М. А. Иванов

ОБЗОР MPEG-ПОДОБНЫХ МЕТОДОВ КОДИРОВАНИЯ ВИДЕОДАННЫХ*

ВВЕДЕНИЕ

С возникновения первых цифровых видеосистем началась активная разработка методов и алгоритмов сжатия видеоданных. Для качественной оцифровки телевизионного сигнала шириной 6 МГц необходимо работать с частотой дискретизации не менее 12 МГц (теорема Котельникова), т. е. брать как минимум 12 млн. отсчетов в секунду. При использовании восьмибитного кодирования получим скорость передачи сигнала яркости (для черно-белой картинки) 100—120 Мбит/с. Добавим сюда цветовую информацию и служебные сигналы, получим скорость 270 Мбит/с. Обработать в реальном времени, передать и сохранить такой поток зачастую не под силу даже современным компьютерным системам.

В 1988 г. немногочисленная группа молодых экспертов (MPEG — Motion Picture Expert Group) взялась за разработку формата хранения видеоинформации на CD-ROM и ее воспроизведения со скоростью около 1,5 Мбит/с. В январе 1992 г. опубликованы спецификации MPEG-1 — первого представителя семейства MPEG. В качестве стандарта они были приняты почти через два года, к декабрю 1993 г.

Логическим продолжением MPEG-1 стал формат MPEG-2, разработанный не столько для хранения, сколько для передачи данных. Ключевой особенностью MPEG-2 является возможность разделить результирующий, подготовленный к передаче, видеосигнал на несколько независимых потоков, содержащих сигналы различного качества.

С приходом идеи телевидения высокой четкости (High Definition TV — HDTV) началась работа над форматом MPEG-3. Впоследствии оказалось, что MPEG-2 (с некоторыми уточнениями) вполне пригоден для этих целей, и работы над MPEG-3 были прекращены.

Зато начались работы над MPEG-4 — принципиально новым алгоритмом компрессии и передачи цифровых аудио- и видеосигналов, предназна-

* Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований (грант № 01-01-794) и Министерства образования РФ.

ченным для «кодирования аудио- и видеоданных с очень малыми скоростями передачи». И хотя разработка его спецификаций началась еще до массового признания Всемирной паутины, MPEG-4-подобные алгоритмы кодирования занимают лидирующее положение среди методов Internet-вещания.

В данной статье мы рассмотрим основные алгоритмы сжатия видеоданных в MPEG-подобных системах кодирования. Следует заметить, что эта статья ни в коей мере не является описанием какого-либо MPEG-подобного формата (подробные описания форматов можно взять на сайте www.mpeg.org). Здесь собраны лишь основные методики их построения. Способы кодирования аудиоданных также не будут рассмотрены.

1. ТИПЫ КАДРОВ

Все MPEG-подобные методы видеокодирования выделяют из видеопоследовательности (последовательности кадров) кадры нескольких типов. К каждому типу кадра применяется свой алгоритм кодирования. Основные типы кадров таковы:

- I-Frame — ключевые кадры (intra-frames);
- P-Frame — кадры, при кодировании/декодировании которых используется предыдущий кадр (predicted frames);
- B-Frame — кадры, при кодировании/декодировании которых используется как предыдущий, так и последующий кадр видеопоследовательности (bi-directional frames).

Рассмотрим каждый тип кадра подробнее.

I-Frame — так называемые ключевые или опорные кадры. Кодирование/декодирование кадров этого типа происходит без участия других кадров видеопоследовательности. Опорные кадры кодируются с максимально возможным качеством, т.к. по ним будут восстанавливаться последующие кадры, следовательно, они имеют больший размер по сравнению с другими кадрами, поэтому нежелательно иметь большое число опорных кадров в видеопоследовательности. Ключевые кадры, как правило, расставляются через определенные промежутки времени, поскольку декодирование произвольного кадра требует декодирования опорного кадра, т.е. если требуется декодировать какой-то кадр, то нужно декодировать ближайший предшествующий ему опорный кадр, а затем все кадры между опорным и требуемым кадром. Также ключевые кадры вставляются в моменты сильного изменения изображения по причинам, которые будут изложены ниже.

P-Frame — кадры, построенные на основе предыдущего кадра с использованием компенсации движения объектов сцены. Это наиболее часто встречающиеся кадры в кодированной видеопоследовательности. Их размер существенно меньше размера ключевых кадров. При их построении проводится сравнение декодированного предыдущего кадра и исходного текущего кадра. Производится попытка обнаружить похожие области в обоих кадрах, чтобы выяснить, как изменились местоположения объектов сцены. Такие изменения записываются в форме векторов движения, по которым будет восстановлен текущий кадр.

B-Frame — кадры, построенные на основе предыдущего и последующего кадров видеопоследовательности. По аналогии с P-Frame находятся вектора движения для перемещения объектов в моменты времени между предыдущим и текущим кадрами, а также между текущим и последующим кадрами. Далее возможны два способа построения: либо хранятся оба вектора движения, и по ним интерполяцией восстанавливается текущий кадр, либо вектора вообще не хранятся и восстановление происходит по взвешенной сумме векторов движения из предыдущего и последующего кадров. В первом случае размер кадра получается больше, чем размер P-Frame, но качество изображения заметно лучше, во втором случае размер кадра получается на порядок меньше, но качество при этом существенно хуже.

Поясним порядок расстановки, порядок кодирования/декодирования и порядок следования кадров в закодированной видеопоследовательности на рис. 1.

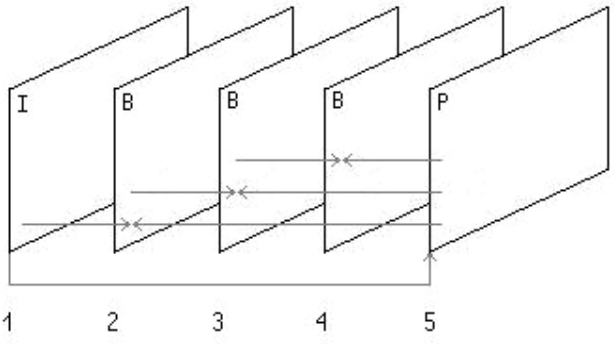

Рис. 1. Расстановка I-, P- и B- кадров, их зависимость друг от друга

Порядок кодирования/декодирования кадров в видеопоследовательности, представленной на рис. 1, следующий: сначала кодируется кадр 1 — ключевой кадр. Затем кодируется кадр 5 — P-Frame, далее, принимая кадр 1 за предыдущий, а кадр 5 — за последующий, кодируется кадр 2 — B-Frame, затем предыдущим кадром считаем кадр 2, а последующим — по-прежнему кадр 5, повторяем процедуру кодирования B-Frame.

В закодированной видеопоследовательности кадры должны следовать в порядке, допускающем их декодирование. Следовательно, первым в закодированной последовательности будет записан ключевой кадр 1, затем P-кадр 5, затем по порядку B-кадры 2, 3, 4.

При декодировании первым будет декодирован и отображен ключевой кадр 1, затем будет декодирован, но не отображен P-кадр 5, далее будут по порядку декодированы и отображены кадры 2, 3, 4 и после этого будет отображен кадр 5. Таким образом, возникнет небольшая задержка, равная времени декодирования кадра (~1/30 секунды), она может быть скомпенсирована задержкой в самом начале видеопоследовательности.

Методика выбора типа кадра может быть различной. Например, можно сравнивать так называемые среднеквадратичные разности кадров[1]:

$$MSE(F_1, F_2) = \frac{1}{w * h} \sum_{y=1}^{h} \sum_{x=1}^{w} \sqrt{(F_1(x,y) - F_2(x,y))^2} ,$$

где $F_k(x,y)$ — значение точки с координатами (x,y) в k-м кадре, h и w — соответственно высота и ширина кадра.

Можно также сравнивать максимальную по модулю разность кадров:

$$MAD(F_1, F_2) = \max_{y=1\ldots h, x=1\ldots w} \left| F_1(x,y) - F_2(x,y) \right| .$$

При использовании первого и второго способов экспериментальным путем определяются пороговые значения для каждого типа кадра и, в соответствии с полученным значением меры, выбирается типа кадра. Третий и наиболее эффективный способ определения типа кадра связан с поиском векторов движения и будет изложен ниже.

[1] Здесь и далее мы будем рассматривать только черно-белые видеопоследовательности. Цветные изображения могут быть рассмотрены как набор из трех черно-белых компонент, например в формате RGB или YCrCb. В последнем случае следует учесть, что яркостная компонента Y имеет большую значимость, чем цветовые компоненты Cr и Cb.

2. КОДИРОВАНИЕ КЛЮЧЕВЫХ КАДРОВ

Основными операциями при кодировании/декодировании ключевых кадров в MPEG-подобных видеокодерах являются операции прямого и обратного дискретного косинусного преобразования, а также операция квантования.

Дискретное косинусное преобразование (ДКП) является разновидностью преобразования Фурье. Оно позволяет перейти от пространственного представления изображения к его спектральному представлению и обратно. Результат ДКП для функции *F(x,y)*, заданной матрицей значений на дискретном множестве $\{1 \le y \le N, 1 \le x \le N\}$, есть матрица частотных коэффициентов *D* размера *NxN*.

Схема вычисления коэффициентов матрицы *D* определяется следующим выражением:

$$D(i,j) = \frac{1}{\sqrt{2N}} W(i) W(j) \sum_{y=1}^{N} \sum_{x=1}^{N} F(x,y) \cos\left(\frac{(2x+1)\pi i}{2N}\right) \cos\left(\frac{(2y+1)\pi j}{2N}\right),$$

где

$$W(i) = \begin{cases} \frac{1}{\sqrt{2}}, & i = 0; \\ 1, & i > 0. \end{cases}$$

Выражение для обратного преобразования матрицы гармоник, применяемое при распаковке изображения, записывается в виде:

$$F(x,y) = \frac{1}{\sqrt{2N}} \sum_{i=1}^{N} \sum_{j=1}^{N} W(i) W(j) D(i,j) \cos\left(\frac{(2x+1)\pi i}{2N}\right) \cos\left(\frac{(2y+1)\pi j}{2N}\right).$$

Для применения ДКП кадр разбивается на квадратные блоки (как правило, размера 8x8 точек), и затем преобразование применяется к каждому блоку в отдельности.

Время вычисления ДКП определяется по таким формулам $O(N^2)$, причем все операции должны быть проведены в числах с плавающей запятой, что существенно замедляет вычисления и, следовательно, скорость кодирования/декодирования. На практике применяется метод вычисления коэффициентов ДКП через стандартные операции умножения матриц. Схема

вычисления частотных коэффициентов выглядит таким образом: $D = C \cdot F \cdot C^T$, где

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

Для перехода от чисел с плавающей запятой к целым числам используется умножение всех коэффициентов на подходящую степень двойки. После вычисления коэффициентов ДКП происходит обратное деление (умножение и деление на степень двойки может быть очень быстро реализовано на уровне регистров процессора с помощью операций побитового сдвига влево и вправо).

После вычисления коэффициентов ДКП следует этап их квантования. Этот этап нужен для подавления высокочастотных составляющих сигнала, которые менее заметны для человеческого глаза, чем низкочастотные, а также для уменьшения избыточности информации, что позволяет добиться более высоких коэффициентов сжатия. Квантование выглядит как поэлементное деление матрицы коэффициентов ДКП D на матрицу квантования Q. Элементы матрицы Q увеличиваются в сторону увеличения индексов. Например:

$$Q = \begin{pmatrix} 3 & 5 & 7 & 9 & 11 & 13 & 15 & 17 \\ 5 & 7 & 9 & 11 & 13 & 15 & 17 & 19 \\ 7 & 9 & 11 & 13 & 15 & 17 & 19 & 21 \\ 9 & 11 & 13 & 15 & 17 & 19 & 21 & 23 \\ 11 & 13 & 15 & 17 & 19 & 21 & 23 & 25 \\ 13 & 15 & 17 & 19 & 21 & 23 & 25 & 27 \\ 15 & 17 & 19 & 21 & 23 & 25 & 27 & 29 \\ 17 & 19 & 21 & 23 & 25 & 27 & 29 & 31 \end{pmatrix}$$

В реальных кодерах используется не одна матрица квантования, а некоторое их множество (обычно 100 матриц). Это позволяет регулировать качество сжатого изображения и степень компрессии — чем больше значения коэффициентов квантования, тем хуже качество и тем выше степень компрессии.

После такой операции амплитуды высокочастотных составляющих сигнала, находящиеся ближе к правому нижнему углу матрицы коэффициентов, обратятся в нули, что существенно повысит коэффициент сжатия. Для того чтобы учесть большое количество нулей в правом нижнем углу матрицы отквантованных коэффициентов, используется специальный обход «зигзаг» в сочетании с *run-length* кодированием.

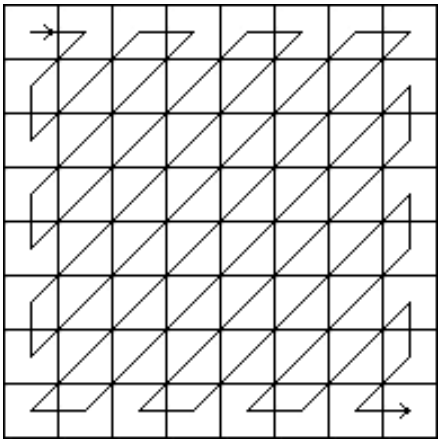

Рис. 2. Порядок обхода "зигзаг"

Суть *run-length* кодирования в том, что при обходе зигзагом будут встречаться длинные последовательности, состоящие из нулей. Нет смысла записывать каждый ноль. Выгодней записывать битовый флаг — ноль/не ноль, за которым следует либо количество нулей в последовательности, либо ненулевое значение коэффициента.

3. КОМПЕНСАЦИЯ ДВИЖЕНИЯ

Компенсация движения (motion compensation) в MPEG-подобных кодерах основана на так называемом предсказании движения (motion estimation) и кодировании остаточного изображения (residual).

Для предсказания движения кодируемый кадр разбивается на некоторые регулярные области (обычно это квадратные блоки размером 8x8, 16x16, 32x32, 64x64 точки). Далее для каждой такой области пытаются найти ее прообраз в предыдущем и/или последующем кадрах и записывают смещение этой области в виде вектора движения. Затем для получения остаточно-

го изображения найденный блок вычитается из блока в текущем кадре. Метод кодирования полученной разности будет рассмотрен далее.

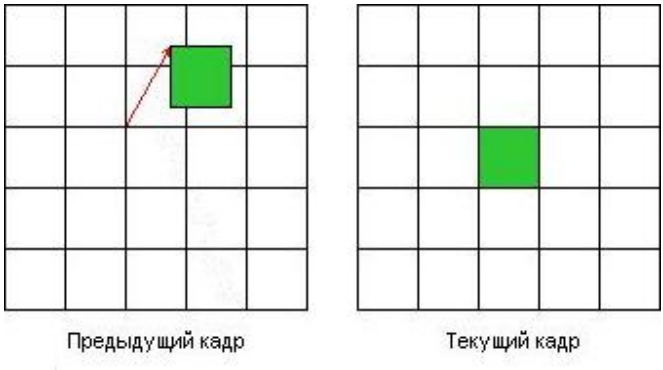

Рис. 3. Вектор движения для блока текущего кадра

На рис. 3 приведен пример нахождения вектора движения для одного блока текущего кадра в предыдущем кадре.

Исходя из соображений оптимизации поиска вектора по скорости, не имеет смысла искать вектор для текущего блока во всем предыдущем кадре. Обычно ограничиваются некоторой областью поиска, так называемый макроблок, представляющий собой квадратный блок, размеры которого в несколько раз превышают размер самого блока. Размеры макроблока могут меняться от 32x32 до 256x256. Нетрудно заметить, что чем больше размеры макроблока, тем более быстрые движения можно предсказать подобным методом, соответственно, тем медленней будет происходить поиск вектора движения.

Существуют несколько вариаций алгоритма поиска вектора движения. Они различаются скоростью работы и качеством предсказания. Рассмотрим наиболее часто употребляемые алгоритмы.

Полный перебор. Из названия метода ясно, что при его использовании перебираются все возможные варианты векторов движения. Порядок перебора следующий: фиксируем одну из координат вектора и перебираем все возможные значения второй координаты, затем меняем первую координату и т.д. Очевидно, таких векторов $(S_M - s)^2$ штук, где S_M — размер макроблока, s — размер блока. Далее из всех возможных векторов выбирается тот, который минимизирует разность блока и его прообраза (так называе-

мая ошибка поиска вектора движения) по некоторой мере. Обычно используется одна из мер, приведенных выше: MSE, MAD. Зачастую пользуются модификацией этого алгоритма, в которой перебор векторов заканчивается по достижении некоторого малого значения ошибки поиска вектора. Это существенно ускоряет время поиска.

Перебор по спирали. В этом алгоритме также перебираются все возможные вектора, но порядок перебора векторов таков, что конец вектора движения описывает расходящуюся или сходящуюся спираль вокруг блока. Перебор заканчивается по достижению порогового значения ошибки поиска. Преимущество этого метода в том, что если имеется какая-то информация о скоростях движения объектов в рассматриваемой области, то, выбрав правильное направление спирали (расходящаяся для медленных движений или сходящаяся для быстрых), можно достаточно быстро получить вектор с небольшой ошибкой поиска.

Логарифмический поиск. Зададимся радиусом $R_0 = \frac{S_M - s}{2}$. Вычислим ошибки поиска для всех блоков, центры которых находятся на расстоянии R_0 от центра макроблока (здесь под расстоянием на дискретной плоскости подразумевается мера $\rho(A,B) = R \Leftrightarrow (|X_A - X_B| = |Y_A - Y_B| = R)$. Выберем блок с наименьшей ошибкой поиска, примем центр этого блока за новый центр поиска, уменьшим радиус в два раза и произведем еще одну итерацию поиска. Итерации прекращаются либо при $R_n = 1$ пикселю, либо по достижении малого порогового значения ошибки поиска, либо когда ошибка поиска для центра очередной итерации меньше ошибок для окрестных блоков. При использовании этого алгоритма перебирается всего лишь $\log_2(S_M - s)$ векторов, что существенно ускоряет поиск. Проблема этого метода в низком качестве найденного вектора движения, ошибка поиска получается больше, чем при использовании вышеприведенных методов.

Поиск с предварительным масштабированием. Этот метод заключается в том, что предварительно блок и макроблок уменьшают в два раза по вертикали и горизонтали каким-нибудь быстрым способом, например, простым усреднением по четырем точкам. Затем в уменьшенных блоках проводится поиск либо полным перебором, либо спиралевидным перебором. Далее найденный вектор (x, y) увеличивается в два раза и проводится его уточнение путем рассмотрения 9-ти окружающих векторов: $(x \cdot 2, y \cdot 2)$,

$(x \cdot 2 \pm 1, y \cdot 2 \pm 1)$. Из них выбирается вектор, дающий минимальную ошибку. Такой способ позволяет достаточно быстро и точно найти вектор движения и поэтому он используется чаще всего.

Поскольку только очень медленный точный перебор дает оптимальный вектор движения, а также в силу того, что в реальном движении объекты состоят не только из квадратных блоков, на практике методы поиска векторов применяются в сочетании с алгоритмами уточнения векторов и улучшения предсказания движения. Рассмотрим некоторые из них.

Нахождение локального минимума ошибки. Суть метода ясна из его названия. После нахождения первого приближения вектора движения проводятся несколько итераций по нахождению локального минимума ошибки поиска в однопиксельной окрестности вектора.

Поточечная интерполяция вектора движения. Поскольку в реальных видеопоследовательностях движение объектов происходит не квадратными блоками, имеет смысл произвести поточечную интерполяцию векторов движения. Пусть найдены вектора для всех блоков текущего кадра $\{P_{n,m}\}$. Интерполируем вектор движения на все точки текущего блока с помощью поверхности второго порядка $P(x, y) = ax^2 + bxy + cy^2 + dx + ey + f$, где x, y — координаты точки внутри блока. Должны быть соблюдены граничные условия: на границах блоков вектора движения равны среднему вектору движения граничащих блоков (для внутренних точек границ это два блока, для угловых точек — четыре блока). Получается система из девяти уравнений на шесть неизвестных. Эта система решается любым удобным способом. Полученные вектора движения для каждой точки кадра учитывают движения не только одного блока, но и соседних с ним. Недостаток этого способа — большая сложность вычислений с плавающей запятой. Следует заметить, что этот способ применяется не только при декодировании. Возможна вариация алгоритма, в которой во время кодирования производится поточечная интерполяция векторов движения, вычисляется ошибка и для каждого блока сохраняется битовый признак — производить или не производить интерполяцию для данного блока.

Компенсация движения перекрывающимися блоками. Этот метод является блочным аналогом предыдущего метода. При декодировании очередного блока учитывается не только его вектор движения, но и вектора четырех соседних блоков. Отложим вектора движения четырех соседних блоков в текущем макроблоке — получим блоки L, R, T, B (левый, правый,

верхний и нижний соседние соответственно). Составим из них две матрицы LR и TB следующим образом: в матрицу LR войдут первые $\frac{n}{2}$ столбцов матрицы L и последние $\frac{n}{2}$ матрицы R, а в матрицу TB войдут первые $\frac{n}{2}$ строк матрицы T и последние $\frac{n}{2}$ строк матрицы B, расположенные в тех строках/столбцах, в которых они располагались в исходных матрицах. Далее, пусть A — текущий блок, полученный по текущему вектору движения, новое значение текущего блока вычисляется по формуле: $C(x,y) = A(x,y) \cdot H_0(x,y) + LR(x,y) \cdot H_1(x,y) + TB(x,y) \cdot H_2(x,y)$, где

$$H_0 = \begin{pmatrix} 4 & 5 & 5 & 5 & 5 & 5 & 5 & 4 \\ 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 \\ 5 & 5 & 6 & 6 & 6 & 6 & 5 & 5 \\ 5 & 5 & 6 & 6 & 6 & 6 & 5 & 5 \\ 5 & 5 & 6 & 6 & 6 & 6 & 5 & 5 \\ 5 & 5 & 6 & 6 & 6 & 6 & 5 & 5 \\ 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 \\ 4 & 5 & 5 & 5 & 5 & 5 & 5 & 4 \end{pmatrix}, \quad H_1 = \begin{pmatrix} 2 & 1 & 1 & 1 & 1 & 1 & 1 & 2 \\ 2 & 2 & 1 & 1 & 1 & 1 & 2 & 2 \\ 2 & 2 & 1 & 1 & 1 & 1 & 2 & 2 \\ 2 & 2 & 1 & 1 & 1 & 1 & 2 & 2 \\ 2 & 2 & 1 & 1 & 1 & 1 & 2 & 2 \\ 2 & 2 & 1 & 1 & 1 & 1 & 2 & 2 \\ 2 & 2 & 1 & 1 & 1 & 1 & 2 & 2 \\ 2 & 1 & 1 & 1 & 1 & 1 & 1 & 2 \end{pmatrix},$$

матрица H_2 получается из матрицы H_1 поворотом на 90°.

Таким образом, данный метод корректирует текущий блок, основываясь на данных о движении соседних блоков. В самом простом варианте метод не требует хранения никаких дополнительных данных. Аналогично вариации предыдущего метода, при кодировании можно хранить битовый признак — проводить или не проводить компенсацию перекрывающимися блоками, основываясь на сравнении ошибки с компенсацией и без компенсации. Добавим, что этот метод достаточно быстро работает и дает неплохие результаты.

4. КОДИРОВАНИЕ ОСТАТОЧНОГО ИЗОБРАЖЕНИЯ

Поскольку изображение, построенное из предыдущего кадра по векторам движения, довольно сильно отличается от кодируемого кадра, необхо-

димо провести дополнительную коррекцию изображения. Для этого вычисляется так называемое остаточное изображение. Оно равно поточечной разности оригинального кадра и его аппроксимации, построенной по векторам движения.

В MPEG-подобных видеокодерах остаточное изображение кодируется аналогично ключевому кадру. Сначала применяется ДКП, затем квантование. Следует заметить, что величины в остаточном изображении имеют меньшие значения, чем в ключевом кадре, следовательно, меньше будут и коэффициенты ДКП. Матрица ДКП для остаточного изображения будет состоять преимущественно из высокочастотных коэффициентов. Поэтому квантование остаточного изображения должно быть более мягким, чтобы обеспечить приемлемое качество изображения.

В кодировании остаточного изображения имеется еще один нюанс. Если вектор движения для какого-то блока был найден плохо, т.е. ошибка поиска велика, то может оказаться выгодней кодировать не вектор движения и остаточное изображение для блока, а собственно исходный блок так, как это делалось в ключевом кадре.

5. ОКОНЧАТЕЛЬНОЕ КОДИРОВАНИЕ ДАННЫХ

Для кодирования получившихся данных — вектора движения, отквантованные коэффициенты и т.д. — обычно применяются энтропийные методы кодирования. Они основаны на составлении таблицы частот вхождения символов в кодируемую последовательность с последующим присвоением наиболее часто встречающимся символам наиболее коротких кодов.

Типичным представителем таких способов кодирования является код Хаффмана. Обычно пользуются его модификацией — динамическим кодом Хаффмана, который не требует хранения таблицы частот, поскольку эта таблица может быть построена во время декодирования.

Еще одним представителем класса энтропийных кодировщиков является арифметический кодер. Скорость его работы немного ниже, чем у кода Хаффмана, но он позволяет достичь более высоких коэффициентов сжатия. Трудность его использования заключается в том, что патенты на большую часть модификаций арифметического кодера принадлежат ряду крупных фирм, тогда как код Хаффмана является свободным для использования.

Код Хаффмана и арифметический кодер являются универсальными кодировщиками, пригодными для данных любых типов. Если же имеется некоторая информация о кодируемых данных, то выгоднее применять специ-

альные методы кодирования или преобразования, которые уменьшат объем данных перед универсальными кодерами. Рассмотрим некоторые подходы такого рода.

Аппроксимация по соседним элементам. Когда кодируется последовательность данных, элементы которой соответствуют блокам кадра (коэффициенты ДКП, компоненты векторов движения и т.д.), то имеет смысл применять такой метод аппроксимации. Элементы, соответствующие первой строке и первому столбцу блоков, остаются неизменными. Все остальные элементы записываются как разность элемента и полусуммы его верхнего и левого соседей. В силу предположения о том, что близко лежащие блоки ведут себя примерно одинаково, такое преобразование позволит уменьшить амплитуду элементов и, следовательно, увеличить коэффициент сжатия.

Квадро-дерево. Этот способ применяется для кодирования двумерных массивов битовых признаков. Суть его заключается в следующей рекурсивной процедуре. Если кодируемый массив состоит из одинаковых символов, то сохраняем специальный код, код этого символа, и прекращаем итерации для этого массива. Если в массиве присутствуют и нули, и единицы, то разбиваем массив на четыре четверти и к каждой из них снова применяем ту же процедуру. Рекурсия прекращается в случае, когда одно из измерений массива становится равным 1.

6. ПРЕ- И ПОСТФИЛЬТРАЦИЯ ВИДЕОПОСЛЕДОВАТЕЛЬНОСТЕЙ

В заключение обзора MPEG-подобных методов видеокодирования скажем несколько слов о предварительной и конечной фильтрации видеопоследовательностей.

Префильтрация. Основной задачей префильтрации является подготовка изображения к сжатию. Она обычно состоит в обработке данных набором высокочастотных фильтров, удаляющих шумы, повышая тем самым коэффициент сжатия. Поскольку кодирование является довольно трудоемкой вычислительной задачей, префильтр должен работать очень быстро. Поэтому традиционные схемы фильтрации не всегда подходят для практического использования. Зачастую применяют простейшие сглаживающие схемы типа «крест», которые могут быть легко оптимизированы под конкретный процессор.

Постфильтрация. В силу того, что на всех этапах кодирования видеопоследовательности каждый кадр был разбит на блоки, которые обрабатывались отдельно, на декодированном изображении зачастую видны границы между блоками. Это так называемый эффект блокинга. Для уменьшения этого эффекта применяются различные схемы деблокинга, суть которых в том, что они сглаживают значения на границах блоков в пределах 1-2–ух точек с каждой стороны.

Еще один вид артефактов, появляющихся при декодировании, это так называемый москито-эффект или эффект Гиббса. Он возникает из-за того, что на этапе квантования коэффициентов ДКП происходит отбрасывание высокочастотных гармоник. Поэтому при обратном преобразовании возникают шумы в высокочастотной области. Визуально они проявляются в виде расплывчатых разводов вокруг четких границ объектов, например, на границах букв. Эту проблему частично решают демоскито-фильтры, которые являются высокочастотными фильтрами, настроенными на определенную частоту шума, которая зависит от коэффициента квантования блока.

СПИСОК ЛИТЕРАТУРЫ

1. **Рабинер П., Гоулд Б.** Теория и применение цифровой обработки сигналов: Пер. с англ. — М.: Мир, 1978.
2. **Прэтт У.** Цифровая обработка изображений. Кн.1 и 2: Пер. с англ. — М.: Мир, 1982.. — 312 и 480 с.
3. **Desmet S., Deknuydt B., Li N., Van Eycken L., and Oosterlink A.** A simple algorithm to extract realistic motion fields out of video sequence // Proc. of the EOS-SPIE Intern. Sympos. on Fiber Optic Networks and Video Communications, Berlin, Germany. — 1993. — Vol. 1977. — P. 248–254.
4. **Wuyts T., Van Eycken L., and Oosterlink A.** Calculating motion vectors for an interpolated motion field: Proc of the Conf on Advanced Image and Video Communications, SPIE, Amsterdam. — 1995. — Vol. 2451. —P. 148–155
5. **ITU-T** Recommendation H.263: Video coding for low bitrates communication. — 1996.