УДК 681.5.17

МЕТОД РЕКУРСИВНОГО ПОИСКА ЭЛЕМЕНТОВ ИЗОБРАЖЕНИЯ ФУНКЦИОНАЛЬНО ЗАДАННЫХ ПОВЕРХНОСТЕЙ

С. И. Вяткин

Институт автоматики и электрометрии СО РАН,
630090, г. Новосибирск, просп. Академика Коптюга, 1
E-mail: sivser@mail.ru

Рассмотрены проблема синтеза изображений высокого качества в реальном времени и способ задания трёхмерных объектов на основе функций возмущения. Предложен метод рекурсивного поиска элементов изображения функционально заданных объектов с использованием графических ускорителей. Показаны преимущества такого подхода перед фреймбуферным методом визуализации.

Ключевые слова: функции возмущения, теоретико-множественные операции, тайловая технология визуализации.

DOI: 10.15372/AUT20170307

Введение. Функционально заданные объекты рассматриваются как замкнутые подмножества евклидова пространства E^3, описываемые функцией $F(\mathbf{X}) \geq 0$, где F — непрерывная вещественная функция, $\mathbf{X} = (x, y, z)$ — задаваемая координатными переменными точка в E^3. Здесь $F(\mathbf{X}) > 0$ определяет точки внутри поверхности, $F(\mathbf{X}) = 0$ — точки на границе, $F(\mathbf{X}) < 0$ — точки, лежащие снаружи и не принадлежащие поверхности. Методы конструирования таких объектов разработаны достаточно хорошо, однако их визуализация в реальном масштабе времени остаётся проблемой. Сейчас существуют два подхода для быстрой визуализации функционально заданных объектов. В первом случае осуществляется полигонизация функциональных поверхностей [1] и дальнейшая визуализация полигональных моделей в реальном времени; во втором — визуализация функционально заданных объектов без предварительных преобразований с применением графических ускорителей [2].

Для визуализации функционально заданных объектов используются методы трассировки лучей [3]. Однако эти методы очень медленные даже при их реализации на графических ускорителях. Поэтому предпринимаются попытки ускорить вычисления различными способами. Так, например, в [4] были применены KD-деревья. Метод сферы-трэйсинга с нахождением наибольшего радиуса описан в [5]. Трассировка луча с анализом интервала для сложных функций предложена в [6]. Метод с переменным шагом при отслеживании лучей, реализованный на графическом ускорителе, рассмотрен в [7]. Более подробный анализ этих методов можно найти в [2].

Цель данной работы — создание метода эффективного рекурсивного поиска элементов изображения функционально заданных объектов на основе функций возмущения с применением графических акселераторов (далее — тайловая (tile) технология визуализации).

Функции возмущения. Для описания сложных геометрических объектов используются функции отклонения (второго порядка) от базовой квадрики [8]. Свободные формы строятся с помощью квадрик и представляются композицией базовой квадрики и возмущений:

$$F'(x, y, z) = F(x, y, z) + \sum_{i=1}^{N} f_i R_i(x, y, z). \qquad (1)$$

Здесь f_i — формфактор, $R(x, y, z)$ — возмущение:

$$R_i(x, y, z) = \begin{cases} Q_i^3(x, y, z), & \text{если } Q_i(x, y, z) \geq 0, \\ 0, & \text{если } Q_i(x, y, z) < 0, \end{cases} \tag{2}$$

где $Q(x, y, z)$ — возмущающая квадрика.

Для гладкой поверхности степень должна быть больше двух, как в (2). Это условие гарантирует непрерывность функции и её производной.

Теоретико-множественные операции. Геометрическая модель создаёт условия для конструирования объектов и их композиций различной сложности [9], используя множество геометрических операций Φ_i, определяемое математически следующим образом [10]:

$$M^1 + M^2 + \ldots + M^n \to M. \tag{3}$$

Для формирования моделей сложных объектов на базе функций возмущения применяются теоретико-множественные операции объединения и пересечения. Бинарная операция $(n = 2)$ (3) объектов G_1 и G_2 означает операцию $G_3 = \Phi_i(G_1, G_2)$ с определением

$$f_3 = \psi(f_1(\mathbf{X}), f_2(\mathbf{X})) \geq 0, \tag{4}$$

где ψ — непрерывная вещественная функция двух переменных.

Тайловая технология визуализации. В работе [2] предложен бинарный поиск элементов изображения функционально заданных объектов, относящийся к так называемому фрейм-буферному методу визуализации. Фрейм-буферный метод предусматривает хранение и модификацию каждого пиксела изображения в памяти кадра и растрирует примитивы в памяти произвольного доступа. Предлагаемая тайловая технология визуализации, т. е. виртуально-буферный метод — это растеризация примитивов в промежуточные порции экранной памяти и повторное использование виртуальных порций для конструирования полного кадра. Сравнение этих методов описано в [11].

Главный недостаток метода [2] заключается в том, что для объектов небольших и средних размеров в сцене выполняется много лишних вычислений. Это связано непосредственно с архитектурой графических ускорителей, поскольку условные переходы — очень затратные по времени вычисления операции, вместо которых применялась параллельная обработка данных. Для того чтобы программа работала действительно эффективно, надо учитывать особенности аппаратного обеспечения. В последнее время увеличение вычислительной мощности происходит именно за счёт параллельных вычислений. При этом отмечено, что центральный процессор бóльшую часть кадра простаивает. В данной работе предлагается повысить производительность визуализации за счёт увеличения транспортной задержки, являющейся главным недостатком тайловой технологии визуализации [11].

Процесс растрирования разбивается на два этапа и распределяется между центральным процессором и графическим ускорителем. Центральный процессор выполняет деление объектного пространства по четверичному дереву. В алгоритме поиска куб делится на меньшие части, для которых проводится тест на пересечение с объектом. Процесс деления можно разбить на две части. В первую очередь куб делится на четыре части в плоскости XY. Далее рассматривается каждая часть отдельно. Если пересечения с заданным объёмом не происходит, то из дальнейшего рассмотрения данная часть исключается. При наличии пересечения с другими частями аналогичная процедура деления повторяется. В общем случае процесс заканчивается, когда рассматриваемой части соответствует клетка

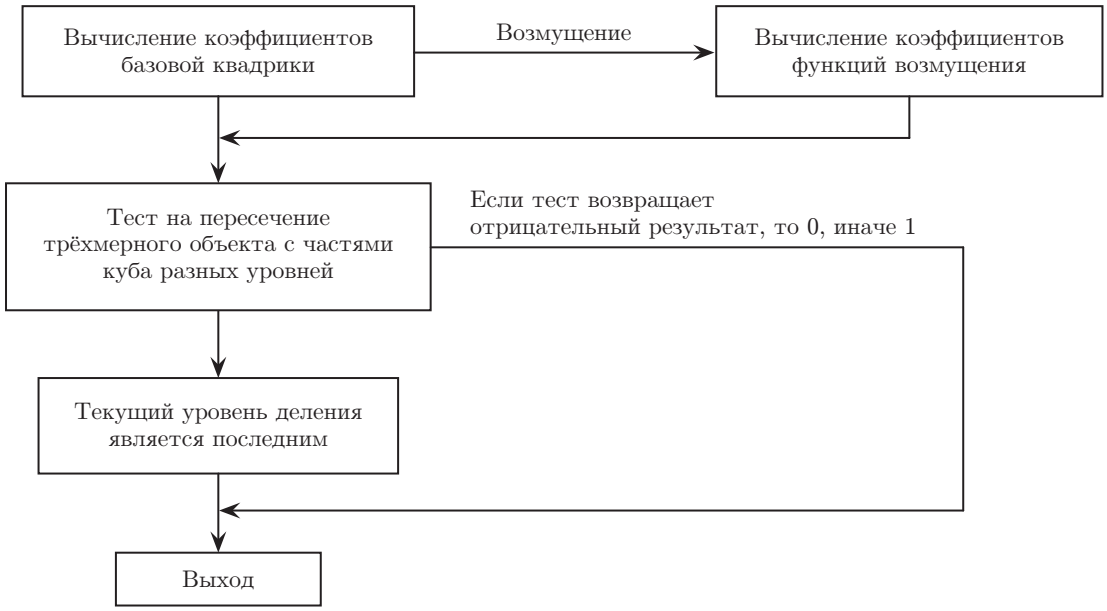

Рис. 1. Алгоритм четверичного деления квадрики

определённого размера (тайл). Преимущество такого подхода в том, что можно отбросить на ранней стадии бóльшие части куба, в которых нет заданного объекта. На рис. 1 показан алгоритм четверичного деления квадрики. Примитивами промежуточного описания являются фрагменты пересечения геометрических объектов с клетками.

Второй этап вычислений: обработка списка объектов, определение видимости и цвета пикселов — возложен на графический акселератор. На вход графического акселератора поступают фрагменты объекта. Далее фрагмент тестируется на пересечение с лучом, направленным вдоль оси Z, и производится бинарный поиск ближайшей точки пересечения луча с объектом [2]. Задача состоит в нахождении первой точки, в которой функция обращается в нуль. Выявив такую точку для каждого луча, можно вычислить координату z. Затем в каждом пикселе определяется нормаль. Имея все координаты и нормали в каждом пикселе, можно применить модель локального освещения. В итоге получится изображение гладкого объекта с учётом освещения.

Уменьшение времени на визуализацию достигается за счёт эффективного использования вычислительных ресурсов графического акселератора с архитектурой CUDA от компании NVIDIA. При реализации метода учитывается влияние скорости работы с памятью. Максимально задействованы регистры и совместно используемая память. Во всех остальных случаях процессоры работают с общей памятью графического акселератора.

В функции графического акселератора входило вычисление координат точек поверхностей, нормалей и освещения. Центральным процессором выполнялись геометрические преобразования, а также растрирование примитивов в сетке тайлов и формирование списка фрагментов с вычислением всех необходимых параметров. Для визуализации был взят интерфейс прикладного программирования DirectX. Тестирование производилось на процессорах Intel Core2 CPU E8400 3.0 GHz и GPU 470 GTX. На рис. 2 показаны результаты тестирования зависимости времени вычисления кадра от числа задаваемых функций возмущения для конкретного теста двух методов (см. таблицу). Из диаграммы видно, что в среднем время вычисления уменьшилось на порядок. При этом транспортная задержка увеличилась вдвое. Отметим, что производительность возрасла не только для объектов среднего и небольшого размера, но и для крупных с большим числом возмущений. Поскольку объект разбивается на тайлы, количество возмущений фрагментов сокращается.

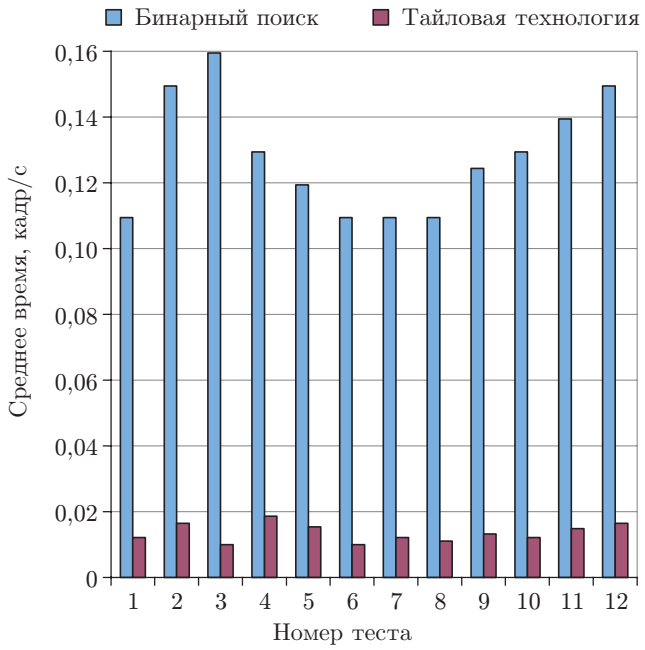

Рис. 2. Диаграмма тестов

Распределение числа задаваемых возмущений для объектов кадра в различных тестах

№ теста	Количество объектов в кадре	Типы возмущений*							
		1	2	3	4	5	6	7	8
1	1	4	—	—	—	—	—	—	—
2	22	4	4	4	1	1	4	3	3
3	22	4	4	4	3	3	1	3	1
4	3	1	1	3	—	—	—	—	—
5	3	1	1	4	—	—	—	—	—
6	10	2	—	—	—	—	—	—	—
7	1	4	—	—	—	—	—	—	—
8	1	5	—	—	—	—	—	—	—
9	1	6	—	—	—	—	—	—	—
10	1	7	—	—	—	—	—	—	—
11	1	8	—	—	—	—	—	—	—
12	1	9	—	—	—	—	—	—	—

*Тесты для равного количества объектов в кадре и с равным числом функций возмущения для каждого объекта отличаются типом функции возмущения.

Заключение. В данной работе предлагается эффективный метод растрирования (поиска элементов изображения поверхности), суть которого сводится к разделению экрана на клетки и конвейеризации вычислений с помощью промежуточного описания кадра в виде списка примитивов. Разбиение вычислений на две фазы с использованием промежуточного описания кадра позволяет достичь максимальной производительности на этапе пиксельных вычислений, требующих наибольших ресурсов и определяющих производительность системы в целом.

Предложенные способ задания трёхмерных объектов и метод визуализации имеют преимущества перед известными подходами. К основным достоинствам предлагаемых способа и метода следует отнести: простоту вычисления точек поверхности с быстрым поиском и отбраковкой областей, не занятых объектами сцены; уменьшение количества поверхностей для описания криволинейных объектов.

СПИСОК ЛИТЕРАТУРЫ

1. **Vyatkin S. I.** Polygonization method for functionally defined objects // Intern. Journ. Automat., Control and Intell. Syst. 2015. **1**, N 1. P. 1–8.
2. **Вяткин С. И.** Метод бинарного поиска элементов изображения функционально заданных объектов с применением графических акселераторов // Автометрия. 2014. **50**, № 6. С. 89–96.
3. **Hart J. C.** Ray tracing implicit surfaces // SIGGRAPH-93 Course Notes: Design, Visualization and Animation of Implicit Surfaces. 1993. N 25. P. 1–16.
4. **Foley T., Sugerman J.** KD-tree acceleration structures for a GPU raytracer // Proc. SIGGRAPH/Eurographics Workshop on Graphics Hardware. N. Y.: ACM, 2005. P. 15–22.
5. **Hart J. C.** Sphere tracing: a geometric method for the antialiased ray tracing of implicit surfaces // The Visual Comput. 1994. **12**, N 10. P. 527–545.
6. **Mitchel D.** Robust ray intersection with interval arithmetic // Proc. on Graphics Interface. Toronto: Canadian Information Processing Society, 1990. P. 68–74.
7. **Liktor G.** Ray tracing implicit surfaces on GPU // Comput. Graph. and Geometry. 2008. **10**, N 3. P. 36–53.
8. **Вяткин С. И.** Моделирование сложных поверхностей с применением функций возмущения // Автометрия. 2007. **43**, № 3. С. 40–47.
9. **Вяткин С. И.** Метод интерактивного моделирования функционально заданных объектов без предварительной триангуляции поверхности // Автометрия. 2015. **51**, № 6. С. 70–78.
10. **Вяткин С. И.** Преобразования функционально заданных форм // Программные системы и вычислительные методы. 2014. № 4. С. 484–499.
11. **Вяткин С. И., Долговесов Б. С., Фомичев В. М.** Растеризационные методики и архитектуры систем визуализации реального времени // Тр. 17-й Междунар. конф. по компьютерной графике и ее приложениям «Графикон-2007». М.: МГУ, 2007. С. 211–218.

Поступила в редакцию 10 января 2017 г.