Алгоритмы сжатия информации

Д. Мастрюков

Часть 7. Сжатие графической информации

Введение

Настольные компьютеры обычно используют для обмена информацией с пользователем свой монитор, что делает компьютерную графику важным вопросом для разработчиков программного обеспечения. Миллионы человеко-часов и миллиарды долларов вкладываются в совершенствование способов отображения данных на компьютерном экране.

Значительные капиталовложения в компьютернуга технику, способную эффективно работать с такими графическими средами как Microsoft Windows или X/Motif, создали целый класс компьютеров, способных воспроизводить на экране сложные графические образы с качеством, приближающимся к телевизионному или полиграфическому. В свою очередь появление таких компьютеров вызвало к жизни множество программ, использующих их графические возможности. Программы, работающие с графическими образами, обычно относятся к таким разным категориям, как компьютерные игры, образование, настольные издательские системы, а также многим другим. Но у них всех есть одна общая особенность: графические файлы, с которыми они работают, занимают большое количество дискового пространства.

Сжатие в стандартах графических файлов

В конце 70-х - начале 80-х годов технологии сжатия графических образов развивались в основном в направлении применения обычных алгоритмов сжатия без потерь к графическим данным. Широко распространенные в мире персональных компьютеров форматы хранения графических образов, такие как GIF, PCX, BMP, используя те или иные методы сжатия без потерь, позволяют сокращать объем файла от 10 до 90 процентов. Однако с ростом объема хранимой графической информации такой подход перестал себя оправдывать. Конечно, уменьшение объема графической информации в два раза — это вещь, которой стоит заниматься, но пользователи и разработчики систем мультимедиа так быстро «забивали» свои диски

картинками, что компьютеры, на которых такие системы могли полноценно работать, становились недостижимо дороги, не говоря уже о том, что перспектива полноценного видео на персональном компьютере становится совершенно недосягаема без специальной технологии сжатия, уменьшающей объем информации примерно на порядок.

Проблемы использования традиционных методов

«Обычные» программы и данные в компьютере, как правило, достаточно хорошо сжимаются с помощью методов, использующих статистику встречаемости либо отдельных символов, либо строк сообщения, описанных в предыдущих статьях этой серии. В отличие от них графические образы реального мира, такие, например, как отсканированная цветная фотография, сжимаются гораздо хуже. Для методов сжатия без потерь, базирующихся на использовании частот встречаемости отдельно взятых элементов сообщения (кодирование Хаффмена, арифметическое кодирование), проблемой является то, что цветовые пикселы достаточно равномерно распределены по изображению.

Спектральный анализ изображения позволяет, не внося видимых искажений, отбросить наименее значимую часть информации

Для алгоритмов, ищущих повторяющиеся подстроки (LZSS, LZW), проблема в том, что они рассматривают сообщение как цепочку символов, в то время как у картинки помимо ширины есть еще и высота, и вероятность появления вертикальных повторяющихся строк равна вероятности появления обычных горизонтальных строк. Кроме того, из-за обычных погрешностей дискретизации аналогового изображения действительно повторяющиеся фрагменты преобразуются в «чуть-чуть» различные, что существенно снижает эффективность кодирования.

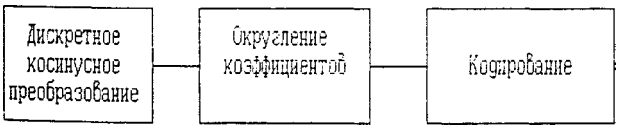

Рис. 1. Последовательность работы алгоритма JPEG

Сжатие с потерями

Очевидно, что дискретные изображения, будучи, так же как и дискретный звук, цифровой моделью аналогового явления, становятся отличными кандидатами для сжатия с потерями. Первые исследования в этой области начались в конце 70-х. В конце 80-х годов появились первые коммерческие реализации алгоритмов такого типа в виде сопроцессорных плат. Также в конце 80-х две крупнейшие в мире организации стандартов CCITT и ISO объединили свои усилия для выработки стандарта формата графического файла, сжатого с потерями.

К выработке стандарта были привлечены и ученые, и инженеры, что помогло избежать сдерживающих развитие технологии последовательностей стандартизации. Стандарт получил свое название JPEG по по имени группы разработчиков — Joint Photographic Experts Group. Спецификация JPEG состоит из нескольких частей, включая сжатие с потерями и без. Далее в этой статье речь пойдет о части JPEG, реализующей сжатие с потерями.

Шаги работы кодера JPEG

При работе компрессора JPEG последовательно выполняются следующие три операции (рисунок 1).

Эти три ступени сжатия образуют мощный компрессор, способный сжимать графические образы примерно в десять раз, теряя очень немного в качестве изображения.

Дискретное косинусное преобразование

Ключевым компонентом работы алгоритма является дискретное косинусное преобразование (далее ДКП). ДКП представляет собой разновидность преобразования Фурье и, так же как и оно, имеет обратное преобразование (ОДКП). Графическое изображение можно рассматривать как совокупность пространственных волн, причем оси X и Y совпадают с шириной и высотой картинки, а по оси Z откладывается значение цвета соответствующего пиксела изображения. ДКП позволяет переходить от пространственного представления картинки к ее спектральному представлению и обратно. Воздействуя на спект-

ральное представление картинки, состоящее из «гармоник», то есть отбрасывая наименее значимые из них, можно балансировать между качеством воспроизведения и степенью сжатия.

Формулы прямого и обратного ДКП представлены на рисунках 2 и 3. ДКП преобразует матрицу пикселов размером NxN в матрицу частотных коэффициентов соответствующего размера. Несмотря на видимую сложность, закодировать эти формулы достаточно просто.

В получившейся матрице коэффициентов низкочастотные компоненты расположены ближе к левому верхнему углу, а высокочастотные — справа и внизу. Это важно потому, что большинство графических образов на экране компьютера состоит из низкочастотной информации. Высокочастотные компоненты не так важны для передачи изображения. Таким образом ДКП позволяет определить, какую часть информации можно безболезненно выбросить, не внося серьезных искажений в картинку. Трудно представить себе, как эту задачу можно было бы решить, не преобразуя исходное изображение.

$$DCT(i,j)=\frac{1}{\sqrt{2N}}C(i)C(j)\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}pixel(x,y)\cos\left[\frac{(2x+1)i\pi}{2N}\right]\cos\left[\frac{(2x+1)i\pi}{2N}\right]$$

$$C(x)=\begin{cases}\frac{1}{\sqrt{2}}, & x=0\\ 1, & x>0\end{cases}$$

Рис. 2. Дискретное косинусное преобразование

$$DCT(i,j)=\frac{1}{\sqrt{2n}}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}C(i)\,C(j)\,DCT(i,j)\cos\left[\frac{(2x+1)\,i\pi}{2N}\right]\cos\left[\frac{(2y+1)\,j\pi}{2N}\right]$$

$$C(x)=\begin{cases}\frac{1}{\sqrt{2}}, & x=0\\ 1, & x>0\end{cases}$$

Рис. 3. Обратное дискретное косинусное преобразование

$$C(i,j)=\begin{cases}\frac{1}{\sqrt{N}}, & i=0\\ \sqrt{\frac{2}{N}}\cos\left[\frac{(2j+1)\,i\pi}{2N}\right], & i>0\end{cases}$$

Рис. 4. Матрица косинусного преобразования

Реализация ДКП

Нетрудно заметить, что время, необходимое для вычисления каждого элемента матрицы ДКП, сильно зависит от ее размера. Так как используются два вложенных цикла, время вычислений составляет O(NxN). Одним из наиболее болезненных последствий является то, что практически невозможно выполнить ДКП всего изображения сразу. В качестве решения этой проблемы группа разработчиков JPEG предложила разбивать изображение на блоки размером 8x8 точек. Увеличивая размеры блока ДКП, можно добиться улучшения результатов сжатия, но не до бесконечности, так как мало вероятно, что сильно удаленные точки «похожи» друг на друга.

По определению ДКП для его реализации потребуется два вложенных циклами тело циклов будет выполняться NxN раз для каждого элемента матрицы ДКП. Значительно более эффективный вариант вычисления коэффициентов ДКП может быть реализован через перемножение матриц.

Формула ДКП может быть записана несколько по-другому:

ДКП = КП x Точки x КПт

Здесь КП — матрица косинусного преобразования размером NxN, элементы которой определяются по формуле, приведенной на рисунке 4; точки — матрица размером NxN, состоящая из точек картинки; КПт — транспонированная матрица КП; оператор V обозначает умножение матриц. Если рассматривать алгоритм JPEG и приведенную программу, то размер всех матриц составляет 8x8 элементов. При перемножении матриц «цена» вычисления одного элемента результирующей матрицы составляет N умножений и N сложений, при вычислении матрицы ДКП — 2 x N соответственно. По сравнению с O(NxN) это заметное повышение производительности.

Пути повышения производительности

Конечно, коммерческие реализации алгоритма JPEG отличаются от приведенной здесь значительно более высокой производительностью. Одним из путей ее повышения является использование только целочисленной арифметики. В данной реализации для достижение лучших результатов в качестве изображения используются обычные операции с плавающей точкой. Очевидно, версии ДКП с округ-

ленными целочисленными значениями значительно быстрее. Кроме того, так как ДКП является разновидностью преобразования Фурье, то все методы ускорения преобразования Фурье могут быть применены и здесь.

Округление коэффициентов

Из рисунка 1 видно, что ДКП представляет собой преобразование информации без потерь и не осуществляет никакого сжатия. Напротив, ДКП подготавливает информацию для этапа сжатия с потерями или округления.

Округление представляет собой процесс уменьшения количества битов, необходимых для хранения коэффициентов матрицы ДКП за счет потери точности.

Стандарт JPEG реализует эту процедуру через матрицу округления. Для каждого элемента матрицы ДКП существует соответствующий элемент матрицы округления. Результирующая матрица получается делением каждого элемента матрицы ДКП на соответствующий элемент матрицы округления и последующим округлением результата до ближайшего целого числа. Как правило, значения элементов матрицы округления растут по направлению слева-направо и сверху-вниз.

Выбор матрицы округления

Очевидно, что от выбора матрицы округления зависит баланс между степенью сжатия изображения и его качеством после восстановления. Стандарт JPEG позволяет использовать любую матрицу округления, однако ISO разработала набор стандартных матриц для округления. Эти матрицы были получены в результате длительного тестирования членами JPEG и образуют прочную основу для дальнейших экспериментов.

Матрица округления, используемая в данной реализации (см. листинг), строится при помощи очень простого алгоритма. Для того чтобы определить шаг роста значений в матрице округления, задается одно значение в диапазоне [1,25], называемое фактором качества. Затем матрица заполняется следующим образом:

```
for ( i = 0; i < N; i++ )
for ( j = 0; j < N; j++ )
Matrix[i][j] = 1 + ( 1 + i + j ) * QualityFactor;
```

Фактор качества задает интервал между соседними уровнями матрицы округления, расположенными на ее диагоналях. Пример полученной таким образом матрицы округления приведен на рисунке 5. Необходимо отметить, что при таких значениях матрицы округления коэффициент в матрице ДКП, расположенный в ячейке с координатами (7,7), должен принимать значение не меньше 16, чтобы после округления иметь значение, отличное от 0, чтобы влиять на декодируемое изображение. Таким образом, операция округления является единственной фазой работы JPEG, где происходит потеря информации.

3	5	7	9	11	13	15	17
5	7	9	11	13	15	17	19
7	9	11	13	15	17	19	21
9	11	13	15	17	19	21	23
11	13	15	17	19	21	23	25
13	15	17	19	21	23	25	27
15	17	19	21	23	25	27	29
17	19	21	23	25	27	29	31

Рис. 5. Матрица округления с фактором качества, равным 2

Кодирование

Заключительная стадия работы кодера JPEG — это собственно кодирование. Оно выполняет три действия над округленной матрицей ДКП, для того чтобы повысить степень сжатия.

Первое действие — это замена абсолютного значения коэффициента, расположенного в ячейке (0,0) матрицы, на относительное. Так как соседние блоки изображения в значительной степени «похожи» друг на друга, то кодирование очередного (0,0) элемента как разницы с предыдущим дает меньшее значение.

Затем, как это показано на рисунке 6, коэффициенты матрицы ДКП обходятся зигзагом. После чего нулевые значения кодируются с использованием алгоритма кодирования повторов (RLE), а потом результат обрабатывается с помощью «кодирования энтропии» (в терминологии JPEG), то есть алгоритмов Хаффмена или арифметического кодирования, в зависимости от реализации.

Обход зигзагом

Одна из причин того, что алгоритм JPEG сжимает столь эффективно, состоит в том, что большое количество коэффициентов в матрице ДКП после операции округления превращается в нули. Это побудило разработчиков JPEG обрабатывать их не так, как значения остальных коэффициентов. Нули сжимаются с помощью алгоритма кодирования повторов, что в данной ситуации позволяет работать и быстро, и хорошо. Для того чтобы увеличить потенциальное количество нулей, которые могут встретиться подряд, матрица обходится не так, как обычно, например по строкам и по столбцам, а зигзагом по диагонали, как показано на рисунке 6.

«Кодирование энтропии»

В данной реализации предлагается упрощенная схема кодирования. Результатом работы являются тройки следующего вида:

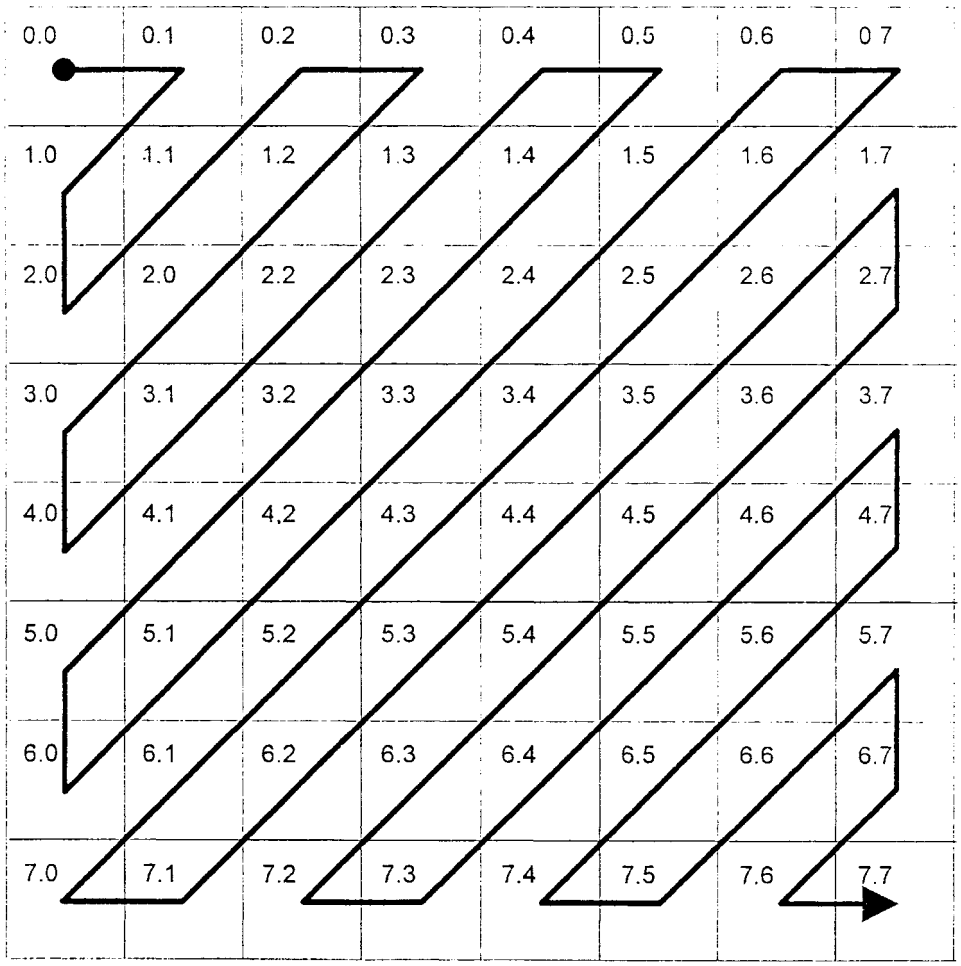

Рис. 6. Обход округленной матрицы ДКП зигзагом

<КоличествоНулей, КоличествоБитов, Коэффициент >

КоличествоНулей— количество повторяющихся нулей, предшествовавших текущему (ненулевому) элементу матрицы ДКП;
КоличествоБитов— количество битов, следующих далее, кодирующих значение коэффициента;
Коэффициент — значение ненулевого элемента матрицы ДКП.

Соответствие между полями КоличествоБитов и Коэффициент приведено в таблице 1.

Разумеется, такое кодирование не столь эффективно, как кодирование Хаффмена, но на «хороших» данных оно дает аналогичные результаты.

Таблица 1

Соответствие значений полей КоличествоБитов и Коэффициент

КоличествоБитов	Коэффициент
1	[-1,1]
2	[-3,-2],[2,3]
3	[-7,-4],[4,7]
4	[-15,-8],[8,15]
5	[-31,-16],[16,31]
6	[-63,-32],[32,64]
7	[-127,-64],[64,127]
8	[-255,-128], [128,255]
9	[-511,-256], [256,511]
10	[-1023, -512], [512,1023]

Комментарии к листингу

Приведенный пример реализации стандарта JPEG не представляет собой законченной программы, так как не поддерживает работу с файлами промышленных графических форматов, таких как BMP, PCX, GIF, TIFF, EPS и так далее. Автор умышленно стремился избежать загромождения собственно алгоритмов кодирования-декодирования сложными процедурами чтения и преобразования графических файлов.

Кроме того, данный пример реализует упрошенную схему обработки цветов изображения, подразумевая восьмибитовое значение в качестве единственного цветового компонента пиксела изображения. В случае использования более сложной цветовой модели, такой как RGB например, описанная процедура должна быть выполнена раздельно для каждого цветового компонента.

Заключение

Эта статья завершает серию, посвященную алгоритмам сжатия информации. В статьях этой серии были рассмотрены практически все существующие алгоритмы сжатия информации в вычислительных системах, за исключением, быть может, фрактального сжатия и MPEG. Большое количество вопросов типа: «Wanted: xxx compression algorythm source» — в телеконференции Internet «comp.compression» позволяет автору надеяться, что примеры приведенных программ найдут применение у читателей. •

Литература

1. **Storer J.A. Data Compression: Methods and Theory.- USA: Computer Science Press, 1988.**
2. **Nelson M. The Data Compression Book.- USA: M&T Publishing, 1991.**
3. **Wallace G.K. The JPEG Still Picture Compression Standard// Communication of the ACM.- 1991.- Vol. 34.- № 4.**
4. **Д. Мастрюков. Алгоритмы сжатия информации. Часть 5. Алгоритмы сжатия в драйверах устройств// Монитор.- 1994.- № 4.**
5. **Д. Мастрюков. Алгоритмы сжатияинформации. Часть 1. Сжатие по Хаффмеиу// Монитор.- 1993.- № 7-8.**