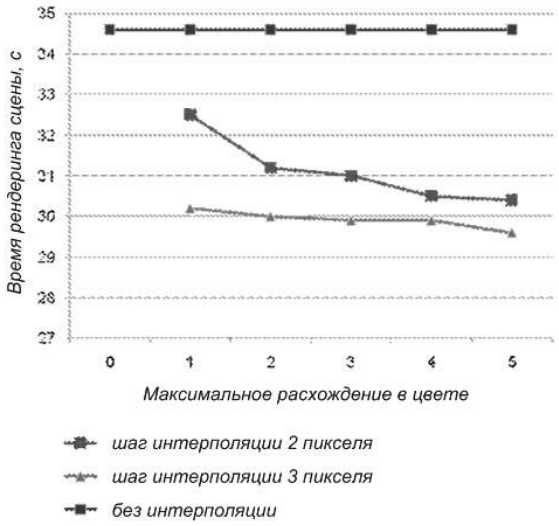

Рисунок 7 – Результаты исследований для SIMD архитектуры

Использование коэффициента максимального расхождения в цвете, равного 5, в среднем, дает 2% ускорения по сравнению с коэффициентом, равным 2.

Проверка сохранения качества изображения выполнена в системе Matlab 2008r2. При этом производилось попарно сравнение максимальных отличий в компонентах RGB (последовательно,1 – сравнение по красной компоненте, 2 – по зеленой, 3 – по синей) двух получаемых изображений.

```
d2 = imread('c:/temp/d2.bmp','bmp');
d5 = imread('c:/temp/d5.bmp','bmp');
>> max(max(dif_image))
ans(:,:,1) =
   3
ans(:,:,2) =
   2
ans(:,:,3) =
   5
```

Учитывая слабое восприятие глазом синего канала цветового спектра, результаты удовлетворяют коэффициенту максимального расхождения в цвете, близкому к 2. Т.е. при проведении экспериментов ухудшения качества изображения не наблюдалось.

Реализация и исследование модифицированного алгоритма трассировки лучей на системе с MIMD архитектурой

Реализация распараллеленного алгоритма трассировки лучей была построена, основываясь на объектно-пространственном делении сцены. Это означает, что каждый исполняющий элемент обрабатывает свою часть пространства (сцены) – свои наборы пикселей. Алгоритм построен таким образом, что процесс с рангом 0 (master thread,

MT) разделяет строки на равное количество между другими исполнительными процессами (slave thread, ST) и начинает от них ожидать сформированный массив пикселей.

Схема MIMD параллельного алгоритма трассировки лучей с интерполяцией по строкам приведена на рис.8.

Параллельный алгоритм был построен для двух исполняющих сред – Microsoft HPC Cluster 2008 и Scientific Linux 4.6. Для первой среды использовался язык программирования C# и библиотека – обертка MPI.NET, представляющая собой интерфейс доступа к функциям библиотеки MPICH2 из среды CLR. Эта реализация использовалась для отладки логики выполнения.

Мастер поток определяет количество процессов в коммуникаторе MPI. После этого выполняется анализ количества потоков, на основе чего формируется пакет, содержащий данные - задания каждому процессу. Затем происходит рассылка пакетов всем вычислительным процессам в коммуникаторе MPI. Далее каждый процесс выполняет инициализацию полученных данных и обработку своей части сцены. Отправка результатов осуществляется после обработки одной строки.

В качестве времени работы алгоритма фиксируется максимальное из времен работы процессов.

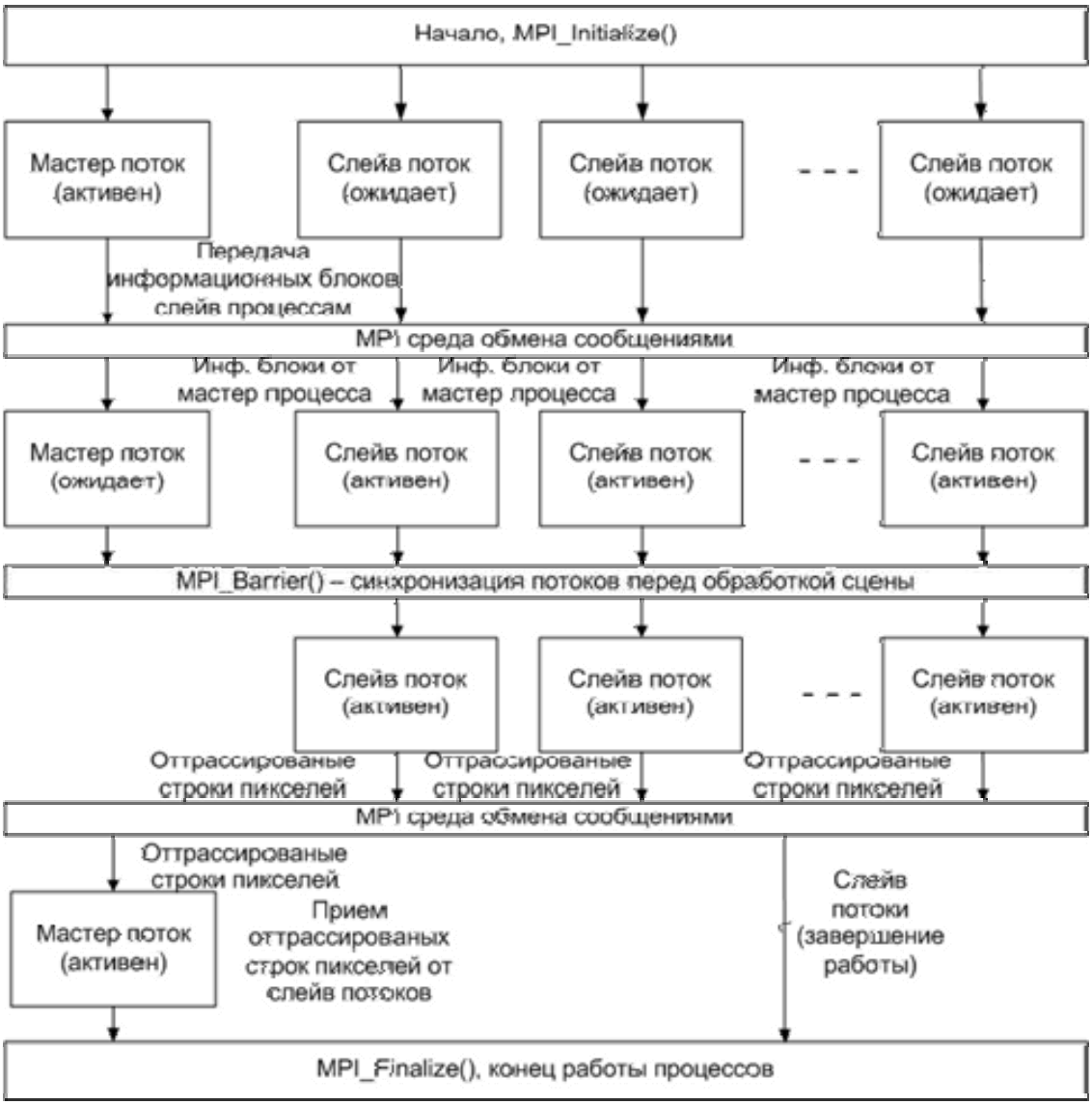

Рисунок 8 - Схема выполнения и взаимосвязей процессов MIMD алгоритма

Реализация модифицированного алгоритма трассировки лучей для MIMD архитектуры была выполнена на языке C++ с использованием библиотеки STL и библиотеки MPICH (реализация стандарта MPI) и протестирована на кластере NeClus при ФКНТ, КИ, ДонНТУ.

Исследовалось формирование изображений с размером 600 на 600 пикселей и 1800 на 1200 пикселей. Такой выбор обусловлен тем, что в настоящее время популярным является формат FullHD (Full High Definition), для которого в качестве стандартного используется изображение размером 1820 на 1200 пикселей.

При тестировании на одном процессе трассирование с разрешением 600 на 600 пикселей выполнялось за 47,7 секунды. При увеличении разрешающей способности до 1800 на 1200 пикселей время трассировки возросло до 98,7 секунды.

Следующий эксперимент выполнялся для модифицированного алгоритма трассировки лучей на 3-х, 4-х и 5-ти потоках обработки с использованием линейной строчной интерполяции с размером пиксельного сегмента, равным 3, для разрешающей способности 600 на 600 пикселей. Коэффициент расхождения в цвете варьировался от 1-го до 5-ти.

Результаты (рис.9) показали, что лучший результат получен для использования 5-ти потоков и коэффициента максимального расхождения в цвете, равного 3.

Такая комбинация дает прирост быстродействия около 11% по сравнению с реализацией на 5-ти процессах без использования модификации алгоритма и 82% по сравнению с реализацией на первом процессоре.

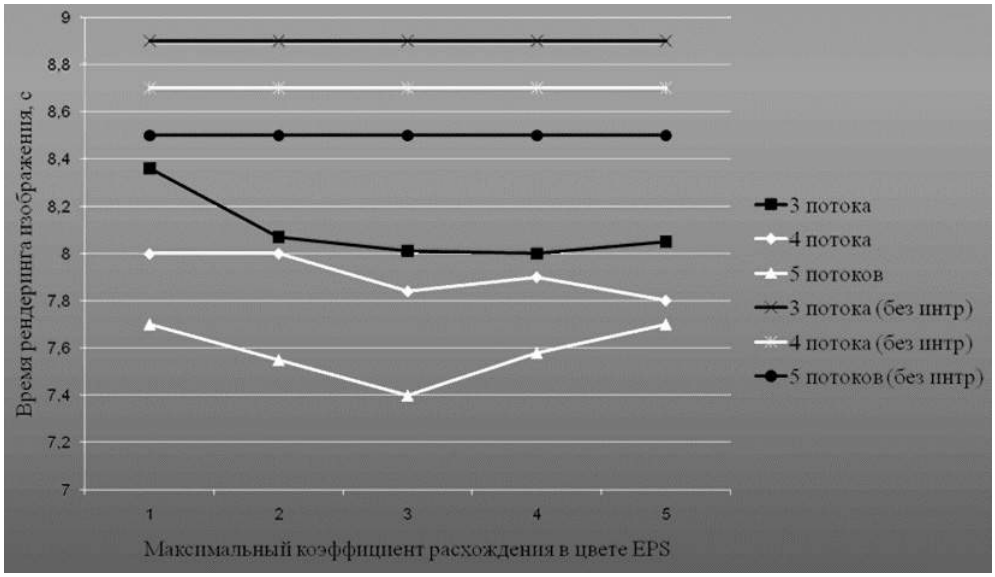

Рисунок 9 - Результаты исследований для MIMD архитектуры

При проведении исследований быстродействия модифицированного алгоритма трассировки лучей для разрешения экрана 1800 на 1200 пикселей также варьировался шаг трассировки (2 и 3 пикселя), коэффициент расхождения в цвете (от 1-го до 5-ти) и количество потоков обработки (3-5). Лучшие результаты получены для шага трассировки, равного 3, использования 5-ти потоков и коэффициента максимального расхождения в цвете, равного 4. Такая комбинация показала прирост около 16% по сравнению с реализацией на 5-ти процессах без применения модификации алгоритма (при трассировании всех пикселей).

Выводы

Проведенные исследования показали, что предложенная модификация метода трассировки лучей ускоряет процесс трассировки. Прирост быстродействия зависит как от шага трассирования, так и от коэффициента расхождения в цвете, при неудовлетворении которому производится уменьшение шага трассирования.

Направлением дальнейших исследований является экспериментальная проверка метода блочной интерполяции с целью определения параметров его эффективности.

Список литературы

1. Plunkett D.J. The Vectorization of a Ray-Tracing Algorithm for Improved Execution Speed / D.J. Plunkett, M.J. Bailey // IEEE Computer Graphics and Application. – 1985. – Vol.5, № 8. – P. 53-60.

2. Schmittler Jörg Realtime. Ray Tracing of Dynamic Scenes on an FPGA Chip / Jörg Schmittler // Computer Science, Saarland University, Germany. – 2004. – P.8.

3. SIMD Ray Stream Tracing / Wald Ingo, Christiaan P. Gribble, Solomon Boulos, Andrew Kensler // University of Utah. – 2007. – P.1.

4. Humphreys Greg TigerSHARK: A Hardware Accelerated Ray-tracing Engine / Humphreys Greg, Ananian C. Scott // Princeton University. – 2005. – P.3.

5. Malcheva R.V. The problems of modeling and rendering of the realistic complex scenes / R.V. Malcheva // Proceedings of ECCPM 2002, Portoroz. – 2002. – P. 537-538.

6. Мальчева Р.В. Применение строчной межпиксельной интерполяции для ускорения трассирования лучей / Р.В. Мальчева, Мохаммад Юнис // Материалы двенадцатого международного научно-практического семинара. Практика и перспективы развития партнерства в сфере высшей школы. – Донецк: ДонНТУ, 2011. – №12, Кн.2. - С. 72-74.

7. Мальчева Р.В. Применение блочной межпиксельной интерполяции для ускорения трассирования лучей / Р.В. Мальчева, С.А.Ковалев, Мохаммад Юнис // Материалы XVIII МНТК. Машиностроение и техносфера XXI века. – Донецк, 2011. – Т2. – С.189-191.

8.Robertson Alan R. Historical development of CIE recommended color difference equations / Robertson Alan R. // Color Research & Application. – 1990. – P.167-170.

9. Message Passing Interface for .NET [Электронный ресурс] – http://osl.iu.edu/ research/mpi.net/

10. Интерактивная трассировка лучей с использованием SIMD инструкций INTEL / Intel. – 2009: [Електроний ресурс]. – Режим доступу: http://software.intel.com/ru-ru/articles/interactive-ray-tracing/

Надійшла до редколегії 20.10.1011

Р.В. МАЛЬЧЕВА, М. ЮНІС , А. ДЖАМІЛЬ
Донецький національний технічний університет

Дослідження впливу кроку трасування промінів і коефіцієнту різниці у кольорі на тривалість виконання формування зображення

Розглянута реалізація модифікованого алгоритму трасування промінів з використанням порядкової міжпіксельної інтерполяції на SIMD і MIMD архітектурах. Описані умови проведення досліджень впливу кроку трасування й значення коефіцієнту різниці у кольорі на тривалість виконання формування зображення. Наведені результати досліджень.

Трасування промінів, піксель, інтерполяція, коефіцієнт різниці у кольорі, дослідження

R.V. MALCHEVA, M. YOUNIS, A. JAMIL
Donetsk National Technical University

Research of an Effect of Step of Ray-Tracing and Coefficient of Difference in Color Components on the Time of an Image Generation

The realization on SIMD and MIMD computer architectures of a modified ray-tracing algorithm with application of row interpixel interpolation is considered. The configuration of hardware and variants of initial values to investigate the effect of step of ray-tracing and coefficient of difference in color components on the time of an image generation are described. The results of researches are done.

Ray-tracing, pixel, interpolation, coefficient of difference in color components, investigation