методики показали высокую эффективность применения УВХ совместно с БИС АЦП параллельного типа. Динамическая погрешность АЦП на частотах до 7 МГц — ±1 квант.

ЛИТЕРАТУРА

1. **Плата** для быстрого аналого-цифрового преобразования.— Электроника, 1981, № 16.
2. **Stuart R. Michaels.** Mend flash-converter flaws with track/hold core.— END, 1981, N 30.
3. **Ефремов А. И., Касперович А. Н., Литвинов Н. В., Шалагинов Ю. В.** Широкополосный аналого-цифровой преобразователь.— Автометрия, 1981, № 6.
4. **Беломестных В. А., Вьюхин В. Н., Касперович А. Н.** Об одном способе экспериментального определения динамических свойств быстродействующих АЦП.— Автометрия, 1976, № 5.
5. **Нил М., Мьюто А.** Динамический контроль аналого-цифровых преобразователей.— Электроника, 1982, № 4.

Поступило в редакцию 31 января 1984 г.

УДК 681.3.06

Б. Х. ЗИНГЕР

(Новосибирск)

О РЕАЛИЗАЦИИ ПРОСТРАНСТВЕННОЙ СОРТИРОВКИ ПО ПРИОРИТЕТАМ

В данной статье приводится краткое описание реализации алгоритма предварительной пространственной сортировки по приоритетам, предложенного в [1].

Назначение сортировки по приоритетам состоит в том, чтобы выполнить заранее (до визуализации) многие вычисления, связанные с удалением невидимых поверхностей. Они проводятся независимо от положения и ориентации наблюдателя: один раз для всех возможных изображений сцены. Такой подход позволяет резко сократить объем вычислений во время генерации изображения. При этом достигается столь значительное сокращение вычислительных затрат, что становится возможной генерация трехмерных визуальных сцен в реальном масштабе времени.

Входной информацией для рассматриваемого алгоритма служит база данных, содержащая описание модели визуальной сцены. Любой объект аппроксимируется набором плоских многоугольников (граней). Каждая грань представлена упорядоченным набором вершин с соответствующими координатами в трехмерном пространстве.

В процессе сортировки граней для обеспечения приемлемого быстродействия и возможности обработки данных большого объема необходима гибкая организация структур данных и выбор адекватных методов обработки.

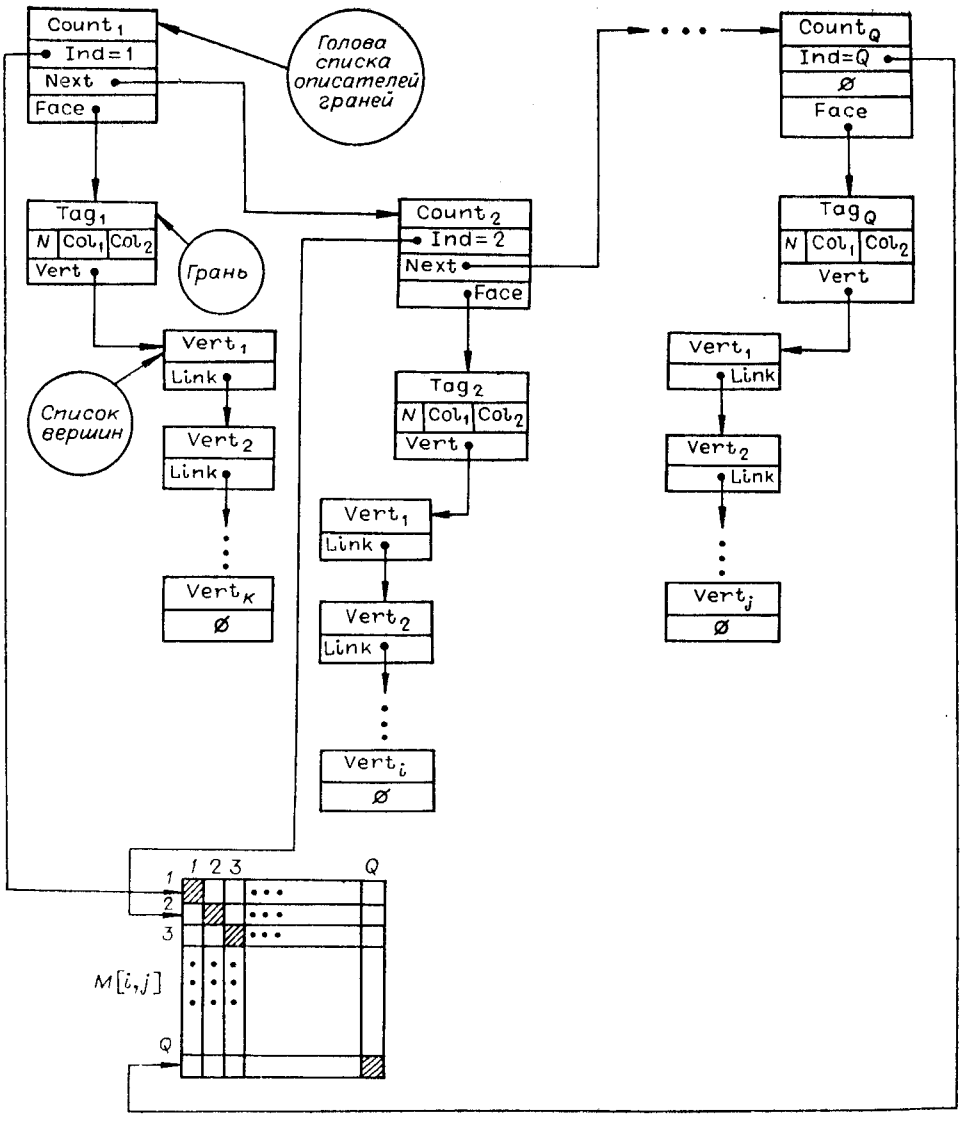
В общем случае при реализации алгоритма сортировки объем требуемой памяти может превысить выделенные для задачи ресурсы, поэтому для обеспечения работоспособности программы (возможно, с меньшей эффективностью) необходимо повторное использование освободившихся областей памяти. Списочная структура дает возможность организовать сложную иерархию памяти и позволяет решить проблему обработки и представления данных с нестандартной организацией [2]. Под каждый элемент списка динамически выделяется память, так что в случае необходимости она может быть перераспределена.

Таким образом, для обеспечения эффективной обработки данных сложной организации необходимо, во-первых, использовать списки, во-вторых, обеспечить повторное использование свободных областей памяти, в-третьих, временно неиспользуемые данные целесообразно хранить на устройствах прямого доступа [3].

На рисунке показан линейный список описателей граней, каждый из которых представляет собой структуру, описывающую определенные атрибуты грани и содержащую указатель грани. Грань, в свою очередь, является структурой, включающей остальные параметры грани и указатель на список вершин. Вершины организованы в линейный список.

Описатель грани состоит из четырех полей: поле счетчика (Count) — целое число, указывающее на количество граней, закрывающих данную; поле ключа (Ind) — целое число (номер строки в матрице отношений); поле указателя (Next) — указатель на следующий элемент списка описателей; поле указателя (Face) — указатель грани.

Грань — это структура памяти из пяти полей: нормаль (N) — нормаль грани; тега (Tag) — битовая информация о режимах обработки; Col_i — цвет грани, осве-



щенной источником света; Col_2 — цвет грани в тени; Vert — указатель на первую вершину.

Вершина описывается структурой четырех типов. Во всех вариантах используется поле указателя (Link) — указатель на следующую вершину. Описание типа вершины может варьироваться (например, вершина для аппроксимации криволинейного объекта, для описания ограниченной текстуры и т. п.). Каждая вершина определена координатами (x, y, z) и текстурой. Текстура вершины задается нормированным вектором (псевдонормалью) и двумя трехкомпонентными скалярами — цвет при освещении и цвет в тени. Текстурная информация в обработке не участвует, однако при рассечении граней новые вершины должны быть снабжены правильной текстурой, которая линейно распространяется на ребра.

Для описания отношений между гранями применяется двумерный битовый массив $M[i, j]$ — «матрица отношений». Если грань Γ_i закрывает грань Γ_j ($\Gamma_i > \Gamma_j$ [1]), то $M[i, j] := 1$.

Во время сортировки списка граней основной список разбивается на подсписки, которые обрабатываются рекурсивно. В целях эффективного использования оперативной памяти подсписки, временно не участвующие в обработке, хранятся на внешних носителях. В качестве вспомогательной памяти используется память на магнитных дисках, организованная в виде файла прямого доступа [4]. Подсписки обрабатываются в порядке, обратном порядку поступления. В дальнейшем структуру данных такого типа будем называть «файловым стеком».

Заметим, что изложенный метод исключает образование «дыр» — неиспользованных участков дисковой памяти.

Алгоритм сортировки. Основу алгоритма сортировки составляет рекурсивная процедура SORT. В данном случае рекурсивный метод обработки является наиболее естественным, так как он отражает особенности структуры данных.

Задача состоит в построении такого порядка граней, при котором для любого положения наблюдателя после исключения граней заднего плана множество оставшихся граней должно быть отсортированным в порядке «удаленности» от точки наблюдения (будем полагать одну грань более удаленной по сравнению с другой, если она закрывает вторую грань от наблюдателя). Такое множество граней назовем упорядоченным.

Из множества граней $\{\Gamma_1, \ldots, \Gamma_Q\}$ с помощью отношений, введенных в [1], выделяется упорядоченное подмножество. Если такое подмножество найдется, то все его элементы будут переписаны в файл отсортированных граней и исключены из исходного множества. После исключения упорядоченного подмножества оставшееся подмножество не содержит максимального элемента.

Опишем процедуру выделения упорядоченного подмножества граней. Суть ее состоит в следующем. Каждая грань сравнивается со всеми остальными, т. е. грань Γ_i, $i = \{1, \ldots, n\}$, сравнивается со всеми гранями Γ_j, $j = \{i+1, \ldots, n\}$. Для этого организуются два вложенных цикла (внешний — по i, а внутренний — по j). На каждом шаге после сравнения граней выполняются следующие действия: вычисляются элементы $M[i, j]$ i-й строки и $M[j, i]$ i-го столбца матрицы отношений; увеличивается на единицу поле Count в описателе грани D_i или D_j (соответственно при $\Gamma_i > \Gamma_j$ или $\Gamma_j > \Gamma_i$).

После окончания внутреннего цикла по значению счетчика Count в поле описателя D_i можно определить, является ли грань Γ_i максимальной (т. е. может ли грань Γ_i подаваться на отображение раньше всех других граней). В этом случае: а) грань Γ_i переписывается в файл отсортированных граней и удаляется из исходного списка граней; б) структуры памяти, соответствующие грани Γ_i, предварительно подготовленные для повторного использования, присоединяются к списку свободных участков памяти; в) по i-му столбцу матрицы отношений определяются все грани Γ_j, где $j = \{1, \ldots, n, i \neq j\}$, которые закрывали грань Γ_i; г) для каждой такой грани $\Gamma_j (\Gamma_i > \Gamma_j)$ счетчик Count в описателе D_j уменьшается на единицу и бит $M[j, i]$ в матрице отношений устанавливается равным единице.

Заметим, что после удаления грани Γ_i в подмножестве граней $\{\Gamma_1, \ldots, \Gamma_{i-1}\}$ возможно возникновение максимальных элементов, т. е. граней, в описателях которых поле Count равно нулю. Тогда для таких граней необходимо проделать те же действия, что и для грани Γ_i.

Другими словами, по завершении внутреннего цикла каждый раз вызывается рекурсивная процедура, которая на подмножестве граней $\{\Gamma_1, \ldots, \Gamma_i\}$ определяет все грани, которые можно упорядочить. Затем продолжается внешний цикл до тех пор, пока i не достигнет значения n.

Чтобы отсортировать подмножество, не содержащее максимального элемента, проведем разделяющую плоскость через первую по порядку грань. Эта плоскость разделит все объектное пространство (список граней) на два полупространства (два списка), причем сама грань, через которую проводится разделяющая плоскость, попадает в полупространство, где будет заведомо максимальной. Таким образом, каждая грань либо целиком лежит в каком-то полупространстве, либо рассекается и дает по представителю в каждом из них. Далее организуются два файловых стека, соответствующих подразделению граней на полупространства. Оба списка готовятся для повторного использования и присоединяются к списку свободных участков памяти. К каждому полупространству применяется процедура выделения упорядоченного подмножества. Все описанные выше операции повторяются до полного исчерпания списков, и результатом работы алгоритма являются упорядоченные подмножества граней, перемежающиеся разделяющими плоскостями, в обоих полупространствах которых имеются свои упорядоченные подмножества.

Для подмножества граней, не содержащих циклов, определение приоритетов сводится к «топологической сортировке» [5]. Для выделения упорядоченного подмножества граней используется алгоритм, подобный описанному в [5], но значительно усовершенствованный. Приведенный выше алгоритм статической сортировки граней по приоритетам был реализован в ИАиЭ СО АН СССР в 1980 г.

ЛИТЕРАТУРА

1. **Зингер Б. Х., Талыкин Э. А.** Предварительная пространственная сортировка — основа алгоритма удаления невидимых поверхностей для систем приоритетного типа.— Автометрия, 1983, № 6.
2. **Фостер Дж.** Обработка списков.— М.: Мир, 1974.
3. **Катпан Г.** Операционные системы.— М.: Мир, 1976.
4. **Джадд Д. Р.** Работа с файлами.— М.: Мир, 1975.
5. **Кнут Д.** Искусство программирования для ЭВМ.— М.: Мир, 1976, т. 1.

Поступило в редакцию 19 мая 1984 г.