Особенности сжатия видеоданных
по рекомендации H.264

Игорь ГУК
gii@scanti.ru

В настоящее время очень быстро растет потребность в различных сервисах, связанных с обработкой видеоданных. Повсеместно внедряется телевидение высокого качества, увеличивается число домашних пользователей DVD-проигрывателей, широко распространяются видеоконференцсвязь, системы дистанционного обучения и безопасности и т. д. Все это требует высокоэффективных и, главное, стандартизованных алгоритмов сжатия видеопотока. Одно из таких решений — рекомендация H.264, предложенная объединенной группой экспертов, в которую вошли представители от разработчиков рекомендаций MPEG и H.263. Этими специалистами была совместно разработана рекомендация, чье полное наименование звучит так: H.264/MPEG 4 Part 10 «Advanced Video Coding».

Введение

В рекомендации нет явного определения кодера и декодера. В ней описывается синтаксис сжатого видеопотока, а также метод его декодирования. Однако можно считать, что кодер и декодер включают функциональные элементы, показанные на рис. 1 и 2. При реализации видеокодека в соответствии с рекомендацией H.264 необязательно точно повторять структуру, представленную на рис. 1 и 2, но основные элементы должны присутствовать обязательно.

Кодек H.264 содержит практически те же элементы, что и кодеки MPEG1, MPEG2, MPEG4, H.261, H.263. В целом сжатие происходит по тем же принципам, хотя во всех основных элементах есть качественные изменения. Это позволило значительно повысить качество кодирования. По оценкам экспертов, например, более чем в два раза возрастает степень сжатия при том же качестве восстановления исходного изображения.

Кодер (рис. 1) включает две ветви обработки видеопотока:

- канал прямого кодирования (отмечен синим цветом), где обработка данных происходит слева направо;
- канал реконструкции видеоизображения (отмечен сиреневым цветом), в нем обработка данных происходит справа налево.

На рис. 2 показан декодер. Обработка данных происходит справа налево. Сопоставив рис. 1 и 2, можно проследить соответствие между блоками кодера и декодера.

На вход кодера поступает кадр F_n. Обработка кадра происходит по макроблокам, соответствующим фрагментам размером 16×16 точек в исходном изображении. Каждый макроблок может быть обработан в двух режимах: INTRA или INTER. В любом режиме прогноз макроблока P_n формируется на основе восстановленного кадра.

В режиме INTRA прогноз формируется из выборок текущего кадра n, предварительно закодированных и восстановленными (F'_n на рис. 1 и 2). Причем используются выборки кадра до деблокирующего фильтра. В режиме INTER прогноз формируется с учетом изменений, которые произошли в текущем кадре по сравнению с одним или несколькими предыдущими (или последующими). Кадры, служащие для прогноза, должны быть предварительно закодированы и восстановлены. В блоках формирования INTRA- и INTER-прогноза происходит выбор наиболее подходящего способа в зависимости

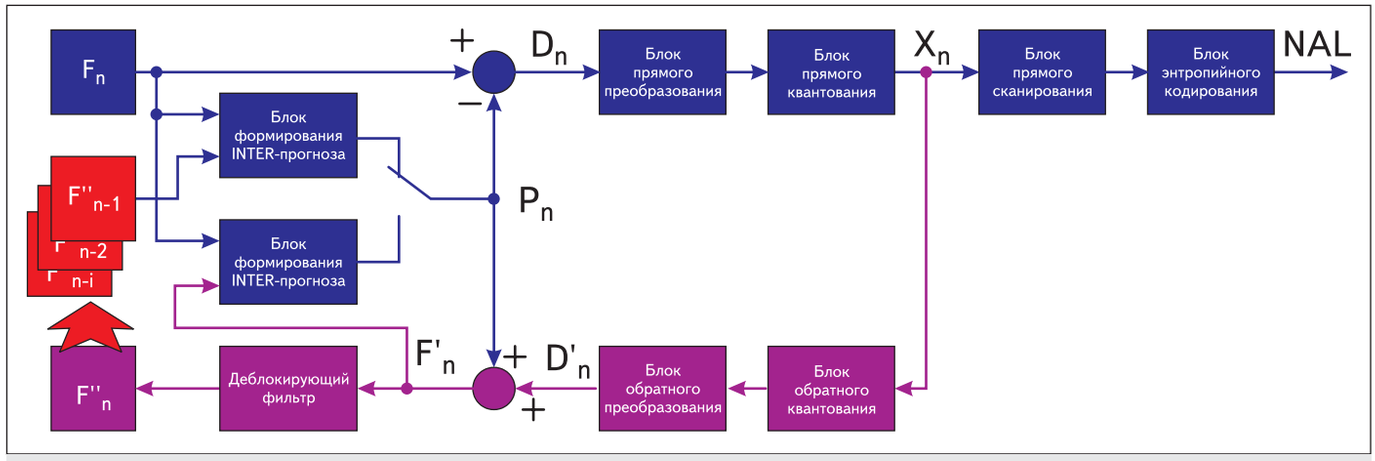

Рис. 1. Кодер

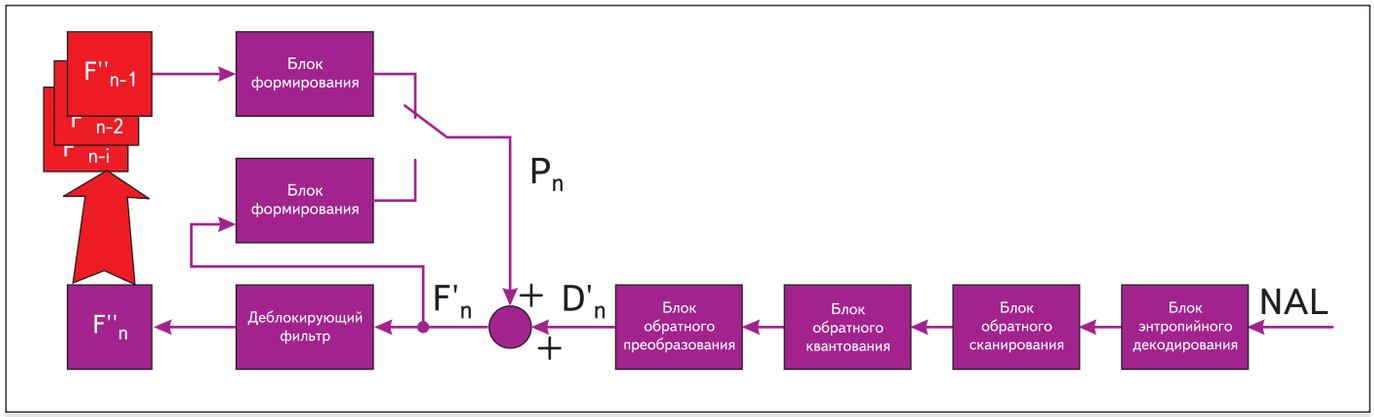

Рис. 2. Декодер

от типа кодируемого кадра. Эти способы подробно описаны в рекомендации, но критериев выбора нет. Задача выбора решается разработчиком видеокодека самостоятельно.

Полученный прогноз P_n вычитается из текущего макроблока. В результате вычисляется макроблок остаточных коэффициентов D_n. Этот макроблок поступает в преобразователь, где происходит частотное преобразование остаточных коэффициентов. Необходимо отметить, что в рекомендации используется квазиортогональное частотное преобразование. Таким образом, значительно уменьшается вычислительная сложность алгоритма, но в процессе сжатия появляются дополнительные искажения. Частотные коэффициенты квантуются (масштабируются), что позволяет произвести сжатие видеоданных с потерями. Полученный набор преобразованных и квантованных коэффициентов X служит исходным для обратного канала реконструкции данных. В дальнейшем коэффициенты переупорядочиваются в блоке прямого сканирования. В результате коэффициенты выстраиваются в линейный массив в порядке возрастания. Затем происходит энтропийное кодирование упорядоченного массива коэффициентов, что обеспечивает сжатие данных без потерь. Закодированные коэффициенты вместе с дополнительной информацией, требуемой для правильного декодирования макроблока (режима прогноза, коэффициент квантования и т. д.), составляют сжатый битовый поток данных (bitstream) абстрактного сетевого уровня (NAL). Этот поток может передаваться по каналу связи либо быть записан на любой носитель для хранения.

В канале реконструкции вначале происходит обратное квантование, затем обратное частотное преобразование. В итоге получаются восстановленные разностные коэффициенты D'_n. Они суммируются с прогнозом P_n, и это позволяет получить восстановленный кадр F'_n. Необходимо отметить, что восстановленный кадр не является идентичными исходному. В него внесены искажения, обусловленные квантованием и квазиортогональным частотным преобразованием. Именно такой кадр будет получен на приемной стороне, и поэтому именно его необходимо использовать для формирования прогноза в режиме INTRA.

При обработке кадра по макроблокам возникают специфические искажения (блочность), проявляющиеся в резких перепадах значений коэффициентов на границе между макроблоками. Для их уменьшения предназначен деблокирующий фильтр. Полученный после него восстановленный кадр служит в качестве опорного для формирования прогноза в режиме INTER. Отметим, что невозможно использовать деблокирующий фильтр в режиме INTRA, так как на момент формирования прогноза в этом режиме обработанной является только часть макроблоков, и полной информации о границе между макроблоками нет.

Декодер получает сжатый bitstream абстрактного сетевого уровня NAL (рис. 2). Структура декодера понятна из рис. 2. Он осуществляет обратные процедуры по отношению к кодеру. Стоит только отметить, что в блоках формирования INTRA- и INTER-прогнозов нет элементов, отвечающих за анализ кадра. Информация о конкретном режиме и способе его реализации извлекается из bitstream.

Формирование INTRA-прогноза

В режиме INTRA прогноз P формируется на основании предварительно закодированных и восстановленных блоков. Для яркостной составляющей изображения прогноз P может быть сформирован как для макроблока размером 16×16, так и отдельно для всех входящих в него блоков размером 4×4. Всего определено 9 дополнительных режимов (способов) формирования прогноза (предсказания) для яркостных блоков размером 4×4, 4 дополнительных режима для яркостных макроблоков размером 16×16 и 4 режима для цветоразностных блоков размером 8×8.

Обработка кадра в кодере происходит справа налево и сверху вниз. Таким образом, формирование прогноза для текущего блока (или макроблока) происходит на основе уже обработанных блоков, расположенных сверху и слева от текущего (обрабатываемого в данный момент). Отметим, что существует только один блок, который не использует прогноза. Это блок, расположенный в левом верхнем углу кадра. Кроме того, прогноз для блоков первого ряда обрабатываемого кадра основан только на блоках, расположенных слева, а прогноз для первого столбца использует только блоки сверху.

Режимы формирования прогноза для яркостных блоков размером 4×4

Отсчеты блока прогноза (обозначенные на рис. 3 как a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p) формируются на основе уже обработанных вертикальных (I, J, K, L) и горизонтальных (A, B, C, D, E, F, G, H) отсчетов предыдущих блоков (рис. 3).

При этом в рекомендации описаны следующие режимы формирования прогноза:

- **Режим 0.** Верхние отсчеты A, B, C, D экстраполируются вертикально.
- **Режим 1.** Левые отсчеты I, J, K, L экстраполируются горизонтально.
- **Режим 2.** Все отсчеты в прогнозе P являются средним из выборок A...D и I...L.
- **Режим 3.** Отсчеты интерполируются под углом в 45° в направлении между нижним левым и верхним правым отсчетами.
- **Режим 4.** Отсчеты интерполируются под углом 45° вниз и направо.
- **Режим 5.** Интерполяция отсчетов производится под углом приблизительно 26,6° влево к вертикальному направлению (отношение ширины к высоте равно 1/2).

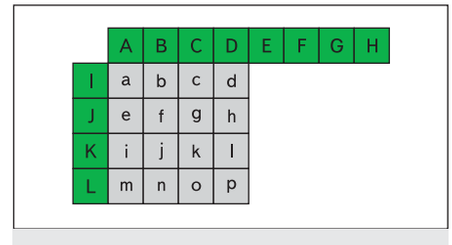

Рис. 3. Маркировка отсчетов для блока прогноза размером 4×4

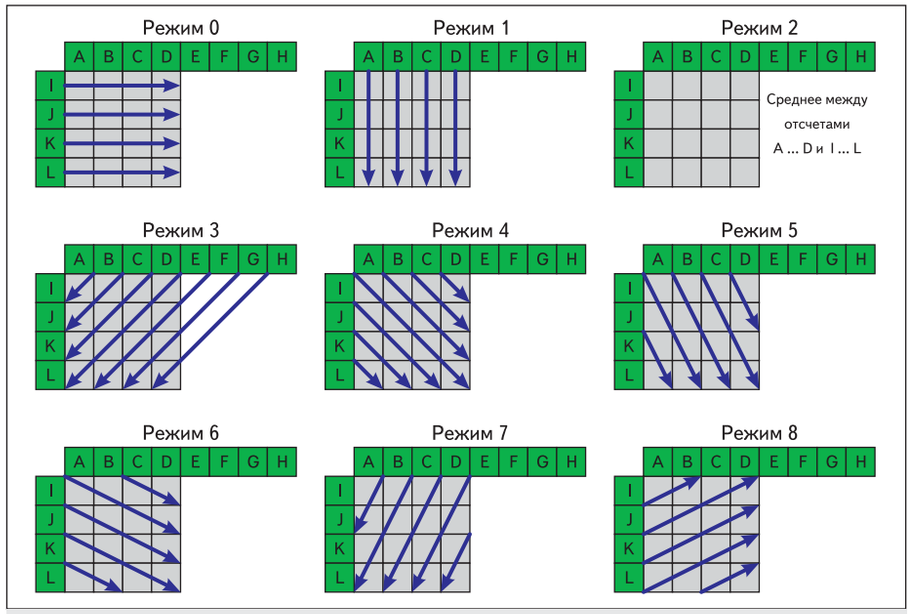

Рис. 4. Режимы формирования прогноза для яркостных блоков размеры 4?4

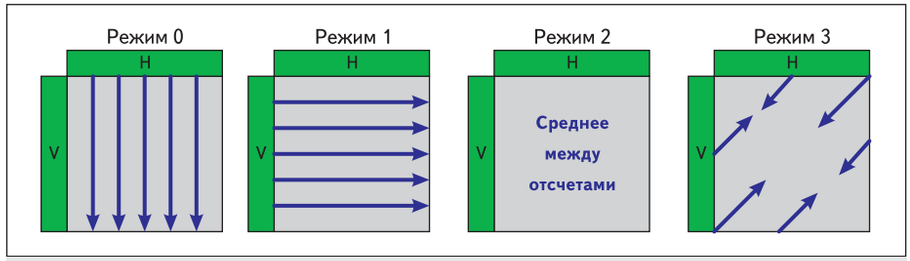

Рис. 5. Режим формирования прогноза INTRA-блоков размером 16×16

- **Режим 6.** Интерполяция отсчетов производится под углом приблизительно 26,6° вниз по отношению к горизонтальному направлению.
- **Режим 7.** Интерполяция отсчетов производится под углом приблизительно 26,6° вправо от вертикального направления.
- **Режим 8.** Интерполяция отсчетов производится под углом приблизительно 26,6° выше горизонтального направления.

Стрелки на рис. 4 указывают направление предсказания в каждом режиме. В режимах 3–8 отсчеты прогноза формируются из средневзвешенных выборок A…L. Например, если выбран режим 4, то отсчет прогноза *d* (рис. 3) высчитывается по формуле:

$$d = \text{round}\left(\frac{B}{4} + \frac{C}{2} + \frac{D}{4}\right).$$

Кодер может выбирать режим предсказания для каждого блока таким образом, чтобы минимизировать разность между прогнозом P и кодируемым блоком.

Режимы формирования прогноза для яркостных блоков размером 16×16

Кодер имеет возможность кодировать не только яркостные блоки размером 4×4, но и размером 16×16. В рекомендации H.264 определены четыре режима формирования прогноза для макроблоков (рис. 5).

- **Режим 0.** Экстраполяция верхних выборок H.
- **Режим 1.** Экстраполяция левых выборок V.
- **Режим 2.** Среднее между верхними H и левыми выборками V.
- **Режим 3.** Линейное сглаживание между верхними H и левыми выборками V.

Режимы формирования прогноза для цветоразностных блоков размером 8×8

Каждый из отсчетов цветоразностного блока размером 8×8 предсказывается от цветоразностных выборок, расположенных выше и/или слева, закодированных и восстановленных. Четыре режима формирования прогноза для цветоразностных составляющих аналогичны режимам для яркостного блока размером 16×16, рассмотренным в разделе «Введение».

Необходимо отметить, если любой из яркостных блоков макроблока кодируется в INTRA-режиме, то и оба цветоразностных блока также должны быть закодированы в INTRA-режиме.

Формирование INTER-прогноза

В INTER-режиме прогноз формируется на основе одного или нескольких предварительно закодированных и восстановленных кадров путем выбора в них опорных блоков, сдвинутых относительно кодируемого блока (прогноз с компенсацией движения). Кодек стандарта H.264 использует тот же принцип компенсации движения, что и в старых стандартах, начиная с H.261. Важными отличиями от более ранних версий являются возможность менять размер блока (от 16×16 до 4×4) и возможность поиска векторов движения с меньшим шагом (1/4 пикселя для яркостных компонент).

Разбиение макроблока на блоки для формирования прогноза с компенсацией движения

В кодере, описываемом в рекомендации H.264, в режиме прогнозирования с компенсацией движения поддерживаются размеры блоков от 16×16 до 4×4 для яркостной составляющей изображения в различных сочетаниях. Яркостная составляющая каждого макроблока (размером 16×16) может быть разбита четырьмя способами, как это показано на рис. 6: 16×16, 16×8, 8×16 или 8×8. Каждая из областей меньшего размера является частью макроблока. Если выбран режим поиска векторов движения для блоков 8×8, то каждый из четырех блоков 8×8 может быть разбит на блоки четырьмя способами, как показано на рис. 7: 8×8, 8×4, 4×8 или 4×4. Возможность делить макроблок на блоки, а те в свою очередь на субблоки, позволяет получить большое количество комбинаций их сочетаний в пределах каждого макроблока. Этот метод разбиения макроблоков на блоки

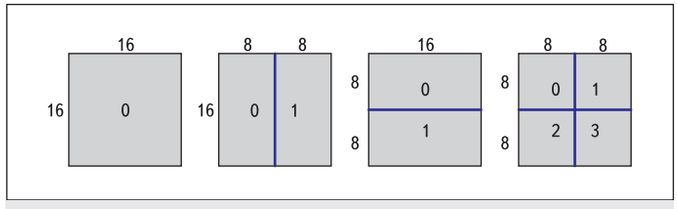

Рис. 6. Разбиение макроблока 16×16 на блоки 16×16, 16×8, 8×16, 8×8

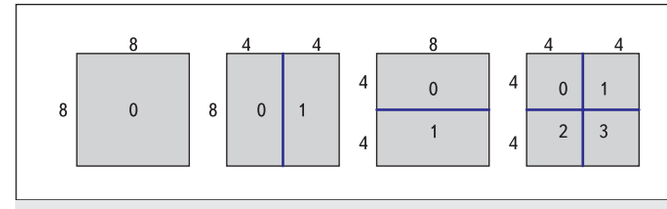

Рис. 7. Разбиение блока 8×8 на субблоки 8×8, 8×4, 4×8, 4×4

и субблоки в режиме компенсации движения получил название структурного дерева компенсации движения.

Для каждого блока и субблока требуется отдельный вектор движения. Каждый вектор движения должен быть закодирован и передан; кроме того, необходимо закодировать и передать конкретный вид структурного дерева для каждого макроблока.

Выбор большого размера блоков (например, 16×16, 16×8, 8×16) означает, что количество бит для передачи данных векторов движения и структуры дерева минимально, однако при этом остаточные коэффициенты могут содержать существенное количество энергии в высокочастотных областях кадра (с высокой детальностью). При выборе маленького размера блоков (например, 8×4, 4×4 и т. д.) можем получить более низкую энергию остаточных коэффициентов после компенсации движения, но потребуется большее число бит для кодирования и передачи данных о векторах движения и структуре дерева. Поэтому выбор размера блоков оказывает существенное влияние на эффективность сжатия. Вообще, большой размер блоков соответствует низкочастотным областям кадра, а маленький — может быть выгоден для высокочастотных.

Разрешающая способность цветоразностных составляющих макроблока (Cr и Cb) равна половине яркостного. Каждый цветоразностный блок разбивается таким же образом, как яркостной, за исключением того, что вертикальные и горизонтальные размеры блока в два раза меньше (если яркостной блок имеет размер 8×16, то соответствующий ему цветоразностный блок имеет размер 4×8; яркостному блоку 8×4 соответствует цветоразностный 4×2; и т. д.). Горизонтальные и вертикальные составляющие каждого вектора движения (одного на блок) делятся на два, когда применяются к цветоразностным блокам.

На рис. 8 показан пример разбиения остаточного кадра (до процедуры компенсации движения). Кодер выбирает «лучший» размер блока для каждой части кадра, то есть такой размер, который минимизирует кодирование остаточных коэффициентов и векторов движения. На рис. 8 тонкими линиями показано разбиение кадра на блоки. В областях кадра, где изменения незначительны (остаточные коэффициенты кажутся серыми), выбирается размер блока 16×16; в областях с большими изменениями (остаточные коэффициенты кажутся черными или белыми) выбираются меньшие размеры блоков.

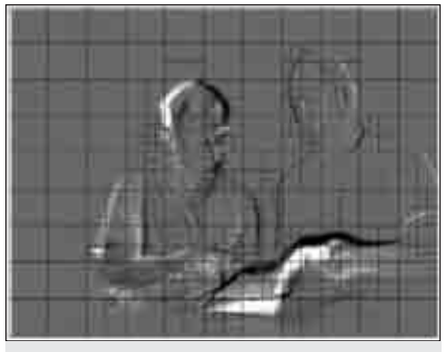

Рис. 8. Оптимальное разбиение кадра на блоки и субблоки

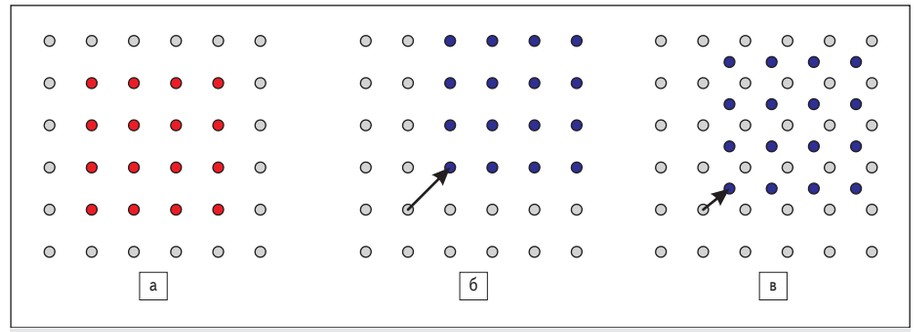

Рис. 9. Пример целочисленного и дробного векторов движения

Дробные значения векторов движения

Каждый блок в режиме INTER-кодирования макроблока предсказывается от блока того же размера в опорном кадре. Смещение между этими двумя областями (вектор движения) имеет минимальное разрешение, равное четверти расстояния между точками (пикселями) в опорном изображении (для яркостной составляющей). Опорных яркостного и цветоразностного блоков для нецелочисленных (дробных) векторов движения в опорном кадре не существует, поэтому их необходимо вычислить на основе ближайших пикселей.

На рис. 9 показан пример целочисленного и дробного векторов движения. Необходимо сформировать прогноз для блока размером 4×4 (красные точки на рис. 9а). Если горизонтальные и вертикальные компоненты вектора движения — целые числа (рис. 9б), то опорный блок в опорном кадре существует (синие точки). Если один или оба вектора движения — дробные числа (рис. 9в), прогноз (синие точки) формируется вставкой между смежными пикселями в опорном кадре (серые точки).

Компенсация движения с четвертьпиксельным разрешением может обеспечивать значительно лучшую эффективность сжатия, чем компенсация с целочисленным разрешением, но при этом увеличивается сложность процедуры поиска векторов движения.

Интерполированные выборки формируются следующим образом. В яркостной компоненте опорного изображения сначала формируются выборки с полупиксельной точностью (рис. 10). На этом рисунке выборки в целочисленных позициях выделены серым цветом. Каждая выборка в полупиксельной позиции, которая является смежной с двумя целочисленными выборками (например, выборки b, h, m, s на рис. 10), интерполируется на основе 6 пикселей в целочисленных позициях при помощи КИХ-фильтра. Коэффициенты фильтра равны: 1/32, –5/32, 5/8, 5/8, –5/32, 1/32.

Например, полупиксельная выборка b рассчитывается от 6 горизонтальных целочисленных выборок E, F, G, H, I и J по формуле:

$$b = \frac{\text{round}(E - 5F + 20G + 20H - 5I + J)}{32}.$$

Точно так же выборка *h* интерполируется на основе выборок A, C, G, M, R и T. После формирования полупиксельных выборок, смежных с целочисленными отсчетами, производится вычисление остальных полупиксельных выборок (например, выборки в позиции *j* на рис. 10) с интерполированием 6 горизонтальных или вертикальных полупиксельных выборок, полученных на первом шаге. Например, выборка *j* формируется на основе выборок *cc*, *dd*, *h*, *m*, *ee* и *ff*. (Обратите внимание, что получается одинаковый результат при интерполировании по горизонтали или вертикали.) Интерполирующий КИХ-фильтр 6-го порядка является относительно сложным (по сравнению, например, с билинейной интерполяцией), но формирует более точный прогноз и, следовательно, обеспечивает лучшую эффективность компенсации движения.

Как только получены все выборки в полупиксельных позициях, на их основе формируются выборки с четвертьпиксельной точностью методом линейной интерполяции (рис. 11). Выборки в четвертьпиксельной позиции формируются на основе двух горизон-

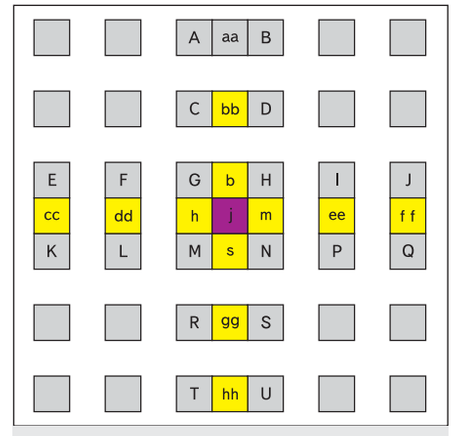

Рис. 10. Вставка в яркостный блок полупиксельных выборок

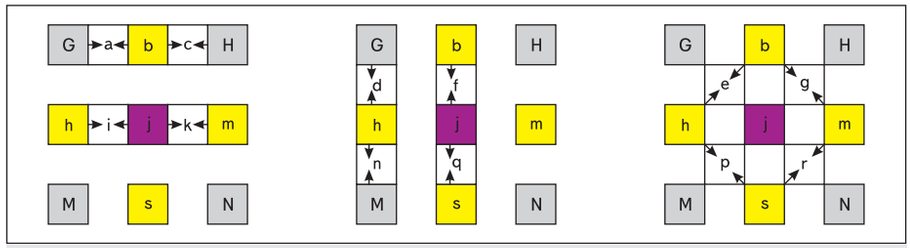

Рис. 11. Вставка в яркостный блок четвертьпиксельных выборок

тальных или вертикальных смежных полуили целочисленных выборок (например, *a*, *c*, *i*, *k* и *d*, *f*, *n*, *q* на рис. 11) путем линейной интерполяции между ними. Например:

$$a = \frac{\text{round}(G + b)}{2}.$$

Оставшиеся четвертьпиксельные выборки (*e*, *g*, *p* и *r* на рис. 11) линейно интерполируются между диагональной парой противоположных полупиксельных выборок. Например, *e* интерполируется на основе выборок *b* и *h*.

Векторы движения с четвертьпиксельным разрешением для яркостной компоненты требуют 1/8-пиксельной точности для цветоразностной компоненты (для формата кадра YUV 4:2:0). Интерполирование выборок в 1/8-пиксельных интервалах производится на основе целочисленных выборок в каждом цветоразностном компоненте. В этом случае используется линейная интерполяция для формирования цветоразностных выборок с 1/8-пиксельной точностью (рис. 12).

В этом случае каждая выборка формируется как линейная комбинация соседних пикселей в целочисленных позициях, например, для выборки *a* аппроксимация производится на основе целочисленных выборок A, B, C и D (1).

Для примера можно рассмотреть случай, показанный на рис. 12 ($d_x = 2$ и $d_y = 3$):

$$a = \frac{\text{round}(30A + 10B + 18C + 6D)}{64}.$$

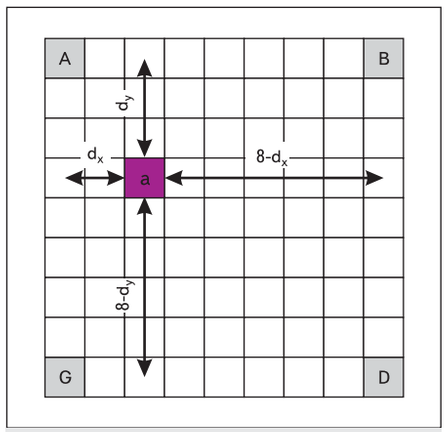

Рис. 12. Вставка в цветоразностный блок 1/8-пиксельных выборок

Преобразование, сканирование и квантование

Каждый разностный макроблок преобразуется, сканируются и квантуется. Предыдущие стандарты, такие как MPEG-1, MPEG-2, MPEG-4 и H.263, использовали в качестве базового преобразования дискретное косинусное преобразование (DCT) для массивов размером 8×8. В рекомендации H.264 определены три вида преобразования в зависимости от типа обрабатываемых блоков:

- преобразование блока размером 4×4 нулевых (DC) яркостных коэффициентов, полученных из яркостных макроблоков размером 16×16, сформированных в режиме INTRA;
- преобразование блока размером 2×2 нулевых (DC) цветоразностных коэффициентов, полученных из цветоразностных блоков любого типа;
- преобразование для всех остальных разностных блоков размером 4×4.

Сканирование макроблоков

Данные в пределах макроблока передаются в порядке, показанном на рис. 13. Если макроблок размером 16×16 обрабатывается в режиме INTRA, то вначале передается блок DC-коэффициентов размером 4×4. Он маркируется значением «–1». Затем передаются остальные (0–15) яркостные блоки в порядке, показанном на рис. 13 (с обнуленными DC-коэффициентами). Блоки 16 и 17 содержат массивы размером 2×2 DC-коэффициентов, полученных из цветоразностных Cb- и Cr-составляющих соответственно. Наконец, передаются оставшиеся цветоразностные блоки 18 и 25 (с обнуленными DC-коэффициентами).

Преобразование и квантование для блоков 0–15 и 18–25

Это преобразование используется для блоков размером 4×4 с номерами 0–15 и 18–25 на рис. 13 после осуществления INTER- или INTRA-обработки. Преобразование основано на DCT, но имеет некоторые фундаментальные отличия:

1. Это целочисленное преобразование (все операции могут быть выполнены в целочисленной арифметике без потери точности).
2. Обратное преобразование, определенное в рекомендации H.264, производит восстановление исходных данных с искажениями, то есть является приблизительным.
3. Основная часть преобразования не имеет умножений, то есть оно основано только на суммированиях и сдвигах.
4. Масштабирующее умножение (входящее в процесс преобразования) интегрировано в процесс квантования (что сокращает общее количество умножений).

Полный процесс преобразования и квантования может быть выполнен при помощи 16-разрядной целочисленной арифметики, и только одно умножение выполняется с небольшой потерей точности.

Приблизительное DCT 4×4

DCT для входного массива **X** размером 4×4 определяется соотношением (2), где

$$a = \frac{1}{2},\ b = \sqrt{\frac{1}{2}}\cos\left(\frac{\pi}{8}\right),\ c = \sqrt{\frac{1}{2}}\cos\left(\frac{3\pi}{8}\right).$$

Данное матричное умножение может быть факторизировано и формула (2) примет следующий вид (3).

$$a = \frac{\text{round}((8-d_x)(8-d_y)A + d_x(8-d_y)B + (8-d_x)d_yC + d_xd_yD)}{64} \quad (1)$$

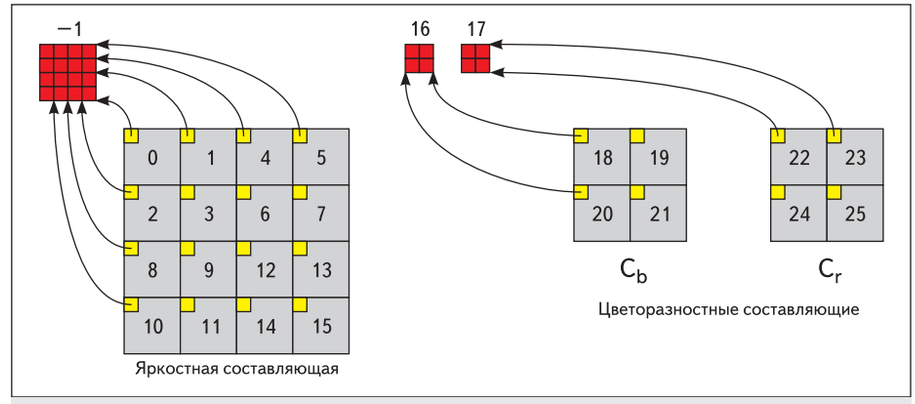

Рис. 13. Порядок сканирования блоков внутри макроблока

$$Y = AXA^T = \begin{bmatrix} a & a & a & a \\ b & c & -c & -b \\ a & -a & -a & a \\ c & -b & b & -c \end{bmatrix} \times [X] \times \begin{bmatrix} a & b & a & c \\ a & c & -a & -b \\ a & -c & -a & b \\ a & -b & a & -c \end{bmatrix} \quad (2)$$

$$Y = (CXC^T) \otimes E = \left(\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & d & -d & -1 \\ 1 & -1 & -1 & 1 \\ d & -1 & 1 & -d \end{bmatrix} \times [X] \times \begin{bmatrix} 1 & 1 & 1 & c \\ 1 & d & -1 & -1 \\ 1 & -d & -1 & 1 \\ 1 & -1 & 1 & -d \end{bmatrix} \right) \otimes \begin{bmatrix} a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \\ a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \end{bmatrix} \quad (3)$$

$$Y = (C_f X C_f^T) \otimes E = \left(\begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} \times [X] \times \begin{bmatrix} 1 & 2 & 1 & c \\ 1 & 1 & -1 & -2 \\ 1 & -1 & -1 & 2 \\ 1 & -2 & 1 & -1 \end{bmatrix} \right) \otimes \begin{bmatrix} a^2 & ab/2 & a^2 & ab/2 \\ ab/2 & b^2/4 & ab/2 & b^2/4 \\ a^2 & ab/2 & a^2 & ab/2 \\ ab/2 & b^2/4 & ab/2 & b^2/4 \end{bmatrix} \quad (4)$$

Произведение матриц CXC^T является «ядром» преобразования. Матрица E представляет собой набор коэффициентов, причем каждый элемент матрицы (CXC^T) должен быть умножен на соответствующий элемент матрицы E (скалярное поэлементное умножение матриц). Коэффициенты a и b те же, что и в формуле (1); коэффициент $d = c/b$ (≈0.414).

Для упрощения процедуры выполнения преобразования предлагается принять $d = 0.5$. Для того чтобы гарантировать ортогональность преобразования, необходимо изменить и коэффициент b, так что получим следующие значения коэффициентов:

$$a = \frac{1}{2},\ b = \sqrt{\frac{2}{5}},\ d = \frac{1}{2}.$$

Второй и четвертый ряды, а также вторая и четвертая колонки матрицы C умножаются на 2, а матрица E изменяется так, чтобы компенсировать данное умножение. (Это позволяет исключить деление на 2 в «ядре» преобразования CXC^T, которое привело бы к потере точности при целочисленной арифметике.) Окончательно преобразование примет вид (4).

Данное преобразование является аппроксимированным (приблизительным) DCT размером 4×4, так как изменение коэффициентов d и b привело к появлению нового преобразования, не идентичного исходному DCT размером 4×4.

Пример

Необходимо обработать при помощи DCT и аппроксимированного преобразования блок X размером 4×4:

j i	0	1	2	3
0	5	11	8	10
1	9	8	4	12
2	1	10	11	4
3	19	6	15	7

Результатом DCT будет матрица (5).

Результатом аппроксимированного преобразования будет матрица (6).

Различие между DCT и целочисленным (7).

Очевидны различия в выходных значениях, которые зависят от коэффициентов b или d. Применяемое в кодеке H.264 приблизительное преобразование позволяет получить практически такую же степень сжатия, что и DCT, и, кроме того, обладает рядом важных преимуществ. Во-первых, «ядро» преобразования CXCT может быть выполнено в целочисленной арифметике при помощи только операций суммирования, вычитания и сдвига (умножение на 2). Во-вторых, динамический диапазон результата действий данного преобразования такой, что везде может быть использована 16-разрядная арифметика без риска выхода за пределы диапазона ±255. В-третьих, масштабирование при помощи матрицы E требует только одного умножения для каждого коэффициента, которое может быть «включено» в процесс квантования (см. ниже).

Обратное преобразование определяется соотношением (8).

В этом случае матрица Y вначале масштабируется матрицей E_i, путем поэлементного умножения. Обратите внимание на коэффициенты ±1/2 в матрицах C и C^T; операции с ними могут быть правильно выполнены без существенной потери точности, потому что коэффициенты Y вначале подвергаются масштабированию.

Прямое и обратное преобразования являются ортогональными, то есть: $T^{-1}(T(X)) = X$.

Квантование

В рекомендации H.264 используется скалярное квантование. К его определению и выполнению предъявляются следующие требования: (a) исключить деление и арифметику с плавающей точкой, (b) включить в процесс квантования масштабирование при помощи матриц E_f и E_i, описанных выше.

Базовой операцией прямого квантования является следующее действие:

$$Z_{ij} = \text{round}\,(Y_{ij}/Q_{step}),$$

где Y_{ij} — коэффициенты преобразования, описанные выше, Q_{step} — шаг квантования, и Z_{ij} — квантованные коэффициенты.

$$Y = AXA^T = \begin{bmatrix} 35.0 & -0.079 & -1.5 & 1.115 \\ -3.299 & -4.768 & 0.443 & -9.010 \\ 5.5 & 3.029 & 2.0 & 4.699 \\ -4.045 & -3.010 & -9.384 & -1.232 \end{bmatrix} \quad (5)$$

$$Y' = (CXC^T) \otimes E = \begin{bmatrix} 35.0 & -0.158 & -1.5 & 1.107 \\ -3.004 & -3.900 & 1.107 & -9.200 \\ 5.5 & 2.688 & 2.0 & 4.901 \\ -4.269 & -3.200 & -9.329 & -2.100 \end{bmatrix} \quad (6)$$

$$Y - Y' = \begin{bmatrix} 0 & 0.079 & 0 & 0.008 \\ -0.295 & -0.868 & -0.664 & 0.190 \\ 0 & 0.341 & 0 & -0.203 \\ 0.224 & 0.190 & -0.055 & 0.868 \end{bmatrix} \quad (7)$$

$$X' = C_i^T (Y \otimes E_i) C_i = \begin{bmatrix} 1 & 1 & 1 & 1/2 \\ 1 & 1/2 & -1 & -1 \\ 1 & -1/2 & -1 & 1 \\ 1 & -1 & 1 & -1/2 \end{bmatrix} \left([Y] \otimes \begin{bmatrix} a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \\ a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \end{bmatrix} \right) \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1/2 & -1/2 & -1 \\ 1 & -1 & -1 & 1 \\ 1/2 & -1 & 1 & -1/2 \end{bmatrix} \quad (8)$$

Таблица 1. Размер шага квантования в кодеке H.264

QP	0	1	2	3	4	5	6	7	8	9	10	11	12	...
Q_{step}	0.625	0.6875	0.8125	0.875	1	1.125	1.25	1.375	1.625	1.75	2	2.25	2.5	
QP	...	18	...	24	...	30	...	36	...	42	...	48	...	51
Q_{step}		5		10		20		40		80		160		224

Общее количество значений шага квантования Q_{step}, определенное в стандарте, равно 52, они передаются как параметр QP, показанный в таблице 1. Обратите внимание, что значение Q_{step} удваивается для каждого приращения QP на 6; Q_{step} увеличивается на 12,5% для каждого приращения QP на 1. Широкий диапазон значений шага квантования позволяет кодирующему устройству точно и гибко управлять обменом между количеством передаваемых бит и качеством.

Коэффициенты масштабирования 2, ab/2 или $b^2/4$ (4) включены в прямое квантование. Вначале входной блок X преобразуется, и получается блок промежуточных коэффициентов $W = CXC^T$. Затем каждый коэффициент W_{ij} квантуется и масштабируется при помощи одной операции:

$$Z_{ij} = \text{round}\left(W_{ij}\frac{PF}{Q_{step}}\right), \qquad (9)$$

где

Позиция в блоке X	PF
(0, 0), (2, 0), (0, 2), или (2, 2)	a^2
(1, 1), (1, 3), (3, 1), или (3, 3)	$b^2/4$
в остальных случаях	ab/2

Операция (PF/Q_{step}) осуществляется в рекомендации H.264 [3] как умножение на коэффициент MF (коэффициент-фактор) и сдвиг вправо, таким образом, исключается операция деления:

$$Z_{ij} = \text{round}\left(W_{ij}\frac{MF}{2^{qbits}}\right), \qquad (10)$$

где

$$\frac{MF}{2^{qbits}} = \frac{PF}{Q_{step}} \text{ и } qbits = 15 + \text{floor}\left(\frac{QP}{6}\right).$$

В целочисленной арифметике (10) может быть выполнено следующим образом:

$$Z_{ij} = (W_{ij}MF + f) \gg qbits, \qquad (11)$$

где «≫» определяет логический сдвиг вправо. В рекомендации определяется коэффициент f как $2^{qbits}/3$ для INTRA-блоков и $2^{qbits}/6$ для INTER-блоков.

Пример

$QP = 4$, следовательно, $Q_{step} = 1.0$.
$(i,j) = (0,0)$, следовательно, $PF = a^2 = 0.25$.
$qbits = 15$, следовательно, $2^{qbits} = 32768$.
Так как

$$\frac{MF}{2^{qbits}} = \frac{PF}{Q_{step}},$$

Таблица 2. Коэффициент умножения MF

QP	Позиция (0, 0), (2, 0), (0, 2), (2, 2)	Позиция (1, 1), (1, 3), (3, 1), (3, 3)	В остальных случаях
0	13107	5243	8066
1	11916	4660	7490
2	10082	4194	6554
3	9362	3647	5825
4	8192	3355	5243
5	7282	2893	4559

получаем $MF = 32768 \times 0.25/1 = 8192$.

Первые 6 значений MF, в зависимости от QP и позиции коэффициента в блоке, могут быть рассчитаны по таблице 2.

Для значений $QP > 5$ коэффициенты умножения MF будут повторяться в соответствии с таблицей 2, а делитель 2^{qbits} будет увеличиваться в два раза для каждого приращения QP на 6. Например, $qbits = 16$ для $6 \le QP \le 11$; $qbits = 17$ для $6 \le QP \le 17$; и так далее.

Обратное квантование

Базовая операция обратного квантования определяется соотношением:

$$Y'_{ij} = Z_{ij}Q_{step}. \qquad (12)$$

Коэффициенты масштабирования для инверсного преобразования (матрица E_i, содержащая коэффициенты a^2, ab и b^2 в зависимости от позиции) включаются в эту операцию совместно с постоянным множителем 64, чтобы избежать ошибки округления:

$$W'_{ij} = Z_{ij} \times Q_{step} \times PF \times 64. \qquad (13)$$

Коэффициенты W'_{ij} являются результатом выполнения «ядра» обратного преобразования ($C_i^T W C_i$: см. (3, 4, 8)). Значения полученных коэффициентов делятся на 64, чтобы компенсировать предварительное масштабирование (это может быть осуществлено при помощи только операций суммирования и сдвига вправо).

Стандарт H.264 непосредственно не определяет Q_{step} или PF. Вместо этого определяется параметр $V = (Q_{step} PF 64)$ для $0 \le QP \le 5$ и для каждого коэффициента в блоке производится пересчет по формуле:

$$W'_{ij} = Z_{ij} \times V_{ij} \times 2^{\text{floor}(QP/6)}. \qquad (14)$$

Пример

- $QP = 3$, следовательно, $Q_{step} = 0.875$ и $2^{\text{floor}(QP/6)} = 1$.
- $(i,j) = (1,2)$, следовательно, $PF = ab = 0.3162$.
- $V = (Q_{step}.PF.64) = 0.875 \times 0.3162 \times 6518$,
- $W'_{ij} = Zij \times 18 \times 1$.

Значение V для $0 \le QP \le 5$ определяется в рекомендации следующим образом (табл. 3):

Таблица 3. Коэффициент обратного квантования V

QP	Позиция (0, 0), (2, 0), (0, 2), (2, 2)	Позиция (1, 1), (1, 3), (3, 1), (3, 3)	В остальных случаях
0	10	16	13
1	11	18	14
2	13	20	16
3	14	23	18
4	16	25	20
5	18	29	23

Коэффициент $2^{\text{floor}(QP/6)}$ в (14) увеличивает результат в два раза при каждом приращении QP на 6.

Преобразование и квантование яркостных DC-коэффициентов блока размером 4×4

Если макроблок размером 16×16 обрабатывается в INTRA-режиме, то каждый остаточный блок размером 4×4 сначала преобразуется «ядром» $C_f X C_f^T$, описанным выше. Полученные яркостные DC-коэффициенты каждого блока размером 4×4 выделяются в отдельный блок и обрабатываются в соответствии с преобразованием Адаманта:

$$Y_D = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \times [W_D] \times \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix}.$$

Матрица W_D представляет собой блок DC-коэффициентов размером 4×4, а матрица Y_D — это блок коэффициентов после преобразования. Полученные коэффициенты $Y_D(i, j)$ делятся на два (с округлением).

После этого коэффициенты $Y_D(i, j)$ квантуются по формуле:

$$Z_{D(ij)} = (Y_{D(ij)}MF + 2f) \gg (qbits + 1),$$

где MF, f и $qbits$ определяются как обычно, при этом MF зависит от позиции (i, j) в пределах блока DC-коэффициентов размером 4×4, как было описано выше.

В декодере обратное преобразование выполняется следующим образом (обратите внимание, что порядок умножения не изменен):

$$W_{QD} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \times [Z_D] \times \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix}.$$

Если QP больше или равно 12, то обратное квантование выполняется следующим образом:

$$W'_{D(ij)} = W_{QD(ij)} \times V_{00} + 2^{1-\text{floor}(QP/6)-2}.$$

Если QP меньше 12, то обратное квантование определяется как:

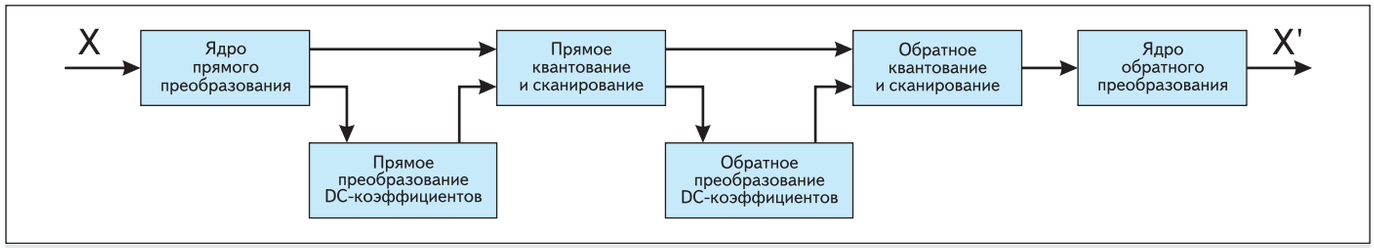

Рис. 14. Процесс прямого и обратного преобразования и квантования

$$W'_{D(ij)} = [W_{QD(ij)} \times V_{00} + 2^{1-\mathrm{floor}(QP/6)}] \gg (2-\mathrm{floor}(QP/6)).$$

Коэффициенты V определяется, как было описано выше. Полученная после процедуры обратного квантования матрица W'_D является блоком *DC*-коэффициентов, которые вставляются в соответствующие блоки, и выполняется «ядро» обратного преобразования ($C_i^T W'C_i$).

Так как в INTRA-макроблоках большая часть энергии сконцентрирована в DC-коэффициентах, данное дополнительное преобразование DC-коэффициентов помогает декоррелировать блок яркостных коэффициентов размером 4×4 (то есть использовать в своих интересах корреляцию между коэффициентами).

Преобразование и квантование блока цветоразностных коэффициентов размером 2×2

Каждый цветоразностный компонент в макроблоке состоит из четырех блоков размером 4×4. Каждый блок размером 4×4 преобразуется так, как это было описано в рвзделе «Формирование INTRA-прогноза». Коэффициенты *DC* каждого блока размером 4×4 группируются в блок размера 2×2 (*WD*), а затем полученный блок преобразуется и квантуется:

$$Y_D = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \times [W_D] \times \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.$$

Квантование блока Y_D размером 2×2 выполняется следующим образом:

$$Z_{D(ij)} = (Y_{D(ij)} MF + 2f) \gg (qbits + 1),$$

где коэффициенты *MF*, *f* и *qbits* определяются как было описано выше.

Затем проводится декодирование, обратная трансформация и обратное квантование:

$$W_{QD} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \times [Z_D] \times \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.$$

Если *QP* больше или равен 6, обратное квантование выполняется следующим образом:

$$W'_{D(ij)} = W_{QD(ij)} \times V_{00} + 2^{1-\mathrm{floor}(QP/6)-1}.$$

Если *QP* меньше, чем 6, обратное квантование выполняется в соответствии с выражением:

$$W'_{D(ij)} = [W_{QD(ij)} \times V_{00}] \gg 1.$$

Коэффициенты, полученные после обратного квантования, размещаются в соответствующих цветоразностных блоках размером 4×4, которые затем обрабатываются, как это описано выше ($C_i^T W'C_i$). Как и в случае с кодированием яркостных DC-коэффициентов в INTRA-режиме, применяется дополнительное преобразование, помогающее декоррелировать блок размера 2×2 цветоразностных DC-коэффициентов и, следовательно, увеличивающее степень сжатия.

На рис. 14 представлен описанный выше полный процесс прямого и обратного преобразования и квантования от момента поступления на вход обрабатываемого блока X до получения на выходе восстановленного блока X'.

Деблокирующий фильтр

Деблокирующая фильтрация осуществляется после обратного преобразования в кодере (при восстановлении и сохранении макроблока для формирования прогноза следующего кадра) и в декодере (при восстановлении и отображении макроблока). Фильтр имеет два преимущества:

- границы блока сглаживаются, улучшая зрительное восприятие декодированных изображений (особенно при больших коэффициентах сжатия);
- отфильтрованный макроблок используется в режиме компенсации движением в кодере, что значительно уменьшает величину остаточных коэффициентов.

Заметим, что в режиме INTRA макроблоки фильтруются, но для формирования прогноза не используются, кроме этого границы кадра не фильтруются в любом режиме.

Фильтрация применяется к вертикальным или горизонтальным границам блоков размером 4×4 в следующем порядке:

1. Фильтрация четырех вертикальных границ яркостного компонента (в порядке a, b, c, d на рис. 15);
2. Фильтрация четырех горизонтальных границ яркостного компонента (в порядке e, f, g, h на рис. 15);
3. Фильтрация двух вертикальных границ каждого цветоразностного компонента (i, j);
4. Фильтрация двух горизонтальных границ каждого цветоразностного компонента (k, l).

Каждая операция фильтрации использует до четырех пикселей с обеих сторон границы (рис. 16). В зависимости от величины текущего шага квантования, режима кодиро-

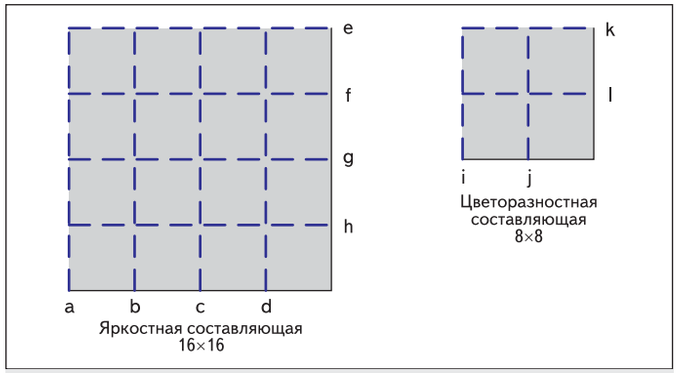

Рис. 15. Порядок фильтрации границ макроблока

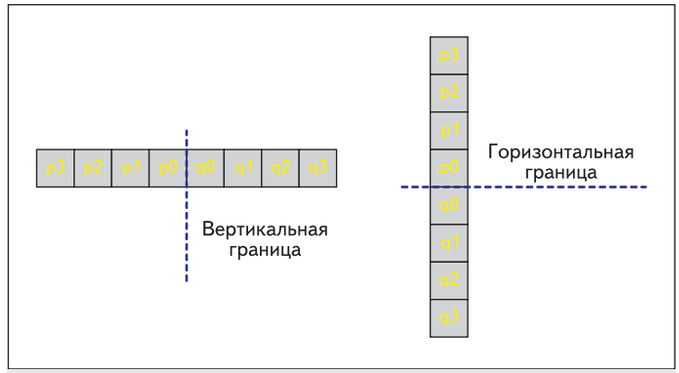

Рис. 16. Выбор смежных пикселей для вертикальных и горизонтальных границ

Исходный кадр | Восстановленный кадр без фильтрации | Восстановленный кадр с фильтрацией

Рис. 17. Пример применения деблокирующего фильтра

вания соседних блоков и градиента выборок изображения поперек границы возможны несколько вариантов фильтрации.

На рис. 17 показан пример применения деблокирующей фильтрации. В данном примере деблокирующий фильтр увеличивает эффективность сжатия незначительно: скорость выходного битового потока уменьшается примерно на 1,5%, а PSNR увеличивается примерно на 1%. Однако субъективное качество восстановленной видеопоследовательности с деблокирующей фильтрацией значительно выше. Увеличение эффективности кодирования, обеспечиваемое деблокирующим фильтром, зависит от содержания последовательности и требуемой выходной скорости закодированного потока.

Энтропийное кодирование

В рекомендации определено несколько алгоритмов кодирования, которые применяются к различным элементам синтаксиса. Так, элементы синтаксиса, относящиеся к кодированию кадра в целом, кодируются кодами фиксированной или переменной длины, а элементы, относящиеся непосредственно к кодированию остаточных коэффициентов, кодируются кодом переменной длины, или арифметическим кодом.

Кодирование переменной длины

В рекомендации определены два вида кодирования с переменной длиной кодового слова:

- экспоненциальное кодирование (Exp-Golomb coding);
- кодирование переменной длины с адаптацией на основе контекста (CAVLC).

Экспоненциальные коды, применяемые в кодеке H.264, — это коды переменной длины с регулярной структурой. Используются для служебных элементов синтаксиса, таких как информация о коэффициенте квантования, режиме кодирования, данные о векторах движения и т. д. Кодовое слово для данного кода имеет следующую структуру:

$$[M]\ [1]\ [INFO],$$

где *INFO* — информационное поле, состоящее из M-bit, *M* — последовательность «0», также состоящая из M-bit.

Создать кодовое слово кодером можно в соответствии с формулами:

$$M = \log2(code\_num + 1),$$

$$INFO = code\_num + 1 - 2M,$$

где *code_num* — кодируемое значение.

Декодирование происходит по следующему алгоритму:

1. Считать *M* начальных нулей, заканчивающихся 1.
2. Считать *M* разрядное поля *INFO*.
3. Определить кодируемое значение по формуле:

$$code\_num = 2M + INFO - 1.$$

Заметим, что для кодируемого значения «0», поля *INFO* и *M* имеют нулевую длину.

Кодируемое значение не является непосредственно тем параметром, который получен в результате обработки кадра. В рекомендации определены три способа замены параметра *v*, полученного в результате обработки кадра, кодируемым значением *code_num*, которое является входным для алгоритма экспоненциального кодирования:

1. *ue*(*v*) — прямое соответствие без знака, *code_num* = *v*.
2. *se*(*v*) — знаковое соответствие:

$$code\_num = 2|v|, \text{ где } v < 0,$$
$$code\_num = 2|v| - 1, \text{ где } v > 0.$$

3. *me*(*v*) — символьное соответствие; параметр *v* соответствует *code_num* согласно таблице, определенной в рекомендации.

Каждый способ (*ue*, *se* и *me*) разработан так, чтобы формировать короткие кодовые слова для часто встречающихся значений параметров и более длинные кодовые слова для менее частых.

Кодирование переменной длины с адаптацией на основе контекста используется для кодирования остаточных коэффициентов. CAVLC разработан с учетом следующих особенностей:

1. После прогнозирования, преобразования и квантования, блоки обычно разрежены (содержат главным образом нули). CAVLC использует кодирование *run-level*, чтобы сжато представить последовательности нулей (вместо последовательности нулей передается одно число, определяющее их количество).
2. Самые большие ненулевые коэффициенты после зигзагообразного сканирования часто завершаются последовательностью ±1. CAVLC позволяет компактно представить последовательность замыкающих высоких частот, равных ±1.
3. Число ненулевых коэффициентов в соседних блоках коррелированно. Поэтому число ненулевых коэффициентов кодируется на основе таблицы поиска, которая выбирается из нескольких, в зависимости от числа ненулевых коэффициентов в соседних блоках.
4. Величина ненулевых коэффициентов обычно выше в начале переупорядоченного массива (около DC-коэффициента) и ниже в конце. CAVLC использует эти свойства, обеспечивая для кодирования уровня остаточного коэффициента несколько таблиц для кодирования, причем выбор той или иной таблицы происходит в зависимости от уровня предыдущего закодированного коэффициента.

Код CAVLC включает следующие поля:

- *coeff_token* — определяет общее количество ненулевых коэффициентов в кодируемом блоке остаточных коэффициентов (*TotalCoeffs*), а также количество коэффициентов в конце блока, равных единице (*T1s*);
- *T*1 — знак единичного коэффициента в конце блока (таких полей может быть три или меньше);
- *Level* — значение ненулевых коэффициентов в блоке (количество полей определяется количеством ненулевых значений коэффициентов кодируемого блока);
- *TotalZeros* — количество нулевых коэффициентов на промежутке от начала блока до последнего ненулевого коэффициента;
- *run_before* — количество нулей перед ненулевым коэффициентом.

Рассмотрим небольшой пример. Имеется блок остаточных коэффициентов размером 4×4:

0	3	−1	0
0	−1	1	0
1	0	0	0
0	0	0	0

Вначале необходимо произвести сканирование данного блока (переупорядочить остаточные коэффициенты). В результате получим одномерный массив:

0,3,0,1,−1,−1,0,1,0,0,0,0,0,0,0,0

Этот массив подвергается кодированию, и в результате получается битовый поток:

000010001110010111101101

Значения его полей приведены в таблице 4. Обратите внимание, что фактически в конце блока четыре единичных коэффициента,

Таблица 4. Пример кодирования остаточных коэффициентов кодом CAVLC

Поле кода	Значение	Код
coeff_token	TotalCoeffs = 5, T1s = 3	0000100
T1 знак (4)	+	0
T1 знак (3)	–	1
T1 знак (2)	–	1
Level (1)	+1	1
Level (0)	+3	0010
TotalZeros	3	111
run_before (4)	1	10
run_before (3)	0	1
run_before (2)	0	1
run_before (1)	1	01
run_before (0)	1	–

но только три из них могут быть закодированы своим знаком. Кроме этого, количество нулей перед первым ненулевым коэффициентом не передается, так как может быть вычислено на приемной стороне из уже полученной информации.

Адаптивное двоичное арифметическое кодирование на основе контекста

Применяемый в кодере H.264 алгоритм арифметического кодирования на основе контекста (CABAC) обеспечивает хорошую степень сжатия вследствие следующих причин:

- выбор вероятностной модели для каждого элемента синтаксиса;
- адаптивная оценка вероятности, основанная на локальной статистике;
- использование арифметического кодирования.

Кодирование символа данных включает следующие стадии:

1. Бинаризация исходного параметра. Этот процесс подобен процессу преобразования символа данных в код переменной длины.
2. Выбор контекстной модели для каждого бита (или «дискрета») бинаризированного символа. Контекстная модель является моделью вероятности для одного или более дискретов. Эта модель может быть выбрана из ряда доступных моделей в зависимости от статистики только что закодированных символов.
3. Непосредственно само арифметическое кодирование. Кодер шифрует каждый дискрет согласно выбранной вероятностной модели.
4. Обновление вероятностной модели. Выбранная модель контекста обновляется на основании фактически закодированных данных (например, если был закодирован дискрет, равный «1», тогда частота повторения «1» увеличивается).

Контекстные модели и схемы бинаризации каждого элемента синтаксиса определены в рекомендации. Существует 267 контекстных моделей, от 0 до 266, для различных элементов синтаксиса. Некоторые модели имеют различное использование в зависимости от типа кодируемого значения. Первоначальный выбор контекстной модели производится в начале кодирования каждой независимой секции видеоизображения (обычно это кадр) в зависимости от коэффициента квантования QP.

В целом, CABAC обеспечивает гораздо более высокую эффективность кодирования по сравнению с VLC за счет большей вычислительной сложности.

На этом краткий обзор рекомендации H.264 закончен. Он подготовлен на основании материалов сайта **ht tp: //w ww.vcodex.c om/**, поэтому вы можете обратиться по указанному адресу для получения более полной информации.