случае организации параллельных вычислений за счет аппаратурного усложнения можно добиться существенного ускорения обработки. Параллельная форма реализации ДЦФ позволяет сократить время задержки выходной последовательности относительно входной в pq раз, что во многих практических приложениях оказывается чрезвычайно важно.

Заключение. Применение ТЧП для вычисления двумерной цифровой свертки обеспечивает абсолютную точность вычислений вследствие отсутствия ошибок округления на всех этапах обработки. Применение параллельной формы реализации ДЦФ и скачущего алгоритма ТЧП позволяет устранить ограничения, накладываемые на характеристики свертки, как то: размеры обрабатываемого массива изображения, динамический диапазон входных сигналов D, и существенно уменьшить задержку реакции ДЦФ на входное воздействие. При этом возрастают аппаратурные затраты на реализацию фильтрации и число операций, требуемое для ее осуществления.

СПИСОК ЛИТЕРАТУРЫ

1. **СБИС** для распознавания образов и обработки изображений /Под ред. К. Фу.—М.: Мир, 1988.
2. **Макклеллан Дж. Х., Рейдер И. М.** Применение теории чисел в цифровой обработке сигналов.—М.: Радио и связь, 1983.
3. **Нуссбаумер Г.** Быстрое преобразование Фурье и алгоритмы вычисления сверток.—М.: Радио и связь, 1985.
4. **Блейхут Р.** Быстрые алгоритмы цифровой обработки сигналов.—М.: Мир, 1989.
5. **Рабинер Л., Гоулд Б.** Теория и применение цифровой обработки сигналов.—М.: Мир, 1978.

Поступила в редакцию 14 июня 1991 г.

УДК 629.7.058.74

А. А. Лубков, В. В. Полубинский

(Новосибирск)

ДИНАМИЧЕСКОЕ РАСПРЕДЕЛЕНИЕ ПАМЯТИ ДАННЫХ В ССВО ВЫСОКОЙ ПРОИЗВОДИТЕЛЬНОСТИ

Предлагается метод динамического распределения памяти для хранения модели окружающей обстановки, основанный на специфике работы систем синтеза визуальной обстановки (ССВО). Метод позволяет значительно уменьшить объем быстродействующего ОЗУ при незначительном аппаратном усложнении.

Локальная память в ССВО. Одной из важнейших характеристик систем синтеза визуальной обстановки является объем памяти для хранения модели окружающей обстановки объектов. По мере роста производительности систем синтеза требования к этой памяти значительно ужесточаются. Для ССВО высокой производительности (100 000 граней за кадр) объем памяти должен составлять не менее 24—160 Мбайт при быстродействии 200 нс [1]. Использование подобной памяти малоприемлемо по причине ее значительных габаритов и стоимости.

В ССВО в любой момент времени отображается только то, что видит наблюдатель. Таким образом, нет необходимости держать все данные в ОЗУ. Желательно использовать эту особенность работы ССВО для уменьшения объема оперативной памяти.

В данной работе проводится анализ структурных единиц информации в системах синтеза и предлагается организация локальной памяти данных для ССВО высокой производительности. Для этого следует хранить информацию

на НМД, а в ОЗУ передавать ее часть, описывающую отображаемый участок сцены.

Структурные информационные единицы данных в ССВО. На основании анализа описания окружающей обстановки в существующих и разрабатываемых системах синтеза [2] рассмотрим основные информационные единицы данных с учетом специфики работы ССВО.

Самой мелкой единицей информации для отображения является грань. Совокупность приоритетно упорядоченных граней — сегмент [3]. Группа сегментов с общей матрицей перемещения — объект.

Процессор базы данных осуществляет обход двоичного дерева разделяющих плоскостей и формирует приоритетно-упорядоченную последовательность адресных ссылок на сегменты [1]. Далее сегменты выбираются из памяти данных и передаются на геометрический процессор. При этом для правильного построения изображения необходимо, чтобы все сегменты отображаемого объекта находились в локальной памяти данных. Для быстрой выборки данных необходим простой механизм вычисления их адресов. Это либо расположение их в последовательных ячейках памяти, либо наличие списка связи ячеек памяти, содержащих описание одного сегмента.

Для организации локальной памяти данных желательно иметь механизм быстрого обновления данных. Обновление определяется двумя операциями: чистки и подкачки. Они не обязательно должны осуществляться совместно, т. е. для проведения операции подкачки не обязательно проводить чистку. Чистку можно осуществлять в случае уменьшения свободного пространства в памяти ниже определенного предела. Подкачка производится при необходимости иметь ту или иную порцию информации в локальной памяти. Критерии осуществления чистки (подкачки) в ССВО предлагаются в [1].

Скорость обновления информации памяти данных. Чистка данных может осуществляться системой синтеза достаточно просто. Для этого вместо ссылки на начальный адрес в команде «Сегмент» в памяти дерева надо поставить признак «отсутствует», а также передать блок памяти, отводимый для этого сегмента, в список свободного пространства. Все операции в этом случае производятся в оперативной памяти, что обеспечивает их достаточное быстродействие.

Подкачка данных связана с дисковыми операциями. Это создает существенные задержки. Время выполнения такой операции ($T_о$) складывается из двух величин: времени установки головки ($T_у$) и времени передачи данных ($T_п$). $T_у$ не зависит от объема передаваемых данных, а $T_п$ прямо пропорционально этому объему:

$$T_о = T_у + Wt.$$

Здесь Wt — время передачи, W — объем информации, t — время передачи единицы информации (здесь и далее все скоростные характеристики НМД приводятся для непрерывных файлов). Следовательно, время, затраченное на передачу единицы информации с диска: $T_е = t + (T_у/W)$, уменьшается при увеличении количества информации, передаваемой за одну операцию. Следовательно, для увеличения пропускной способности канала подкачки желательно передавать большие порции данных.

Верхним ограничением на объем данных для подкачки являются ограничения, присущие конкретному дисковому накопителю, используемому в системе (максимальный размер непрерывного файла). Например, для накопителя CDC-9462 верхний предел составляет примерно 60 Кслов по 16 разрядов.

Если существуют пространственно небольшие близкорасположенные объекты, то целесообразно объединить соответствующие им описания в одну порцию подкачки, так как они попадут в пирамиду видимости практически одновременно. Назовем информацию, соответствующую одной дисковой операции, пакетом подкачки. Таким образом, пакет подкачки включает в себя сегменты и служебную информацию, необходимую для осуществления подкачки/чистки.

В то же время существуют объекты, описание которых занимает небольшой объем памяти, но которые имеют значительную пространственную протяженность (дороги, реки и др.). Такие объекты нецелесообразно объединять в один пакет подкачки, так как они никогда не попадут в один кадр изображения. Следовательно, существуют пакеты со значительно отличающимися объемами.

Доступность элементов данных для функционирования подкачки/чистки в ССВО. Для осуществления операций подкачки, чистки и передачи данных из локальной памяти для дальнейшей обработки необходимо иметь доступ к следующим элементам:

1. Для отображения надо знать начальный адрес и размер каждого сегмента, находящегося в локальной памяти. Также должно быть известно, загружен ли сегмент в память.

2. Для загрузки пакета подкачки с диска необходимо иметь пространство в локальной памяти, знать его начальный адрес и быть уверенным, что данный пакет может быть целиком размещен в этом пространстве. При этом следует запомнить начальный адрес расположения пакета в памяти для последующего отображения и чистки.

При загрузке пакета сегменты располагаются по адресам, заранее неизвестным, а для отображения нужно знать их начальные адреса. По этой причине пакет подкачки на диске должен включать дополнительную информацию. Модель мира включает в себя дерево команд, фиксированное для этой базы данных. Это позволяет заранее знать адреса команд «Сегмент» в памяти дерева [1] и иметь в пакете подкачки на диске вместе с данными еще и адреса расположенных в дереве команд. При загрузке каждого сегмента в соответствующей ему команде устанавливается адрес расположения его описания в локальной памяти.

3. Для чистки необходим начальный адрес пакета подкачки в локальной памяти и его размер. Это дает возможность быстро возвратить занимаемую пакетом память в свободную область.

Для динамического распределения и обеспечения доступа к описанным элементам наиболее подходит списочная структура, позволяющая наиболее полно использовать память и иметь доступ к любому необходимому элементу. Введя в набор команд процессора базы данных операцию «Подкачка» [1], получаем возможность манипуляции с пакетами.

Наиболее важными полями этой команды являются:

— идентификатор пакета (это поле передается в ЭВМ при необходимости загрузки; по таблицам ЭВМ находит адрес пакета на диске и инициирует передачу данных в систему синтеза);

— адрес пакета в локальной памяти (поле определено, если пакет загружен); необходим для проведения чистки;

— наличие в памяти (указывает на то, что данные загружены и могут быть использованы для отображения);

— размер пакета; необходим для операции чистки.

Списочная структура позволяет удобно вести учет свободного пространства. При этом достаточно знать только адрес входа в список, длину его, а также адрес последнего слова в списке. При подкачке данные загружаются в начало списка (соответственно модифицируется адрес входа), а при чистке пространство, занимаемое очищаемыми пакетами, присоединяется к его концу. Длина списка соответственно модифицируется.

Организация локальной памяти. Списочная структура позволяет сохранить быстродействие локальной базы данных, но усложняет оборудование. Необходимая для организации списков память ссылок может иметь значительный объем. Возможным компромиссом может быть разбиение памяти на блоки и связывание в списки блоков.

Для организации динамического распределения предлагается разбиение локальной памяти на равные блоки с длиной, кратной степени 2 (1, 2, 4 и т. д.). В устройство добавляется память ссылок, позволяющая связывать эти блоки в произвольной последовательности. Каждому блоку соответствует ячейка в

памяти ссылок так, что номер блока является адресом соответствующей ему ячейки в памяти ссылок (рис. 1). В каждой ячейке содержится ссылка на следующий блок списка. На рис. 2 показана часть списка, организованного с помощью памяти ссылок. Последовательность блоков такова: 0, 1, 2, 50, ..., 100 и т. д. Стрелками показан порядок связи блоков. Первоначально все блоки связаны в единый список свободных блоков (ССБ).

Для каждого пакета подкачки существует свой описатель, находящийся в памяти дерева и содержащий ссылки на начальный и конечный блоки списка соответствующего пакета, а также количество занимаемых им блоков. Если пакет не загружен в локальную память, то адресные ссылки недействительны. Описатель пакета подкачки (начало, конец, длина) находится в соответствующей команде «Подкачка» [1]. При подкачке пакета из ССБ поочередно извлекаются свободные блоки и подключаются к списку пакета подкачки. При этом длина ССБ декрементируется. На рис. 3 показано изменение связи блоков при подкачке очередного блока данных (блок *L* передается пакету подкачки). Ссылка на начало списка (блок *a*) устанавливается в соответствующей команде «Подкачка». В этой же команде по окончании операции подкачки будет установлена ссылка на последний блок пакета. В каждой команде «Подкачка» содержится также количество требуемых для пакета блоков, которое является константой. Таким образом, после окончания подкачки в этой команде содержится вся необходимая информация о пакете подкачки (ссылки на начало и конец, длина). Эта информация используется при чистке.

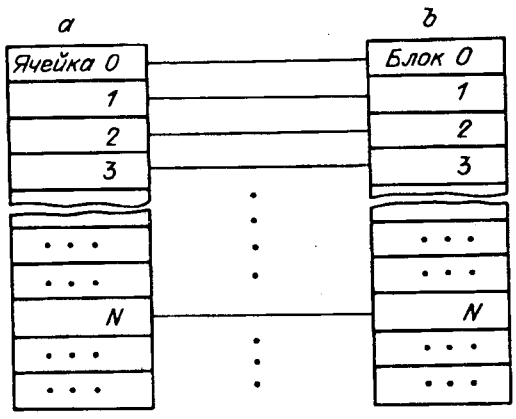

Рис. 1. Память ссылок (*a*) и локальная память данных (*b*)

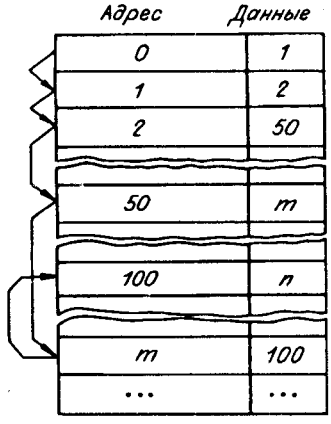

Рис. 2. Память ссылок

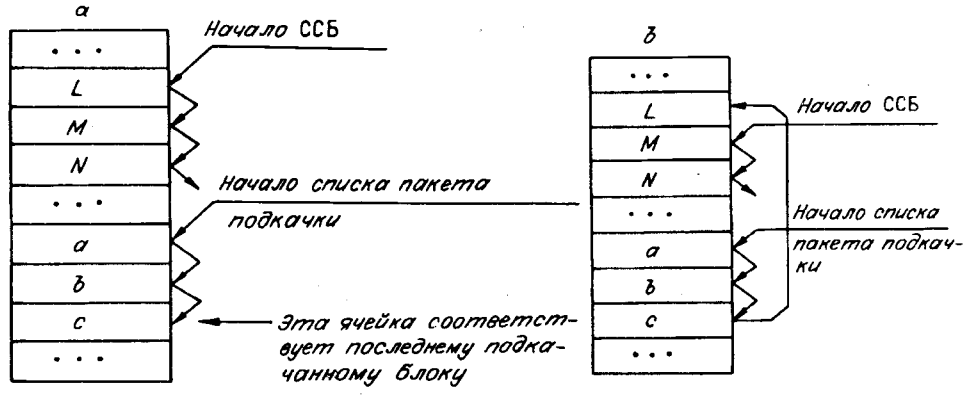

Рис. 3. Память ссылок до подкачки блока данных (*a*) и после подкачки блока данных (*b*)

В процессе подкачки определяются ссылки на начальные адреса сегментов, которые проставляются в соответствующих командах «Сегмент». Адреса этих команд передаются вместе с данными с диска, так как являются константами для каждой модели и определяются при ее подготовке. Адрес сегмента состоит из двух частей: адреса блока (его номера) и адреса в блоке (смещение). При условии, что размер блока является степенью 2, формирование адреса осуществляется без дополнительных вычислений: старшие разряды адреса — номер блока, младшие — смещение. Во время загрузки сегмента возможны две ситуации, связанные с расположением сегмента относительно границ блока:

1. Если очередное слово данных располагается в том же блоке, что и предыдущее, то адрес блока остается прежним, а смещение инкрементируется.

2. Если очередное слово данных располагается в другом блоке данных, то для формирования старших разрядов адреса осуществляется выборка из памяти ссылок, а младшие разряды устанавливаются в 0. Таким образом, производится переключение номера блока, в который происходит запись.

Модификация адреса при считывании информации из памяти данных осуществляется аналогично.

Пример расположения в памяти пакета из четырех сегментов показан на рис. 4. Сегменты *1—4* расположены в трех блоках памяти: *L*, *M* и *N*. Сегмент *1* целиком расположен в блоке *L*, сегмент *2* — также в блоке *L*, сегмент *3* — в блоке *N*. Сегмент *4* начинается в блоке *N* и переходит в блок *M*. Часть блока *M* не используется.

Количество блоков, ссылки на начало и конец пакета подкачки — дескриптор списка пакета подкачки. Аналогичный дескриптор имеет и ССБ. Наличие дескрипторов позволяет осуществлять необходимые для подкачки/чистки операции с пакетами.

Во время последовательного включения блоков в список пакета подкачки из ССБ декрементируется количество свободных блоков. Чистка осуществляется подключением списка очищаемого пакета к ССБ и сбросом бита «Подкачано» в соответствующей команде. Изменение связи блоков при чистке показано на рис. 5, при этом блоки *A*, *B*, *C* включаются в ССБ.

Потери памяти в предлагаемой организации динамического распределения. Любое динамическое распределение памяти приводит к ее потерям. В данном случае потери определяются двумя причинами.

1. Использование дополнительной памяти для организации списков — по одной ячейке на каждый блок.

2. Последний блок каждого пакета подкачки в среднем заполняется на 50 %.

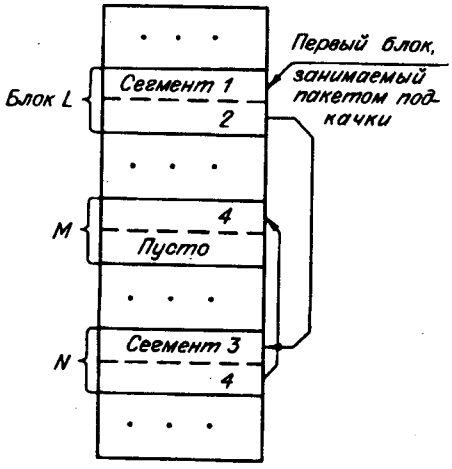

Рис. 4. Локальная память данных

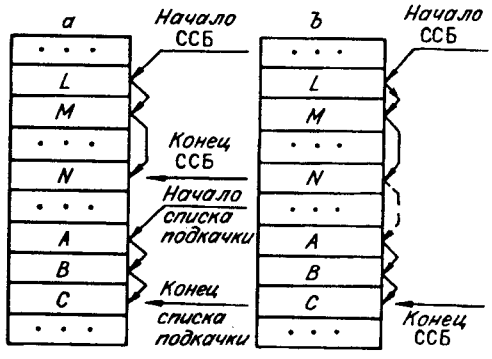

Рис. 5. Память ссылок до чистки (*а*) и после чистки (*b*)

Исходя из этих причин, существует некоторый оптимальный размер блока. При большем размере блока увеличивается неиспользованное пространст- 10 × 256 разрядов, память ссылок 10 K × 14 разрядов, дисковый накопитель CDC-9462, объем данных одного пакета накопителя 80 Мбайт, скорость передачи данных 1,2 Мбайт/с, время полной смены информации в локальной базе данных 10 с.

Использование метода динамического распределения памяти дает возможность значительно (на порядок) уменьшить объем быстродействующей оперативной памяти при незначительном аппаратном усложнении. Объем базы данных и скорость передачи информации в ССВО определяются типом НМД и могут быть увеличены использованием соответствующих накопителей.

СПИСОК ЛИТЕРАТУРЫ

1. Лубков А. А., Полубинский В. В., Храмов С. В. Работа с базой данных в ССВО высокой производительности // Автометрия.—1991.—№ 5.
2. Долговесов Б. С. Архитектура систем отображения трехмерных объектов в реальном времени широкого назначения // Машинная графика 89: Программа и тез. докл. V Всесоюз. конф.—Новосибирск: ИАиЭ СО АН СССР, 1989.
3. Гусев А. В., Ивашин С. Л., Талныкин Э. А. Математические модели сцен в синтезирующих системах визуализации реального времени // Автометрия.—1985.—№ 4.

Поступила в редакцию 10 января 1992 г.

УДК 621.396

А. А. Бычков, В. А. Понькин

(Воронеж)

ОБНАРУЖЕНИЕ ИЗОБРАЖЕНИЙ ПРОСТРАНСТВЕННО-ПРОТЯЖЕННЫХ ЗАТЕНЯЮЩИХ ФОН ОБЪЕКТОВ

Предложена новая аппликативная модель формирования изображений пространственно-протяженных затеняющих фон объектов. На основе этой модели рассмотрены задачи обнаружения детерминированных с неизвестными параметрами, а также случайных изображений пространственно-протяженных объектов, по результатам решения которых установлены границы применимости широко используемой на практике модели аддитивного взаимодействия изображений объекта и фона.

1. Обнаружение изображений пространственно-протяженных объектов (ППО) является одной из важнейших задач радио- и оптической локации, решению которой посвящены многочисленные работы [1—3 и др.]. В этих работах анализ процесса обнаружения ППО выполнен с применением аддитивной модели взаимодействия изображений объекта и фона, традиционно используемой для решения задач обнаружения сосредоточенных целей. Такая модель не учитывает эффект затенения фона, и выявленные на базе ее использования закономерности обнаружения изображений ППО, особенно в оптическом диапазоне длин волн, где пространственная протяженность объектов имеет существенное значение, не в полной мере соответствуют экспериментальным данным [4, 5].