УДК 004.925

А.А. БАРКАЛОВ[1], Р.В. МАЛЬЧЕВА[2], МОХАММАД ЮНИС[2]

[1]*Зеленогорский университет, Польша*
[2]*Донецкий национальный технический университет, Украина*

АППАРАТУРНАЯ РЕАЛИЗАЦИЯ АЛГОРИТМА СТРОЧНОЙ МЕЖПИКСЕЛЬНОЙ ИНТЕРПОЛЯЦИИ

Анализируются современные аппаратурные системы, использующие метод трассировки лучей. Рассматриваются алгоритмы строчной межпиксельной интерполяции с постоянным и переменным шагом. На примере показан механизм формирования цветовых компонент для промежуточных пикселей. Предлагается функциональная организация блока для выполнения алгоритма межпиксельной интерполяции. При реализации данного блока расходы ресурсов FPGA составили 56% логических ячеек и 78% памяти, что делает возможным расширение функций подсистемы введением блока интерполяции.

Ключевые слова: *трассировка лучей, разрешение экрана, интерполяция, длина сегмента, аппаратурная реализация.*

Введение

Трассировка лучей великолепно подходит для моделирования реального мира [1]. Однако повышение разрешающей способности устройств вывода и ужесточение требований к качеству формируемых изображений приводит к значительному времени формирования кадра. По мнению разработчиков графических систем традиционная архитектура визуализации исчерпала свой запас гибкости и масштабируемости, поэтому для решения будущих задач она уже не годится. Поэтому основные усилия направляются на разработку аппаратной инфраструктуры, поддерживающей трассировку лучей [2]. Недавно фирма Intel представила архитектуру Intel Larrabee, в которой для визуализации трехмерной графики будет применяться трассировка лучей.

Представленный на выставке CeBIT 2005 командой разработчиков из университета Саарленд рабочий прототип аппаратного видеоконтроллера выполняет трассировку лучей в реальном времени. Прототип, названный SaarCOR, реализован на микросхеме программируемой логики FPGA семейства Xilinx Virtex-II с немногим более чем 76 тыс. логических ячеек, неполными полутора сотнями аппаратных 18 битных умножителей и двумя с половиной мегабитами памяти. Эти возможности FPGA позволили реализовать в одном кристалле параллельную 64 поточную подсистему трассировки лучей, поддерживающую до 256 источников света, а также интерфейс шины PCI, VGA-интерфейс и аппаратные средства оценки производительности в реальном времени. Тактовая частота получившейся машины трассировки лучей составила 90 MHz. Для сравнения: чипсет NVidia GeForce 5900FX содержит 125 млн транзисторов, а производительность его четырех сотен вычислителей с плавающей точкой примерно в 50 раз выше, чем суммарная производительность SaarCOR. Тем не менее при программной SSE-оптимизированной реализации трассировщика лучей (система OpenRT), выполняющейся на ПК с процессором Intel Pentium 4 2,66 MHz и объемом оперативной памяти 1 GB, сцена, содержащая 187 млн треугольников, визуализируется в четыре с лишним раза дольше, чем при использовании SaarCOR [3].

1. Обоснование возможности применения алгоритма межпиксельной интерполяции

При синтезе изображения методом трассировки лучей для каждого элемента окна наблюдения (камеры) выполняется набор одних и тех же операций: формирование луча; поиск ближайшего объекта на пути следования луча; поиск источников света, в которые попадает луч, вычисление теней; расчет преломлений, отражений. При этом, как показали исследования [4], от 75% до 95% времени формирования изображения приходится на поиск пересечения луча с элементами сцены.

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 межпиксельной интерполяции [5], при этом для каждой строки экрана трассируются пиксели с некоторым шагом.

2. Строчная интерполяция с фиксированной и переменной длиной сегмента пикселей

Для получения желаемого качества отображения необходимо варьировать шаг трассировки в зависимости от значения коэффициента максимального различия в цвете трассируемых пикселей [6].

При применении алгоритма интерполяции трассировка может выполняться для пикселей с фиксированным шагом, тогда алгоритм работы будет иметь вид, показанный на рис. 1.

Длина сегмента интерполируемых пикселей может меняться. На рис. 2 приведен алгоритм межпиксельной интерполяции с применением сегментов переменной длины.

Назначение переменных, использованных при графическом представлении алгоритмов:

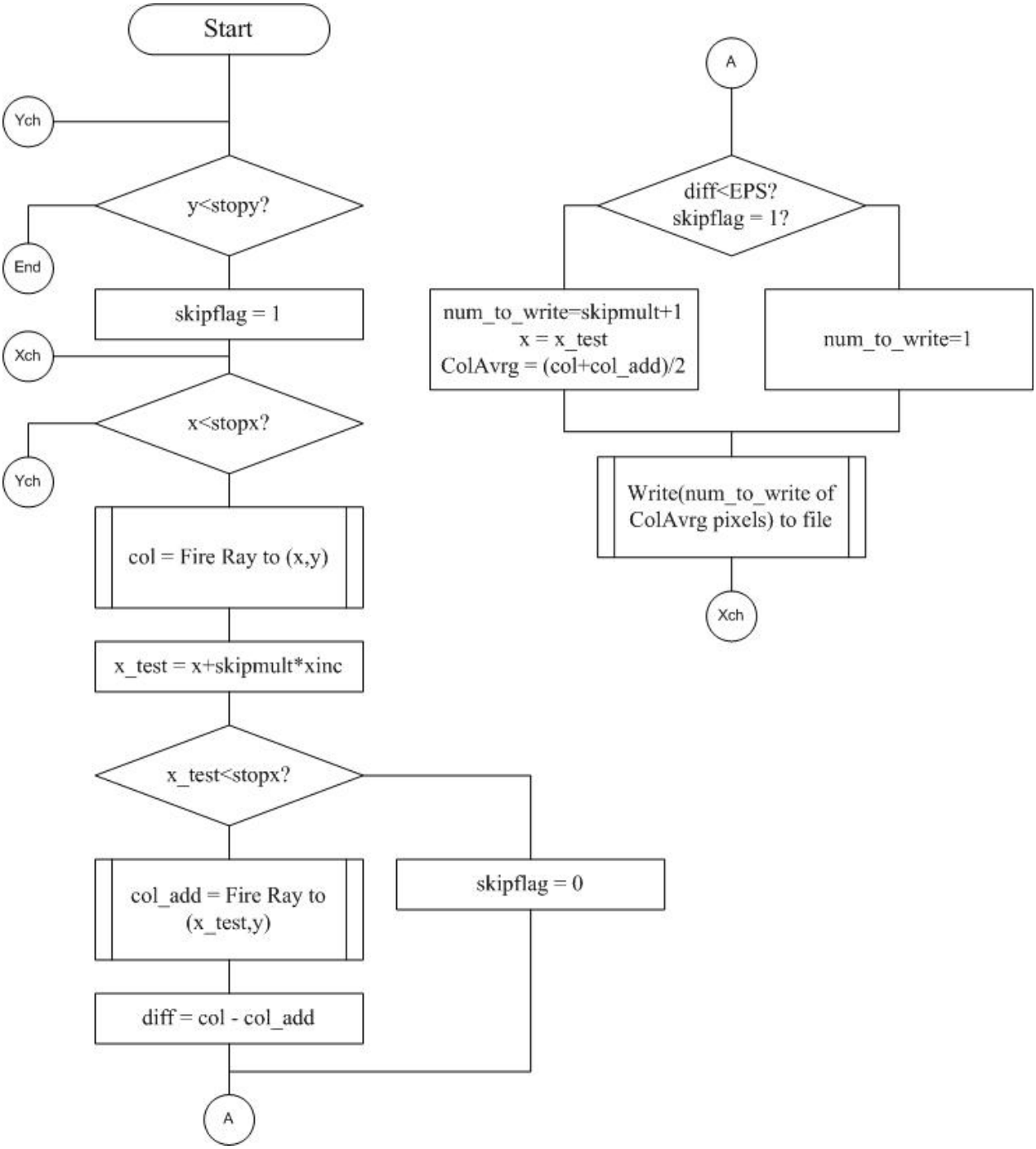

Рис. 1. Алгоритм межпиксельной интерполяции для фиксированной длины сегмента

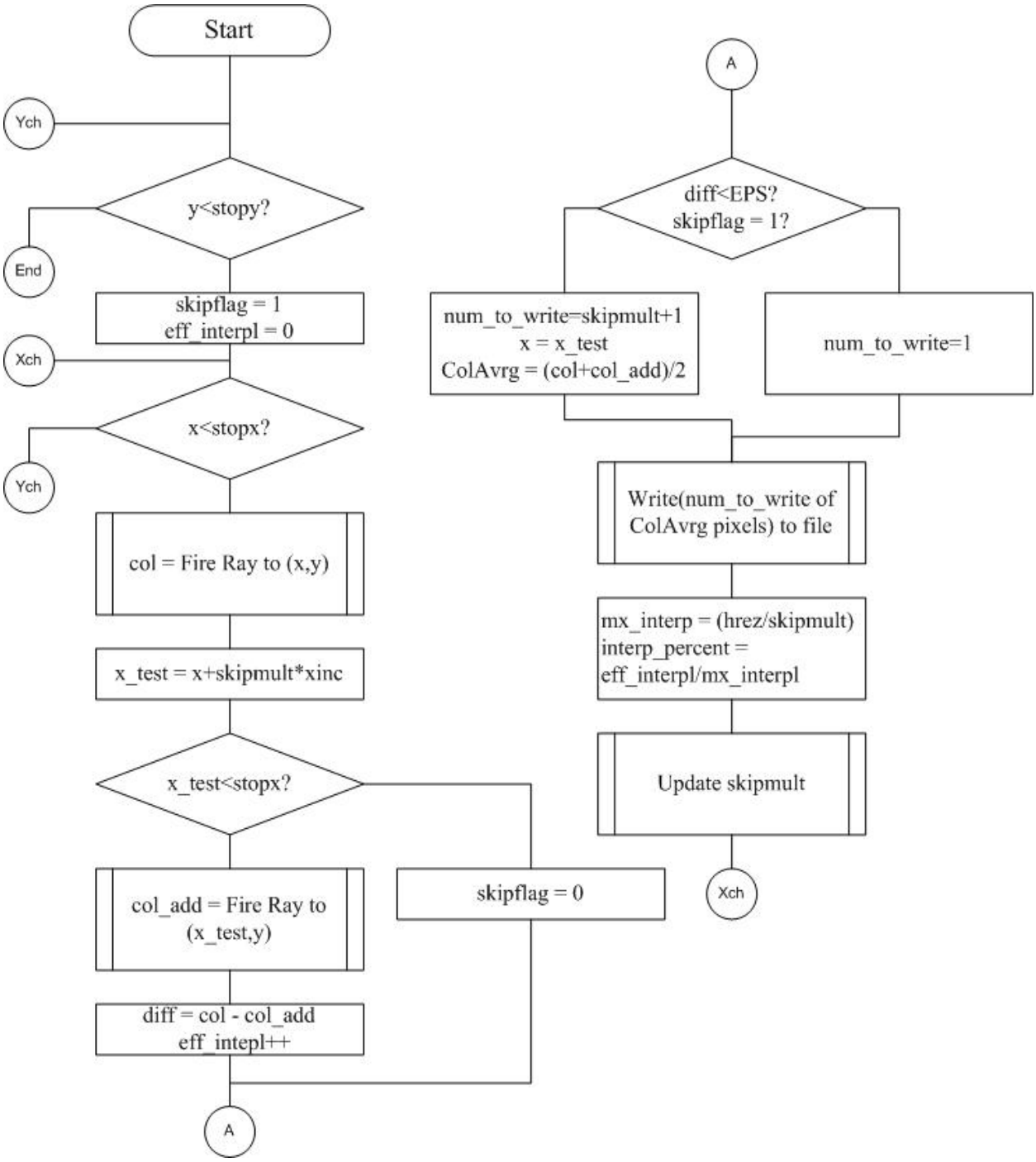

Рис. 2. Алгоритм межпиксельной интерполяции
с применением сегментов переменной длины

- stopx,stopy – разрешение изображения;
- x,y – координата трассируемого пикселя;
- x_test - x-координата пикселя, трассируемого для интерполяции;
- xinc – шаг между трассируемыми пикселями;
- skipmult – длина сегмента для интерполяции;
- skipflag – устанавливается в '0' при x_test>stopx, чтобы предотвратить выход за размеры изображения;
- EPS - установленная допустимая разница в цветовых компонентах для крайних пикселей сегмента;
- col,col_add – цветовые компоненты, заданные RGB-моделью.
-mx_interpl – максимальное число интерполяций, допустимое для заланных значений hres и skipmult;
-eff_interpl - количество успешнах интерполяций для последней строки;

-interpl_percent – качественная оценка шага интерполяции, используемая для установки параметра skipmult.

3. Аппаратурная реализация алгоритма

Хотя алгоритм с переменным шагом трассировки и выглядит как самое оптимальное решение, но на самом деле он не обеспечивает необходимого качества. Дело в том что, шаг для следующего сегмента корректируется на основании предыдущего, что приводит к возможному искажению изображения.

Исходя из этих доводов, для аппаратурной реализации (рис. 3) выбран алгоритм с фиксированным шагом трассировки.

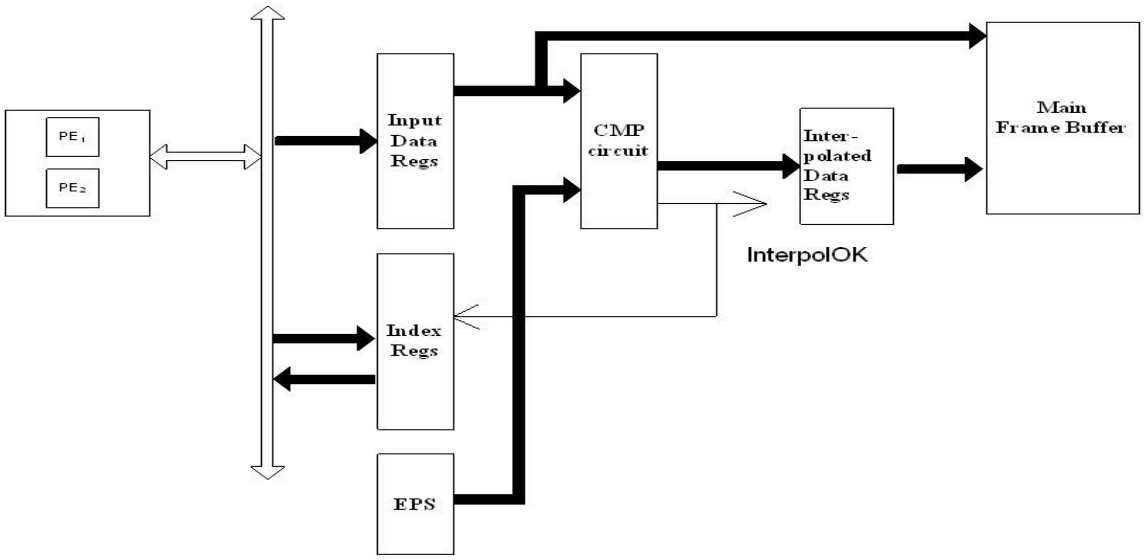

Рис. 3. Функциональная организация блока интерполяции

Рассмотрим работу блока на примере фрагмента строки, содержащего 8 пикселей (рис. 4).

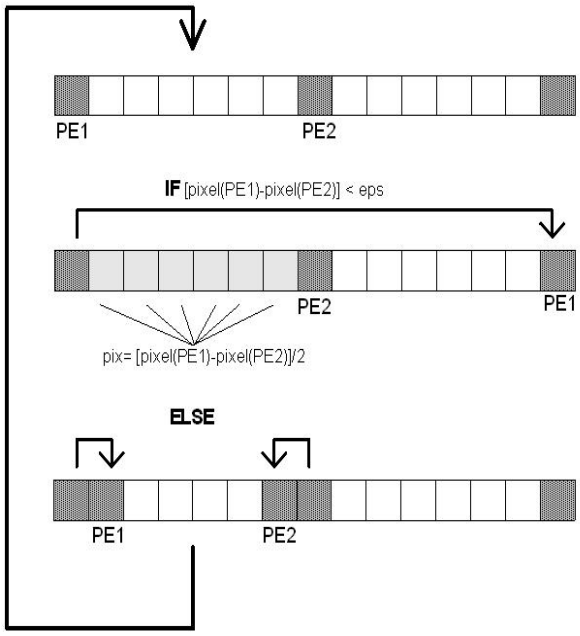

Рис. 4. Последовательность обработки пикселей

Каждому трассируемому пикселю изображения назначен свой процессорный элемент (PE) (возможно виртуальный); два смежных PE вычисляют крайние пиксели одного сегмента; у каждого PE есть идентификатор, который изменяется в пределах от 0 до максимального количества трассируемых пикселей строки N_{max}; все процессорные элементы имеют идентичную структуру.

1. PE1 и PE2 трассируют соответствующие пиксели и посылают два 24 битных значения цветовых параметров на шину данных. Полученные значения помещают во временную пару регистров.

2. При сравнении цветовых параметров генерируется флаг *InterpolOK*, который принимает значение 1 при их допустимом различии (меньше заданного *eps*), иначе - 0.

3. Если флаг установлен, то выполняется переход на интерполяцию. Иначе производиться перезагрузка пары процессорных элементов и пересчет соответствующих пикселей сегмента, обычно, через центральный пиксель сегмента.

4. Операция завершается по *InterpolOK*=1.

Количество процессорных элементов, необходимых для трассирования пикселей при использовании строчной межпиксельной интерполяции может быть рассчитано как:

$$N_{PEr} = \left\lceil \frac{x_m}{l_{segm} + 1} \right\rceil,$$

где x_m - разрешение экрана по координате x; l_{segm} - длина сегмента пикселей при интерполяции.

Исследования [6] показали, что для значитель-

ного числа изображений цветовые компоненты пикселей, трассируемых с шагом, равным 3, воспринимаются человеческим глазом как одинаковые.

Таким образом, для изображения растром (1024x1024), трассируемого сегментами по 4 пиксела, общее количество процессорных элементов уменьшается в 5 раз.

Выводы

Реализация блока интерполяции планируется в качестве расширения процессорного элемента, реализующего трассировку лучей. При его выборе интерес представляет SaarCOR. При его реализации расходы ресурсов FPGA составили 56% логических ячеек и 78% памяти, что делает возможным расширение функций подсистемы введением блока интерполяции.

Литература

1. Интерактивная трассировка лучей с использованием SIMD инструкций INTEL / Intel, 2009: [Електронний ресурс]. – Режим доступу: http://software.intel.com/ru-ru/articles/interactive-ray-tracing. – 16.01.2012 г.

2. Ильин, Ю. Nvidia заинтересовалась разработчиком raytracing-рендереров [Електроний ресурс] / Ю. Ильин. – Режим доступу: http://soft.compulenta.ru/ 357970. – 16.01.2012 г.

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

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

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

6. Мальчева, Р.В. Исследование влияния шага трассирования лучей и коэффициента различия в цвете на время выполнения формирования изображения [Текст] / Р.В. Мальчева, М. Юнис, А. Джамиль // Наукові праці ДонНТУ. Серія «Інформатика, кібернетика та обчислювальна техніка». – Донецьк: ДонНТУ, 2011. – № 14 (188). – С. 195 – 201.

Поступила в редакцию 10.02.2012

Рецензент: д-р техн. наук, проф., проф. кафедры АСУ Ю.А. Скобцов**,** Донецкий национальный технический университет, Донецк, Украина.

АПАРАТУРНА РЕАЛІЗАЦІЯ АЛГОРИТМУ ПОРЯДНОЇ МІЖПІКСЕЛЬНОЇ ІНТЕРПОЛЯЦІЇ

О.О. Баркалов, Р.В. Мальчева, Мохаммад Юніс

Аналізуються сучастні апаратурні системи, які використовують метод трасування прмінів. Розглядаються алгоритми порядкової міжпіксельної інтерполяції з постійним та змінним кроком. На прикладі показаний механізм формування кольорових компонент для внутрішніх пікселій. Пропонується функціональна організація блоку для виконання алгоритму міжпіксельної інтерполяції. При реалізації даного блоку витрати ресурсів FPGA склали 56% логічних комірок і 78% пам'яті, що робить можливим розширення функцій підсистеми введенням блоку інтерполяції.

Ключові слова: трасування промінів, розмір екрану в пікселях, інтерполяція, довжина сегменту, апаратурна реалізація.

HARDWARE REALIZATION OF ALGORITHM FOR THE INTERPIXEL INTERPOLATION

A.A. Barkalov ,R.V. Malcheva , Mohammad Younis

Graphical systems than are used Ray-tracing algorithms are discussed. Algorithms for horizontal interpixel interpolation with constant and non-constant steps are described. A mechanism of the color components formations for intermediate pixels is shown. A functional organization of unit to perform the interpixel interpolation algorithm is proposed. The unit costs of the implementation of this resource were 56% of the FPGA logic cells and 78% of memory, making it possible to extend the functions of the subsystem by the introduction of interpolation unit.

Key words: ray-tracing, size of screen in pixels, interpolation, length of segment, hardware realization.

Баркалов Александр Александрович – д-р техн. наук, проф., профессор Зеленогорского университета, Польша.

Мальчева Раиса Викторовна – канд. техн. наук, доцент, доцент кафедры компьютерной инженерии Национального технического университета, Донецк, Украина.

Юнис Мохаммад – аспирант кафедры компьютерной инженерии Национального технического университета, Донецк, Украина.