

МЕТОД УМЕНЬШЕНИЯ ИТЕРАЦИЙ В АЛГОРИТМАХ СИНТЕЗА ИЗОБРАЖЕНИЙ РЕАЛЬНОГО МАСШТАБА ВРЕМЕНИ

ГУСЯТИН В.М.

Излагается метод уменьшения количества шагов в итерационных алгоритмах поиска точки пересечения для синтеза изображений. При построении графических спецпроцессоров такой алгоритм позволяет увеличить параллелизм вычислений с одновременным уменьшением длины конвейера, что повышает производительность системы в целом.

1. Актуальность

Применение итерационных алгоритмов для синтеза изображений в графических спецпроцессорах систем визуализации тренажеров транспортных средств позволяет спроектировать вычислитель с параллельно-конвейерной структурой высокой производительности [1]. Число шагов итераций — n определяется требуемой точностью вычислений. При этом на каждом шаге в двоичной системе счисления определяется один разряд искомого n -разрядного числа. Для повышения производительности конвейерной структуры необходимо каждый шаг итераций, в связи с существенным временем его выполнения, подразделять на m тактов конвейера. Общее число тактов загрузки такого конвейера от начала поступления данных до получения результата будет равно:

$$N_3 = m \cdot n. \quad (1)$$

При непрерывном поступлении данных на вход искомым результатом на выходе конвейера вычисляется на каждом такте. В случае “разрыва” в поступлении данных образуется “разрыв” в потоке выходных данных результата, что приводит к снижению производительности конвейерной структуры. Максимальное время задержки при длительности такта конвейера t равно

$$T_3 = \tau \cdot m \cdot n. \quad (2)$$

Из (2) следует, что одним из путей сокращения максимального времени задержки является уменьшение общего числа тактов конвейера. Поэтому разработка методов уменьшения количества шагов в итерационных алгоритмах поиска точки пересечения для синтеза изображений является актуальной задачей.

2. Основные положения метода

Для построения алгоритма поиска точки пересечения в соответствии с методом уменьшения итераций используем основные понятия и определения, принятые автором в работе [1]. Алгоритм будем

строить таким образом, чтобы на каждом шаге итерационного процесса (ИП) определялся более чем один разряд искомого числа N . Обозначим количество разрядов, определенных на одном шаге, n_1 . Для упрощения полученных соотношений будем также полагать, что n — число, кратное n_1 , т.е.

$$n = K \cdot n_1, \quad (3)$$

где K — число шагов итераций нового алгоритма.

В соответствии с [2] координаты точки пересечения $P(X, Y, Z)$ проекционного луча (ПЛ) с плоскостями базовой системы координат (с/к XYZ) могут быть найдены из векторного уравнения $\vec{r}_p = \vec{r}_h + \vec{v}_p$. По аналогии с [1] запишем координаты произвольной точки на ПЛ в параметрической форме:

$$x = X_h + X_p \cdot t, y = Y_h + Y_p \cdot t, z = Z_h + Z_p \cdot t, \quad (4)$$

здесь X_p, Y_p, Z_p — проекции на оси с/к XYZ вектора \vec{v}_p , соответствующего ПЛ в методе обратного трассирования в [3]: t — параметр, $t \in \{0, \dots, 1\}$.

Алгоритм строим для того же класса поверхностей, что и в [1]. По аналогии считаем, что до начала работы итерационного алгоритма поверхности, с которыми пересекается ПЛ, предварительно определены. Уравнение поверхности запишем в неявной форме:

$$F(X, Y, Z) = 0. \quad (5)$$

Совместное решение уравнений (4), (5) для нахождения точки пересечения $P_1(X, Y, Z)$ ПЛ с поверхностью будем искать итерационным методом. Для построения ИП параметр t запишем в виде:

$$t_k = t_{k-1} + \eta_k \cdot \delta_k, \quad (6)$$

где k — номер шага итераций, $k \in \{1, 2, \dots, K\}$; δ_k — квант k -го шага итераций, $\delta_k = 2^{-n_1 \cdot k}$; η_k — положительный коэффициент, определяющий количество используемых квантов на k -м шаге итераций.

В отличие от [1], в соответствии с предлагаемым методом, в нашей задаче будем искать экстремум-минимум модуля величины $F(X, Y, Z)$ среди элементов множества $F_j^k(X, Y, Z)$ (в дальнейшем, для упрощения записи — F_j^k), вычисляемых на каждом k -м шаге следующим образом:

$$F_j^k = F_j^k \{ [X_h + X_p(t_{k-1} + j \cdot \delta_k), \\ [Y_h + Y_p(t_{k-1} + j \cdot \delta_k), [Z_h + Z_p(t_{k-1} + j \cdot \delta_k) \} \}, \quad (7)$$

где j — индекс, задающий номер элемента множества F_j^k , $j \in \{0, 1, \dots, J\}$, $J = 2^{n_1}$.

По аналогии с [1] определим элементы множества F_j^k как параметры-индикаторы (ПИ). Совместный анализ величин и знаков ПИ позволяет определить стратегию движения в процессе итераций. Цель каждого k -го шага – определение отрезка на ПЛ (в дальнейшем – выделенный отрезок), в границах которого может находиться результат совместного решения уравнений (4), (5). Здесь важно отметить, что на каждом следующем шаге ИП выделенный отрезок уменьшается в J раз. Величина η_k в (6) определяется на каждом шаге ИП по результатам анализа F_j^k .

3. Алгоритм вычислений

В соответствии с методом строим итерационный алгоритм, который на каждом шаге определяет n_1 разрядов искомого числа N , начиная со старших. Каждый шаг ИП состоит из следующих вычислений.

3.1. Для выбранной поверхности $F(X, Y, Z)=0$ вычисляются элементы множества F_j^k . При этом возможны два случая.

Случай 1. Все элементы множества имеют один знак. Тогда анализ элементов F_j^k состоит из следующих этапов.

Этап 1. Определяется элемент множества с минимальным значением модуля, т.е. $|F_j^k|_{\min}$. Пусть это будет некоторый j -й элемент.

Этап 2. Определяются модули первых разностей:

$$\Delta F_j^k = |F_j^k - F_{j+1}^k|, \Delta F_{j-1}^k = |F_{j-1}^k - F_j^k|.$$

Этап 3. Индексы наименьшего модуля разности, определенной на этапе 2, принимаются в качестве индексов выделенного отрезка. Пусть это будут индексы j и $j+1$. Назовем их соответственно: начальный и конечный индексы.

Этап 4. Коэффициент η_k на этом шаге равен

$$\eta_k = (j)_k, \quad (8)$$

где $(j)_k$ – начальный индекс выделенного отрезка на k -м шаге.

Случай 2. Элементы множества F_j^k имеют разные знаки. При этом, если поверхность $F(X, Y, Z)=0$ не замкнута (например, плоскость), то набор отношений для F_j^k может иметь такой характер распределения знаков:

$$F_0^k > 0; F_1^k > 0; \dots; F_j^k > 0; F_{j+1}^k \leq 0; \dots; F_j^k < 0, \text{ либо}$$

$$F_0^k < 0; F_1^k < 0; \dots; F_j^k \leq 0; F_{j+1}^k > 0; \dots; F_j^k > 0. \quad (9)$$

Тогда для отношений (9) имеем один выделенный отрезок с индексами $j, j+1$ и коэффициент $\eta_k = (j)_k$.

Если поверхность замкнута (например, сфера), отношения могут иметь вид:

$$F_0^k > 0; F_1^k > 0; \dots; F_j^k > 0; F_{j+1}^k \leq 0; F_{j+2}^k < 0; \dots;$$

$$F_{j+5}^k \leq 0; F_{j+6}^k > 0; \dots; F_j^k > 0. \quad (10)$$

Для отношений (10) имеем два выделенных отрезка с индексами $j, j+1$ и $j+5, j+6$ и соответственно коэффициентами $\eta_k = (j)_k$ и $\eta_k = (j+5)_k$.

В случае (10) далее могут выполняться два ИП для каждого из выделенных отрезков, если поверхность прозрачна и необходимо найти две точки пересечения, либо – один ИП для выделенного отрезка с меньшим индексом, если поверхность непрозрачна.

3.2. Вычисляется параметр t_k с учетом полученных η_k на k -м шаге. Начальное условие $t_{\text{нач}} = t_0 = 0$.

3.3. Вычисления по п.п. 3.1, 3.2 продолжаются до $k=K$. Если ни на одном из шагов итераций не выполнялись отношения (9), (10), то точка пересечения ПЛ с поверхностью $F(X, Y, Z)=0$ отсутствует. При выполнении отношений (9), (10) хотя бы на одном шаге точка пересечения существует, причем, координаты этой точки равны координатам любой из границ выделенного отрезка на последнем шаге ИП.

4. Результаты

Предлагаемый метод построения итерационных алгоритмов поиска точки пересечения ПЛ с поверхностью является обобщением алгоритма, изложенного в [1], который получается при $n_1=1$. Метод позволяет уменьшить число итераций и, как следствие, длину конвейера. В предельном случае $K=1$.

С уменьшением числа итераций увеличивается параллелизм вычислений, что в целом повышает производительность системы визуализации. При этом, естественно, увеличиваются аппаратные затраты. При $K=1$ число параллельных ветвей вычисления равно 2^n+1 .

Для рассмотренного метода оказывается возможным оптимизировать структуру вычислителя по критериям числа шагов итераций и аппаратным затратам без потери точности вычислений.

Литература: 1. *Гусятин В.М.* Итерационный алгоритм синтеза изображения в растровой графике реального масштаба времени // Радиоэлектроника и информатика. 1998. №3. С.81-83. 2. *Гусятин В.М.* Математическая модель геометрических преобразований для спецпроцессоров растровой графики // Радиоэлектроника и информатика. 1997. №1. С. 86-87. 3. *Иванов В.П., Батраков А.С.* Трехмерная компьютерная графика. М.: Радио и связь, 1995. 224 с.

Поступила в редколлегию 07.09.2000

Рецензент: д-р техн. наук, проф. Кривуля Г.Ф.

Гусятин Владимир Михайлович, канд. техн. наук, доцент кафедры электронных вычислительных машин ХТУРЭ. Научные интересы: теория и практика построения спецпроцессоров растровых графических систем реального времени. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 40-93-54, 66-61-22.