(19) Евразийское патентное ведомство

(11) 038534 (13) B1

(12) ОПИСАНИЕ ИЗОБРЕТЕНИЯ К ЕВРАЗИЙСКОМУ ПАТЕНТУ

(45) Дата публикации и выдачи патента
2021.09.10

(21) Номер заявки
201791457

(22) Дата подачи заявки
2016.01.26

(51) Int. Cl. ***H04N 19/60*** (2014.01)
H04N 19/12 (2014.01)
H04N 19/157 (2014.01)
H04N 19/167 (2014.01)
H04N 19/136 (2014.01)
H04N 19/176 (2014.01)
H04N 19/70 (2014.01)
H04N 19/159 (2014.01)

(54) СПОСОБ И УСТРОЙСТВО ДЛЯ УЛУЧШЕННЫХ МНОЖЕСТВЕННЫХ ПРЕОБРАЗОВАНИЙ ДЛЯ ОСТАТКА ПРЕДСКАЗАНИЯ ВО ВРЕМЯ КОДИРОВАНИЯ/ДЕКОДИРОВАНИЯ ВИДЕОДАННЫХ

(31) **62/107,996; 62/137,038; 15/005,736**
(32) **2015.01.26; 2015.03.23; 2016.01.25**
(33) **US**
(43) **2017.12.29**
(86) **PCT/US2016/014898**
(87) **WO 2016/123091 2016.08.04**

(71)(73) Заявитель и патентовладелец:
КВЭЛКОММ ИНКОРПОРЕЙТЕД (US)

(72) Изобретатель:
Чжао Синь, Ли Сунгвон, Чэнь Цзяньлэ, Чжан Ли, Ли Сян, Чэнь Ин, Карчевич Марта, Лю Хунбинь (US)

(74) Представитель:
Медведев В.Н. (RU)

(56) US-A1-2012170649
KARCZEWICZ M. ET AL.: "Study of coding efficiency improvements beyond HEVC", 113. MPEG MEETING; 19-10-2015 - 23-10-2015; GENEVA; (MOTION PICTURE EXPERT GROUP OR IS0/IEC JTC1/SC29/WG11), no. m37102, 15 October 2015 (2015-10-15), XP030065470, Section 2.4

(57) Способ и устройство для определения преобразований для использования во время кодирования видео и декодирования видео. Видеокодер и видеодекодер могут выбирать поднаборы преобразований, каждое из которых идентифицирует одно или несколько возможных преобразований. Видеокодер и видеодекодер могут определять преобразования из выбранных поднаборов преобразований.

038534 B1

038534 B1

Данная заявка испрашивает приоритет предварительной заявки США № 62/107996, поданной 26 января 2015 г., и предварительной заявки США № 62/137038, поданной 23 марта 2015 г., все содержание каждой из которых включается в этот документ посредством ссылки.

Область техники

Данное раскрытие изобретения относится к кодированию и декодированию видео.

Уровень техники

Возможности цифрового видео могут встраиваться в широкий диапазон устройств, включая цифровые телевизоры, системы цифрового прямого вещания, системы беспроводного вещания, персональные цифровые помощники (PDA), переносные или настольные компьютеры, планшетные компьютеры, электронные книги, цифровые камеры, цифровые записывающие устройства, цифровые мультимедийные проигрыватели, видеоигровые устройства, игровые приставки, сотовые или спутниковые радиотелефоны, так называемые "смартфоны", устройства для видеоконференцсвязи, устройства для потокового видео и т.п. Цифровые видеоустройства реализуют методики сжатия видео, например описанные в стандартах, заданных MPEG-2, MPEG-4, H.263 ITU-T, H.264/MPEG-4 ITU-T, часть 10, Улучшенное кодирование видео (AVC), H.265 ITU-T. Высокоэффективное кодирование видео (HEVC) и расширениях таких стандартов.

Видеоустройства могут эффективнее передавать, принимать, кодировать, декодировать и/или хранить цифровую видеоинформацию с помощью реализации таких методик сжатия видео.

Методики сжатия видео выполняют пространственное (внутреннее, intra-picture) предсказание и/или временное (межкадровое, inter-picture) предсказание для уменьшения или устранения избыточности, присущей видеопоследовательностям. Для блочного кодирования видео можно разбить секцию (slice) видео (например, видеокадр или часть видеокадра) на видеоблоки. Видеоблоки в секции с внутренним кодированием (I) изображения кодируются с использованием пространственного предсказания относительно эталонных выборок в соседних блоках в том же изображении. Видеоблоки в секции с межкадровым кодированием (P или B) изображения могут использовать пространственное предсказание относительно эталонных выборок в соседних блоках в том же изображении или временное предсказание относительно эталонных выборок в других эталонных изображениях.

Пространственное или временное предсказание приводит к блоку с предсказанием для блока, который будет кодироваться. Остаточные данные представляют собой разности пикселей между исходным блоком, который будет кодироваться, и блоком с предсказанием. Блок с межкадровым кодированием кодируется в соответствии с вектором движения, который указывает на блок эталонных выборок, образующих блок с предсказанием, а остаточные данные указывают разность между кодированным блоком и блоком с предсказанием. Блок с внутренним кодированием кодируется в соответствии с режимом внутреннего кодирования и остаточными данными. Для дополнительного сжатия остаточные данные могут быть преобразованы из области пикселей в область преобразования, что приводит к остаточным коэффициентам, которые затем можно квантовать.

Сущность изобретения

Данное раскрытие изобретения описывает методики для определения преобразований для использования в формировании блока коэффициентов из блока преобразований как часть кодирования видео и преобразований для использования в формировании блока преобразований из блока коэффициентов как часть декодирования видео. В некоторых примерах видеокодер может определять множество поднаборов преобразований. Также видеодекодер может определять множество поднаборов преобразований. Видеокодер и видеодекодер могут выбирать поднабор преобразований для множества поднаборов преобразований с использованием неявных методик, которые не обязательно требуют дополнительной сигнализации и определяют преобразования из выбранных поднаборов преобразований. Таким образом, видеокодер и видеодекодер могут выбирать из довольно большого набора преобразований с минимальным увеличением количества информации, которую нужно сигнализировать.

В одном примере раскрытие изобретения описывает способ декодирования видеоданных, содержащий определение множества поднаборов преобразований, при этом каждый поднабор идентифицирует одно или несколько возможных преобразований, где по меньшей мере один поднабор преобразований идентифицирует множество возможных преобразований, выбор первого поднабора преобразований из множества поднаборов преобразований для левого преобразования для текущего блока коэффициентов видеоданных, выбор второго поднабора преобразований из множества поднаборов преобразований для правого преобразования для текущего блока коэффициентов видеоданных, определение левого преобразования из выбранного первого поднабора преобразований, определение правого преобразования из выбранного второго поднабора преобразований, определение текущего блока преобразований на основе левого преобразования, правого преобразования и текущего блока коэффициентов, и восстановление видеоблока на основе текущего блока преобразований и блока с предсказанием.

В одном примере раскрытие изобретения описывает способ кодирования видеоданных, содержащий определение множества поднаборов преобразований, при этом каждый поднабор идентифицирует одно или несколько возможных преобразований, где по меньшей мере один поднабор преобразований идентифицирует множество возможных преобразований, выбор первого поднабора преобразований из

множества поднаборов преобразований для левого преобразования для текущего блока преобразований в видеоблоке видеоданных, выбор второго поднабора преобразований из множества поднаборов преобразований для правого преобразования для блока преобразований в видеоблоке видеоданных, определение левого преобразования из выбранного первого поднабора преобразований, определение правого преобразования из выбранного второго поднабора преобразований, определение текущего блока коэффициентов на основе левого преобразования, правого преобразования и текущего блока преобразований, и формирование потока двоичных сигналов видео, который включает в себя информацию, указывающую коэффициенты текущего блока коэффициентов, используемые для восстановления видеоблока.

В одном примере раскрытие изобретения описывает устройство для декодирования видеоданных, содержащее память видеоданных, сконфигурированную для хранения видеоданных и поднаборов преобразований, при этом каждый поднабор идентифицирует одно или несколько возможных преобразований, где по меньшей мере один поднабор преобразований идентифицирует множество возможных преобразований, и видеодекодер, содержащий интегральные схемы, при этом видеодекодер конфигурируется для определения множества поднаборов преобразований из сохраненных поднаборов преобразований, выбора первого поднабора преобразований из множества поднаборов преобразований для левого преобразования для текущего блока коэффициентов видеоданных, выбора второго поднабора преобразований из множества поднаборов преобразований для правого преобразования для текущего блока коэффициентов видеоданных, определения левого преобразования из выбранного первого поднабора преобразований, определения правого преобразования из выбранного второго поднабора преобразований, определения текущего блока преобразований на основе левого преобразования, правого преобразования и текущего блока коэффициентов, и восстановления видеоблока на основе текущего блока преобразований и блока с предсказанием.

В одном примере раскрытие изобретения описывает устройство для кодирования видеоданных, содержащее память видеоданных, сконфигурированную для хранения видеоданных и поднаборов преобразований, при этом каждый поднабор идентифицирует одно или несколько возможных преобразований, где по меньшей мере один поднабор преобразований идентифицирует множество возможных преобразований, и видеокодер, сконфигурированный для определения множества поднаборов преобразований из сохраненных поднаборов преобразований, выбора первого поднабора преобразований из множества поднаборов преобразований для левого преобразования для текущего блока преобразований в видеоблоке видеоданных, выбора второго поднабора преобразований из множества поднаборов преобразований для правого преобразования для блока преобразований в видеоблоке видеоданных, определения левого преобразования из выбранного первого поднабора преобразований, определения правого преобразования из выбранного второго поднабора преобразований, определения текущего блока коэффициентов на основе левого преобразования, правого преобразования и текущего блока преобразований, и формирования потока двоичных сигналов видео, который включает в себя информацию, указывающую коэффициенты текущего блока коэффициентов, используемые для восстановления видеоблока.

В одном примере раскрытие изобретения описывает устройство для декодирования видеоданных, содержащее средство для определения множества поднаборов преобразований, при этом каждый поднабор идентифицирует одно или несколько возможных преобразований, где по меньшей мере один поднабор преобразований идентифицирует множество возможных преобразований, средство для выбора первого поднабора преобразований из множества поднаборов преобразований для левого преобразования для текущего блока коэффициентов видеоданных, средство для выбора второго поднабора преобразований из множества поднаборов преобразований для правого преобразования для текущего блока коэффициентов видеоданных, средство для определения левого преобразования из выбранного первого поднабора преобразований, средство для определения правого преобразования из выбранного второго поднабора преобразований, средство для определения текущего блока преобразований на основе левого преобразования, правого преобразования и текущего блока коэффициентов, и средство для восстановления видеоблока на основе текущего блока преобразований и блока с предсказанием.

В одном примере раскрытие изобретения описывает невременный машиночитаемый носитель информации, хранящий команды, которые при исполнении побуждают видеодекодер в устройстве для декодирования видео определить множество поднаборов преобразований, при этом каждый поднабор идентифицирует одно или несколько возможных преобразований, где по меньшей мере один поднабор преобразований идентифицирует множество возможных преобразований, выбрать первый поднабор преобразований из множества поднаборов преобразований для левого преобразования для текущего блока коэффициентов видеоданных, выбрать второй поднабор преобразований из множества поднаборов преобразований для правого преобразования для текущего блока коэффициентов видеоданных, определить левое преобразование из выбранного первого поднабора преобразований, определить правое преобразование из выбранного второго поднабора преобразований, определить текущий блок преобразований на основе левого преобразования, правого преобразования и текущего блока коэффициентов, и восстановить видеоблок на основе текущего блока преобразований и блока с предсказанием.

В одном примере раскрытие изобретения описывает устройство для кодирования видеоданных, содержащее средство для определения множества поднаборов преобразований, при этом каждый поднабор

идентифицирует одно или несколько возможных преобразований, где по меньшей мере один поднабор преобразований идентифицирует множество возможных преобразований, средство для выбора первого поднабора преобразований из множества поднаборов преобразований для левого преобразования для текущего блока преобразований в видеоблоке видеоданных, средство для выбора второго поднабора преобразований из множества поднаборов преобразований для правого преобразования для блока преобразований в видеоблоке видеоданных, средство для определения левого преобразования из выбранного первого поднабора преобразований, средство для определения правого преобразования из выбранного второго поднабора преобразований, средство для определения текущего блока коэффициентов на основе левого преобразования, правого преобразования и текущего блока преобразований, и средство для формирования потока двоичных сигналов видео, который включает в себя информацию, указывающую коэффициенты текущего блока коэффициентов, используемые для восстановления видеоблока.

В одном примере раскрытие изобретения описывает невременный машиночитаемый носитель информации, хранящий команды, которые при исполнении побуждают видеокодер в устройстве для кодирования видео определить множество поднаборов преобразований, при этом каждый поднабор идентифицирует одно или несколько возможных преобразований, где по меньшей мере один поднабор преобразований идентифицирует множество возможных преобразований, выбрать первый поднабор преобразований из множества поднаборов преобразований для левого преобразования для текущего блока преобразований в видеоблоке видеоданных, выбрать второй поднабор преобразований из множества поднаборов преобразований для правого преобразования для блока преобразований в видеоблоке видеоданных, определить левое преобразование из выбранного первого поднабора преобразований, определить правое преобразование из выбранного второго поднабора преобразований, определить текущий блок коэффициентов на основе левого преобразования, правого преобразования и текущего блока преобразований, и сформировать поток двоичных сигналов видео, который включает в себя информацию, указывающую коэффициенты текущего блока коэффициентов, используемые для восстановления видеоблока.

Подробности одного или нескольких примеров излагаются на прилагаемых чертежах и в описании ниже. Другие признаки, цели и преимущества станут очевидны из описания, чертежей и из формулы изобретения.

Краткое описание чертежей

Фиг. 1A-1E - таблицы, иллюстрирующие примеры типов преобразований.

Фиг. 2 - блок-схема, иллюстрирующая примерную систему кодирования видео, которая может использовать методики, описанные в данном раскрытии изобретения.

Фиг. 3 - концептуальная схема, иллюстрирующая пример схемы преобразования на основе остаточного квадродерева при высокоэффективном кодирование видео (HEVC).

Фиг. 4 - концептуальная схема, иллюстрирующая пример сканирования коэффициентов на основе группы кодирования в HEVC.

Фиг. 5 - блок-схема, иллюстрирующая примерный видеокодер, который может реализовать методики, описанные в данном раскрытии изобретения.

Фиг. 6 - блок-схема, иллюстрирующая примерный видеодекодер, который может реализовать методики, описанные в данном раскрытии изобретения.

Фиг. 7 - концептуальная схема, иллюстрирующая пример блочной компенсации движения с перекрытием (OBMC) в соответствии с процессом кодирования видео, заданным в стандарте H.263 ITU-T.

Фиг. 8A и 8B - концептуальные схемы, иллюстрирующие части блока для OBMC.

Фиг. 9 - блок-схема алгоритма, иллюстрирующая примерный способ декодирования видеоданных.

Фиг. 10 - блок-схема алгоритма, иллюстрирующая примерный способ кодирования видеоданных.

Подробное описание

Данное раскрытие изобретения имеет отношение к нескольким преобразованиям, применяемым для остатка внутреннего или межкадрового предсказания. Методики можно использовать применительно к улучшенным видеокодекам, например расширениям стандарта высокоэффективного кодирования видео (HEVC) или следующему поколению стандартов кодирования видео.

При кодировании видео видеокодер формирует остаточный блок путем вычитания выборочных значений текущего блока из выборочных значений блока с предсказанием. Видеокодер разделяет остаточный блок на один или несколько блоков преобразований и применяет преобразование (например, дискретное преобразование частоты, такое как дискретное косинусное преобразование (DCT)) к одному или нескольким блокам преобразований, чтобы преобразовать остаточные значения в одном или нескольких блоках преобразований из области пикселей в частотную область. В частотной области преобразованные блоки называются блоками коэффициентов, которые включают в себя один или несколько значений коэффициентов преобразования.

Во время декодирования видеодекодер выполняет обратный процесс. Например, видеодекодер применяет обратное преобразование к блоку коэффициентов, чтобы преобразовать блок коэффициентов в блок преобразований (например, преобразовать из частотной области в область пикселей). Блок преобразований является одним блоком остаточного блока, и видеодекодер добавляет остаточные значения остаточного блока к выборочным значениям блока с предсказанием, чтобы восстановить текущий блок.

Только для простоты описания данное раскрытие изобретения описывает видеокодер и видеодекодер как определяющие преобразование, используемое для процесса кодирования и декодирования соответственно. Однако следует понимать, что видеокодер применяет преобразование к блоку преобразований, чтобы сформировать блок коэффициентов, и что видеодекодер применяет обратное преобразование к блоку коэффициентов, чтобы восстановить блок преобразований. Соответственно, преобразование, которое применяет видеодекодер, является обратным к преобразованию, которое применяет видеокодер. Поэтому в данном раскрытии изобретения, когда видеодекодер описывается как определяющий преобразование и/или применяющий преобразование, следует понимать, что видеодекодер определяет преобразование, которое является обратным к преобразованию, определенному видеокодером, и/или что видеодекодер применяет преобразование, которое является обратным к преобразованию, применяемому видеокодером.

Данное раскрытие изобретения описывает примерные методики для определения преобразования, которое применяется к блоку преобразований в остаточных значениях для кодирования коэффициентов преобразования или применяется к блоку коэффициентов в коэффициентах преобразования для декодирования остаточных значений. Например, видеокодер и видеодекодер могут создавать множество поднаборов преобразований, при этом каждое поднабор преобразований идентифицирует множество возможных преобразований. Возможные преобразования относятся к разным типам преобразований, например, разным типам DCT и разным типам дискретных синусных преобразований (DST). Видеокодер и видеодекодер выбирают поднабор (поднаборы) преобразований и определяют преобразования из выбранного поднабора (поднаборов) преобразований, которые используются для определения блока коэффициентов из блока преобразований для кодирования видео или блока преобразований из блока коэффициентов для декодирования видео.

Таким образом, видеокодер и видеодекодер могут определять, какие преобразования использовать, из большего набора возможных преобразований, что дает возможность лучшей адаптации к меняющейся статистике блока преобразований без чрезмерной нагрузки на полосу пропускания потока двоичных сигналов. Например, некоторые методики ограничивают то, сколько преобразований доступно, что может привести к плохой производительности кодирования, поскольку статистика блока преобразований такова, что никакие из доступных преобразований не эффективны. Могут быть и другие, лучшие преобразования, но эти преобразования недоступны из-за ограничений.

В описанных в данном раскрытии изобретения методиках, поскольку доступно больше преобразований, видеокодер и видеодекодер могут использовать преобразование, которое обеспечивает лучшую производительность кодирования, чем возможна при ограниченном наборе преобразований. Кроме того, как описано подробнее, остается низкой служебная нагрузка сигнализации, используемая для указания, какое преобразование нужно использовать, чтобы можно было добиться эффективности кодирования, имея больше доступных преобразований и сохраняя низкое влияние на полосу пропускания.

Например, вместо привлечения сигнализированной информации в потоке двоичных сигналов видеодекодер может выбирать, какой поднабор (поднаборы) преобразований использовать, на основе неявных методик, например, на основе режима внутреннего предсказания, местоположения блока преобразований и т. п. Тогда видеодекодер может определять, какое преобразование (преобразования) использовать, из выбранного поднабора (поднаборов) преобразований по возможности на основе одного или нескольких индексов поднаборов преобразований для соответствующих выбранных поднаборов преобразований, сигнализированных в потоке двоичных сигналов, или других факторов, включая, но не только, количество ненулевых коэффициентов, сумму ненулевых коэффициентов или положение ненулевых коэффициентов в блоке коэффициентов.

Даже когда сигнализируется индекс поднаборы преобразований для соответствующего поднабора (поднаборов) преобразований, служебную нагрузку сигнализации можно сохранить низкой, потому что значение индекса охватывает только диапазон поднаборы преобразований, а не все возможные преобразования. Например, предположим, что имеется вплоть до 16 возможных преобразований, и что поднабор преобразований включает в себя три возможных преобразования. В этом случае значение индекса будет меняться от 0 до 2, тогда как индекс к списку всех преобразований менялся бы от 0 до 15. Сигнализация меньших значений, например от 0 до 2, может требовать меньше разрядов, чем сигнализация больших значений.

Перед описанием способа, которым создаются и выбираются поднаборы преобразований, нижеследующее описывает стандарты кодирования видео, DCT и DST вообще, разные типы DCT и DST и некоторые существующие методики DCT и DST. Затем раскрытие изобретения описывает некоторые проблемы в существующих методиках с последующими примерными методиками, которые могут решить те проблемы.

Стандарты кодирования видео включают в себя H.261 ITU-T, MPEG-1 Visual ISO/IEC, H.262 ITU-T или MPEG-2 Visual ISO/IEC, H.263 ITU-T, MPEG-4 Visual ISO/IEC и H.264 ITU-T (также известный как MPEG-4 AVC ISO/IEC), включая его расширения Масштабируемого кодирования видео (SVC) и Многовидового кодирования видео (MVC). К тому же недавно Объединенной командой по кодированию видео (JCT-VC) из Экспертной группы в области кодирования видео (VCEG) ITU-T и Экспертной группы по

движущимся изображениям (MPEG) ISO/IEC разработан новый стандарт кодирования видео, а именно Высокоэффективное кодирование видео (HEVC). Окончательный проект спецификации HEVC, называемый в дальнейшем WD HEVC, доступен по ссылке:

http://phenix.int-evry.fr/jct/doc_end_user/documents/14_Vienna/wgll/JCTVC-N1003-v1.zip.

Окончательным проектом стандарта HEVC является H.265 ITU-T, Series H: Audiovisual and Multimedia Systems, Infrastructure of audiovisual services - Coding of moving video, Advanced video coding for generic audiovisual services, Международный союз электросвязи, октябрь 2014 г., и он доступен по ссылке:

http://www.itu.int/rec/T-REC-H.265-201410-I/en.

Нижеследующее является описанием дискретных синусных и косинусных преобразований. Преобразование указывает процесс получения альтернативного представления входного сигнала. Например, преобразование преобразует значения из области пикселей в частотную область (например, при кодировании видео) или из частотной области в область пикселей (например, при декодировании видео). При условии N-точечного вектора $x=[x_0, x_1, ..., x_{N-1}]^T$ и набора заданных векторов $\{\phi_0, \phi_1, ..., \phi_{M-1}\}$, x можно приблизительно выразить или точно представить с использованием линейной комбинации $\phi_0, \phi_1, ..., \phi_{M-1}$, что можно сформулировать следующим образом:

$$\hat{x} = \sum_{i=0}^{M-1} f_i \cdot \phi_i$$

где $\hat{x}$ может быть приближением или эквивалентом x, вектор $f=[f_0, f_1, f_2, ..., f_{M-1}]$ называется вектором коэффициентов преобразования, а $\{\phi_0, \phi_1, ..., \phi_{M-1}\}$ являются базисными векторами преобразований.

В сценарии кодирования видео коэффициенты преобразования приблизительно некоррелированы и разрежены, то есть энергия входного вектора x сосредоточена только на нескольких коэффициентах преобразования, а оставшееся большинство коэффициентов преобразования обычно близко к 0. Например, когда видеокодер преобразует блок преобразований в блок коэффициентов, ненулевые значения коэффициентов в блоке коэффициентов стремятся сгруппироваться в верхнем левом углу блока коэффициентов, и большинство значений коэффициентов нули. Ненулевые коэффициенты, сгруппированные возле верхнего левого угла блока коэффициентов, отражают низкочастотные составляющие, тогда как значения коэффициентов возле нижнего правого угла блока коэффициентов, которые стремятся к нулю, отражают высокочастотные составляющие.

При определенных входных данных оптимальным преобразованием в плане уплотнения энергии является так называемое преобразование Карунена - Лоэва (KLT), которое использует собственные векторы ковариационной матрицы входных данных в качестве базисных векторов преобразований. Поэтому KLT фактически является зависимым от данных преобразованием и не имеет общей математической формулировки. Однако при некоторых допущениях, например, что входные данные образуют стационарный марковский процесс первого порядка, в литературе было доказано, что соответствующее KLT фактически является членом синусоидального семейства унитарных преобразований, которое описывается в Jain, A.K., A sinusoidal family of unitary transforms, IEEE Trans. on Pattern Analysis and Machine Intelligence, 1, 356, 1979. Синусоидальное семейство унитарных преобразований указывает преобразования, использующие базисные векторы преобразований, сформулированные следующим образом:

$$\phi_m(k) = A \cdot e^{ike} + B \cdot e^{-ike}$$

где e - основание натурального логарифма, приблизительно равное 2,71828, A, B и e обычно комплексные и зависят от значения m.

Несколько общеизвестных преобразований, включая дискретное преобразование Фурье, косинусное, синусное и KLT (для стационарных марковских процессов первого порядка), являются членами этого синусоидального семейства унитарных преобразований. В соответствии с S. A. Martucci, "Symmetric convolution and the discrete sine and cosine transforms", IEEE Trans. Sig. Processing SP-42, 1038-1051 (1994), полные семейства дискретного косинусного преобразования (DCT) и дискретного синусного преобразования (DST) включают в себя в общем 16 преобразований на основе разных типов, то есть разных значений A, B и e, и полное определение разных типов DCT и DST приводится ниже.

Предположим, что входной N-точечный вектор обозначается $x=[x_0, x_1, ..., x_{N-1}]^T$, и он преобразуется в другой N-точечный вектор коэффициентов преобразования, обозначенный $y=[y_0, y_1, ..., Y_{N-1}]^T$, путем умножения матрицы, процесс чего можно дополнительно проиллюстрировать в соответствии с одной из следующих формулировок преобразований, где k меняется от 0 до N-1 включительно: Тип-I DCT (DCT-1):

$$y_k = \sum_{n=0}^{N-1} \sqrt{\frac{2}{N-1}} \cos\left(\frac{\pi \cdot n \cdot k}{N-1}\right) \cdot w_0 \cdot w_1 \cdot x_n,$$

где

$$w_0 = \begin{cases} \frac{1}{\sqrt{2}}, & \text{если } n = 0 \text{ или } n = N-1 \\ 1 & \text{в противном случае} \end{cases}, \quad w_1 = \begin{cases} \frac{1}{\sqrt{2}}, & \text{если } k = 0 \text{ или } k = N-1 \\ 1 & \text{в противном случае} \end{cases}$$

Тип-II DCT (DCT-2):

$$y_k = \sum_{n=0}^{N-1} \sqrt{\frac{2}{N}} \cos\left(\frac{\pi \cdot (n+0.5) \cdot k}{N-1}\right) \cdot w_0 \cdot x_n,$$

где

$$w_0 = \begin{cases} \frac{1}{\sqrt{2}}, & \text{если } k = 0 \\ 1 & \text{в противном случае} \end{cases}$$

Тип-III DCT (DCT-3):

$$y_k = \sum_{n=0}^{N-1} \sqrt{\frac{2}{N}} \cos\left(\frac{\pi \cdot n \cdot (k+0.5)}{N}\right) \cdot w_0 \cdot x_n,$$

где

$$w_0 = \begin{cases} \frac{1}{\sqrt{2}}, & \text{если } n = 0 \\ 1 & \text{в противном случае} \end{cases}$$

Тип-IV DCT (DCT-4):

$$y_k = \sum_{n=0}^{N-1} \sqrt{\frac{2}{N}} \cos\left(\frac{\pi \cdot (n+0.5) \cdot (k+0.5)}{N}\right) \cdot x_n,$$

Тип-V DCT (DCT-5):

$$y_k = \sum_{n=0}^{N-1} \sqrt{\frac{2}{N-0.5}} \cos\left(\frac{\pi \cdot n \cdot k}{N-0.5}\right) \cdot w_0 \cdot w_1 \cdot x_n,$$

где

$$w_0 = \begin{cases} \frac{1}{\sqrt{2}}, & \text{если } n = 0 \\ 1 & \text{в противном случае} \end{cases}, \quad w_1 = \begin{cases} \frac{1}{\sqrt{2}}, & \text{если } k = 0 \\ 1 & \text{в противном случае} \end{cases}$$

Тип-VI DCT (DCT-6):

$$y_k = \sum_{n=0}^{N-1} \sqrt{\frac{2}{N-0.5}} \cos\left(\frac{\pi \cdot (n+0.5) \cdot k}{N-0.5}\right) \cdot w_0 \cdot w_1 \cdot x_n,$$

где

$$w_0 = \begin{cases} \frac{1}{\sqrt{2}}, & \text{если } n = N-1 \\ 1 & \text{в противном случае} \end{cases}, \quad w_1 = \begin{cases} \frac{1}{\sqrt{2}}, & \text{если } k = 0 \\ 1 & \text{в противном случае} \end{cases}$$

Тип-VII DCT (DCT-7):

$$y_k = \sum_{n=0}^{N-1} \sqrt{\frac{2}{N-0.5}} \cos\left(\frac{\pi \cdot n \cdot (k+0.5)}{N-0.5}\right) \cdot w_0 \cdot w_1 \cdot x_n,$$

где

$$w_0 = \begin{cases} \frac{1}{\sqrt{2}}, & \text{если } n = 0 \\ 1 & \text{в противном случае} \end{cases}, \quad w_1 = \begin{cases} \frac{1}{\sqrt{2}}, & \text{если } k = N-1 \\ 1 & \text{в противном случае} \end{cases}$$

Тип-VIII DCT (DCT-8):

$$y_k = \sum_{n=0}^{N-1} \sqrt{\frac{2}{N+0.5}} \cos\left(\frac{\pi \cdot (n+0.5) \cdot (k+0.5)}{N+0.5}\right) \cdot x_n,$$

Тип-I DST (DST-1):

$$y_k = \sum_{n=0}^{N-1} \sqrt{\frac{2}{N+1}} \sin\left(\frac{\pi \cdot (n+1) \cdot (k+1)}{N+1}\right) \cdot x_n,$$

Тип-II DST (DST-2):

$$y_k = \sum_{n=0}^{N-1} \sqrt{\frac{2}{N}} \sin\left(\frac{\pi \cdot (n+0.5) \cdot (k+1)}{N}\right) \cdot w_0 \cdot x_n,$$

где

$$w_0 = \begin{cases} \frac{1}{\sqrt{2}}, & \text{если } k = N-1 \\ 1 & \text{в противном случае} \end{cases}$$

Тип-III DST (DST-3) :

$$y_k = \sum_{n=0}^{N-1} \sqrt{\frac{2}{N}} \sin\left(\frac{\pi \cdot (n+1) \cdot (k+0.5)}{N}\right) \cdot w_0 \cdot x_n ,$$

где

$$w_0 = \begin{cases} \frac{1}{\sqrt{2}}, & \text{если } n = N-1 \\ 1 & \text{в противном случае} \end{cases}$$

Тип-IV DST (DST-4):

$$y_k = \sum_{n=0}^{N-1} \sqrt{\frac{2}{N}} \sin\left(\frac{\pi \cdot (n+0.5) \cdot (k+0.5)}{N}\right) \cdot x_n ,$$

Тип-V DST (DST-5):

$$y_k = \sum_{n=0}^{N-1} \sqrt{\frac{2}{N+0.5}} \sin\left(\frac{\pi \cdot (n+1) \cdot (k+1)}{N+0.5}\right) \cdot x_n ,$$

Тип-VI DST (DST-6):

$$y_k = \sum_{n=0}^{N-1} \sqrt{\frac{2}{N+0.5}} \sin\left(\frac{\pi \cdot (n+0.5) \cdot (k+1)}{N+0.5}\right) \cdot x_n ,$$

Тип-VII DST (DST-7):

$$y_k = \sum_{n=0}^{N-1} \sqrt{\frac{2}{N+0.5}} \sin\left(\frac{\pi \cdot (n+1) \cdot (k+0.5)}{N+0.5}\right) \cdot x_n ,$$

Тип-VIII DST (DST-8):

$$y_k = \sum_{n=0}^{N-1} \sqrt{\frac{2}{N-0.5}} \sin\left(\frac{\pi \cdot (n+0.5) \cdot (k+0.5)}{N-0.5}\right) \cdot w_0 \cdot w_1 \cdot x_n ,$$

где

$$w_0 = \begin{cases} \frac{1}{\sqrt{2}}, & \text{если } n = N-1 \\ 1 & \text{в противном случае} \end{cases} , \quad w_1 = \begin{cases} \frac{1}{\sqrt{2}}, & \text{если } k = N-1 \\ 1 & \text{в противном случае} \end{cases}$$

Вышеприведенное предоставляет примеры разных типов DCT и DST, в общем имеется 16 типов преобразований. Тип преобразования задается математической формулировкой базисной функции преобразования. Не следует путать тип преобразования и размер преобразования. Тип преобразования относится к базисной функции, тогда как размер преобразования относится к размеру преобразования. Например, 4-точечное DST-VII и 8-точечное DST-VII имеют один и тот же тип преобразования независимо от значения N (например, 4-точечное или 8-точечное).

Без потери общности все вышеупомянутые типы преобразований можно представить с использованием нижеследующей обобщенной формулировки:

$$y_m = \sum_{n=0}^{N-1} T_{m,n} \cdot x_n ,$$

где T - матрица преобразования, заданная определением одного конкретного преобразования, например Тип-I DCT ~ Тип-VIII DCT или Тип-I DST ~ Тип-VIII DST, и векторы строк T, например $[T_{i,0}\ T_{i,1},\ T_{i,2},\ \ldots,\ T_{i,N-1}]$ являются i-тыми базисными векторами преобразований. Преобразование, применяемое к N-точечному входному вектору, называется N-точечным преобразованием.

Также отметим, что вышеприведенные формулировки преобразований, которые применяются к 1-мерным входным данным x, можно представить в виде матричного умножения ниже:

$$y = \boldsymbol{T} \cdot x$$

где T указывает матрицу преобразования, x указывает входной вектор данных, а y указывает выходной вектор коэффициентов преобразования.

Например, видеокодер может выполнять матричное умножение $y = \boldsymbol{T} \cdot x$ для формирования вектора коэффициентов преобразования. Видеодекодер может выполнять обратное матричное умножение для формирования вектора преобразования из вектора коэффициентов преобразования.

Преобразования, которые предложены выше, применяются к 1-мерным входным данным, и преоб-

разования также можно расширить для источников 2-мерных входных данных. Предположим, что X является входным массивом данных M×N. Типичные способы применения преобразования к 2-мерным входным данным включают в себя разделимые и неразделимые 2-мерные преобразования.

Разделимое 2-мерное преобразование последовательно применяет 1-мерные преобразования для горизонтальных и вертикальных векторов X, как сформулировано ниже:

$$Y = C \cdot X \cdot R^T$$

где C и R обозначают соответственно заданные матрицы преобразований M×M и N×N.

Из этой формулировки можно увидеть, что C применяет 1-мерные преобразования для векторов столбцов в X, тогда как R применяет 1-мерные преобразования для векторов строк в X. В последующей части данного раскрытия изобретения для простоты обозначим C и R как левое (вертикальное) и правое (горизонтальное) преобразования, и они образуют пару преобразований. Существуют случаи, когда C равно R и является ортогональной матрицей. В таком случае разделимое 2-мерное преобразование определяется всего лишь одной матрице преобразования.

Неразделимое 2-мерное преобразование сначала реорганизует все элементы X в один вектор, а именно X', выполняя в качестве примера следующее математическое отображение:

$$X'_{(i \cdot N + j)} = X_{i,j}$$

Затем для X' применяется 1-мерное преобразование T', как представлено ниже:

$$Y = T' \cdot X$$

где T' - матрица преобразования (M*N)×(M*N).

При кодировании видео всегда применяются разделимые 2-мерные преобразования, поскольку это требует гораздо меньшего числа операций (добавление, умножение) по сравнению с 1-мерным преобразованием. Как подробнее описано ниже, данное раскрытие изобретения описывает примерные методики, с помощью которых видеокодер и видеодекодер выбирают левое и правое преобразования.

Например, видеокодер и видеодекодер могут определить множество поднаборов преобразований, при этом каждое поднабор преобразований идентифицирует множество возможных преобразований. В качестве примера 16 возможных преобразований (например, DCT-1 - DCT-8 и DST-1 - DST-8) видеокодер и видеодекодер могут определить три поднаборы преобразований, и каждое из поднаборов преобразований включает в себя два или более из 16 преобразований. Видеокодер и видеодекодер могут выбрать одно из трех поднаборов преобразований и определить левое преобразование (например, C) из выбранного поднаборы преобразований и выбрать одно из трех поднаборов преобразований и определить правое преобразование (например, R) из выбранного поднаборы преобразований. Выбранные поднаборы преобразований могут быть разными поднаборами либо одинаковыми поднаборами.

Нижеследующее является описанием типов преобразований, применяемых в HEVC. В традиционных видеокодеках, например H.264/AVC, всегда применяется целочисленное приближение Типа-II 4-точечного и 8-точечного дискретного косинусного преобразования (DCT) для остатка внутреннего и межкадрового предсказания. Остаток внутреннего предсказания относится к остатку от внутреннего предсказания, а остаток межкадрового предсказания относится к остатку от межкадрового предсказания. Остаток, межкадровое предсказание и внутреннее предсказание подробнее описываются ниже. Обычно остаточный блок разделяется на множество блоков преобразований. При кодировании видео преобразования применяются к каждому из блоков преобразований, чтобы сформировать блоки коэффициентов. При декодировании видео преобразования применяются к каждому из блоков коэффициентов, чтобы сформировать блоки преобразований и восстановить остаточный блок.

Чтобы лучше приспособиться к различной статистике остаточных выборок, в видеокодеке нового поколения используются более гибкие типы преобразований помимо Типа-II DCT. Например, в HEVC целочисленное приближение Типа-VII 4-точечного дискретного синусного преобразования (DST) используется для остатка внутреннего предсказания, которое теоретически доказано и опытным путем подтверждено, что Тип-VII DST эффективнее Типа-II DCT для остаточных векторов, сформированных в направлениях внутреннего предсказания, например, Тип-VII DST эффективнее Типа-II DCT для остаточных векторов строк, сформированных горизонтальным направлением внутреннего предсказания. См., например, J. Han, A. Saxena and K. Rose, "Towards jointly optimal spatial prediction and adaptive transform in video/image coding", IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), March 2010, pp. 726-729.

В HEVC целочисленное приближение Типа-VII 4-точечного DST применяется только для остаточных блоков яркости внутреннего предсказания 4×4 (остаточные блоки яркости внутреннего предсказания подробнее описываются ниже). 4-точечное DST-VII, используемое в HEVC, показано на фиг. 1A.

В HEVC для остаточных блоков, которые не являются остаточными блоками яркости внутреннего предсказания 4×4, также применяются целочисленные приближения Типа-II 4-точечного, 8-точечного, 16-точечного и 32-точечного DCT. Фиг. 1B иллюстрирует пример 4-точечного DCT-II; фиг. 1C иллюстрирует пример 8-точечного DCT-II; фиг. 1D иллюстрирует пример 16-точечного DCT-II; и фиг. 1E иллюстрирует пример 32-точечного DCT-II. Фиг. 1A-1E иллюстрируют примеры DCT типа II разного размера,

и подобно фиг. 1A-1E имеются примеры N-точечных DCT и DST разных типов.

Фиг. 2 - блок-схема, иллюстрирующая примерную систему 10 кодирования видео, которая может использовать методики из данного раскрытия изобретения. При использовании в данном документе термин "кодировщик видео" в общем относится к видеокодерам и видеодекодерам. В данном раскрытии изобретения термины "кодирование видео" или "кодирование" могут относиться в общем к кодированию видео или декодированию видео. Видеокодер 20 и видеодекодер 30 в системе 10 кодирования видео представляют собой примеры устройств, которые могут конфигурироваться для выполнения методик для улучшенных множественных преобразований для остатка предсказания в соответствии с различными примерами, описанными в данном раскрытии изобретения.

Как показано на фиг. 1, система 10 кодирования видео включает в себя устройство-источник 12 и устройство-адресат 14. Устройство-источник 12 формирует кодированные видеоданные. Соответственно, устройство-источник 12 может называться устройством кодирования видео или аппаратом кодирования видео. Устройство-адресат 14 может декодировать кодированные видеоданные, сформированные устройством-источником 12.

Соответственно, устройство-адресат 14 может называться устройством декодирования видео или аппаратом декодирования видео. Устройство-источник 12 и устройство-адресат 14 могут быть примерами устройств кодирования видео или аппаратов кодирования видео.

Устройство-источник 12 и устройство-адресат 14 могут быть выполнены в виде любого из широкого диапазона устройств, включающего в себя настольные компьютеры, мобильные вычислительные устройства, блокнотные (например, переносные) компьютеры, планшетные компьютеры, телевизионные приставки, телефонные трубки, например так называемые "интеллектуальные" телефоны, телевизоры, камеры, устройства отображения, цифровые мультимедийные проигрыватели, игровые приставки, автомобильные компьютеры или т.п.

Устройство-адресат 14 может принимать кодированные видеоданные от устройства-источника 12 по каналу 16. Канал 16 может быть выполнен в виде одного или нескольких носителей либо устройств, допускающих перемещение кодированных видеоданных от устройства-источника 12 к устройству-адресату 14. В одном примере канал 16 может быть выполнен в виде одного или нескольких средств связи, которые дают устройству-источнику 12 возможность передавать кодированные видеоданные непосредственно к устройству-адресату 14 в реальном масштабе времени. В этом примере устройство-источник 12 может модулировать кодированные видеоданные в соответствии с неким стандартом связи, например протоколом беспроводной связи, и может передавать модулированные видеоданные устройству-адресату 14. Одно или несколько средств связи могут включать в себя средства беспроводной и/или проводной связи, например радиочастотный (РЧ) спектр либо одну или несколько физических линий передачи. Одно или несколько средств связи могут образовывать часть пакетной сети, например локальной сети, региональной сети или глобальной сети (например, Интернет). Одно или несколько средств связи могут включать в себя маршрутизаторы, коммутаторы, базовые станции или другое оборудование, которое упрощает связь от устройства-источника 12 к устройству-адресату 14.

В другом примере канал 16 может включать в себя носитель информации, который хранит кодированные видеоданные, сформированные устройством-источником 12. В этом примере устройство-адресат 14 может обращаться к носителю информации, например, посредством дискового доступа или карточного доступа. Носитель информации может включать в себя ряд локально доступных носителей информации, например диски Blu-ray, DVD, CD-ROM, флэш-память или другие подходящие цифровые носители информации для хранения кодированных видеоданных.

В дополнительном примере канал 16 может включать в себя файловый сервер или другое промежуточное запоминающее устройство, которое хранит кодированные видеоданные, сформированные устройством-источником 12. В этом примере устройство-адресат 14 может обращаться к кодированным видеоданным, сохраненным на файловом сервере или другом промежуточном запоминающем устройстве, посредством потоковой передачи или загрузки. Файловый сервер может быть неким типом сервера, допускающим хранение кодированных видеоданных и передачу кодированных видеоданных устройству-адресату 14.

Примерные файловые серверы включают в себя веб-серверы (например, для веб-сайта), серверы на протоколе передачи файлов (FTP), сетевые устройства хранения (NAS) и локальные накопители на дисках.

Устройство-адресат 14 может обращаться к кодированным видеоданным посредством стандартного информационного соединения, например Интернет-соединения.

Примерные типы информационных соединений могут включать в себя радиоканалы (например, соединения Wi-Fi), проводные соединения (например, DSL, кабельный модем и т.п.) или их сочетания, которые подходят для обращения к кодированным видеоданным, сохраненным на файловом сервере. Передача кодированных видеоданных от файлового сервера может быть потоковой передачей, загрузкой или их сочетанием.

Методики из данного раскрытия изобретения не ограничиваются беспроводными приложениями или настройками. Методики могут применяться к кодированию видео в поддержку ряда мультимедий-

ных приложений, таких как эфирные телевизионные передачи, кабельные телевизионные передачи, спутниковые телевизионные передачи, передачи потокового видео, например по Интернету, кодирование видеоданных для сохранения на носитель информации, декодирование видеоданных, сохраненных на носителе информации, или других приложений. В некоторых примерах система 10 кодирования видео может конфигурироваться для поддержки однонаправленной или двунаправленной передачи видео для поддержки таких приложений, как потоковая передача видео, воспроизведение видео, телевизионное вещание и/или видеотелефония.

Проиллюстрированная на фиг. 2 система 10 кодирования видео является всего лишь примером, и методики из данного раскрытия изобретения могут применяться к настройкам кодирования видео (например, кодирования видео или декодирования видео), которые не обязательно включают в себя какую-либо передачу данных между устройствами кодирования и декодирования. В других примерах данные извлекаются из локального запоминающего устройства, передаются в потоке по сети или т.п. Устройство кодирования видео может кодировать и сохранять данные в запоминающем устройстве, и/или устройство декодирования видео может извлекать и декодировать данные из запоминающего устройства. Во многих примерах кодирование и декодирование выполняется устройствами, которые не взаимодействуют друг с другом, а просто кодируют данные в запоминающем устройстве и/или извлекают и декодируют данные из запоминающего устройства.

В примере из фиг. 2 устройство-источник 12 включает в себя источник 18 видео, видеокодер 20 и интерфейс 22 вывода. В некоторых примерах интерфейс 22 вывода может включать в себя модулятор/демодулятор (модем) и/или передатчик. Источник 18 видео может включать в себя устройство видеозахвата (например видеокамеру), видеоархив, содержащий ранее захваченные видеоданные, интерфейс источника видеосигнала для приема видеоданных от поставщика видеоконтента и/или систему компьютерной графики для формирования видеоданных, либо сочетание таких источников видеоданных.

Видеокодер 20 может кодировать видеоданные от источника 18 видео. В некоторых примерах устройство-источник 12 передает кодированные видеоданные напрямую устройству-адресату 14 через интерфейс 22 вывода. В других примерах кодированные видеоданные также можно сохранить на носителе информации или файловом сервере для последующего обращения устройства-адресата 14 для декодирования и/или воспроизведения.

В примере из фиг. 2 устройство-адресат 14 включает в себя интерфейс 28 ввода, видеодекодер 30 и устройство 32 отображения. В некоторых примерах интерфейс 28 ввода включает в себя приемник и/или модем. Интерфейс 28 ввода может принимать кодированные видеоданные по каналу 16. Устройство 32 отображения может объединяться с устройством-адресатом 14 или может быть внешним по отношению к нему. Обычно устройство 32 отображения отображает декодированные видеоданные. Устройство 32 отображения может быть выполнено в виде ряда устройств отображения, например жидкокристаллического дисплея (LCD), плазменного дисплея, дисплея на органических светоизлучающих диодах (OLED) или другого типа устройства отображения.

Видеокодер 20 и видеодекодер 30 могут быть реализованы в виде любой из ряда подходящих схем, например одного или нескольких микропроцессоров, цифровых процессоров сигналов (DSP), специализированных интегральных схем (ASIC), программируемых пользователем вентильных матриц (FPGA), дискретной логики, аппаратных средств или любых их сочетаний. Если методики частично реализуются в программном обеспечении, то устройство может хранить команды для программного обеспечения на подходящем постоянном машиночитаемом носителе информации и может исполнять команды на аппаратных средствах, использующих один или несколько процессоров, для выполнения методик из данного раскрытия изобретения. Любое из вышеупомянутого (включая аппаратные средства, программное обеспечение, сочетание аппаратных средств и программного обеспечения, и т. п.) может считаться одним или несколькими процессорами. Каждый из видеокодера 20 и видеодекодера 30 может включаться в один или несколько кодеров или декодеров, любой из которых может встраиваться как часть объединенного кодера/декодера (кодека) в соответствующем устройстве.

Данное раскрытие изобретения в целом может ссылаться на видеокодер 20, "сигнализирующий" или "передающий" некоторую информацию другому устройству, например видеодекодеру 30. Термин "сигнализация" или "передача" в целом может относиться к передаче синтаксических элементов и/или других данных, используемых для декодирования сжатых видеоданных. Такая передача может происходить в реальном масштабе времени или почти в реальном масштабе времени. Наоборот, такая передача может происходить за некий промежуток времени, например, она могла бы происходить при сохранении синтаксических элементов на машиночитаемый носитель информации в кодированном потоке двоичных сигналов во время кодирования, который [поток] затем может извлекаться устройством декодирования в любое время после сохранения на этот носитель.

В некоторых примерах видеокодер 20 и видеодекодер 30 работают в соответствии со стандартом сжатия видео, например упомянутым выше стандартом HEVC, расширениями HEVC или, возможно, следующим поколением стандартов кодирования видео, находящихся на стадии разработки. Только для простоты понимания нижеследующее предоставляет некоторую информацию о стандарте HEVC. Однако описанные в данном раскрытии изобретения методики не следует считать ограниченными стандартом

HEVC.

В HEVC и других стандартах кодирования видео видеопоследовательность обычно включает в себя последовательность изображений. Изображения также могут называться "кадрами". Изображение может включать в себя три массива выборок, обозначенные S_L, S_{Cb} и S_{Cr}. S_L является двумерным массивом (то есть блоком) выборок яркости. S_{Cb} является двумерным массивом выборок цветности C_b. S_{Cr} является двумерным массивом выборок цветности Cr. Выборки цветности в этом документе также могут называться выборками "цветности". В иных случаях изображение может быть монохромным и может включать в себя только массив выборок яркости.

Чтобы сформировать кодированное представление изображения, видеокодер 20 может сформировать набор единиц дерева кодирования (CTU). Каждая из CTU может быть блоком дерева кодирования из выборок яркости, двумя соответствующими блоками дерева кодирования из выборок цветности и синтаксическими структурами, используемыми для кодирования выборок блоков дерева кодирования. Блок дерева кодирования может быть блоком выборок N×N. CTU также может называться "блоком дерева" или "наибольшей единицей кодирования" (LCU). CTU в HEVC могут быть аналогичны в общих чертах макроблокам из других стандартов, например H.264/AVC. Однако CTU не обязательно ограничивается конкретным размером и может включать в себя одну или несколько единиц кодирования (CU). Секция может включать в себя целое число CTU, упорядоченных последовательно при растровом сканировании.

Чтобы сформировать кодированную CTU, видеокодер 20 может рекурсивно выполнить разбиение квадродерева над блоками дерева кодирования у CTU, чтобы разделить блоки дерева кодирования на блоки кодирования, отсюда название "единицы дерева кодирования". Блок кодирования является блоком выборок N×N. CU может быть блоком кодирования из выборок яркости и двумя соответствующими блоками кодирования из выборок цветности изображения, которое имеет массив выборок яркости, массив выборок Cb и массив выборок Cr, и синтаксическими структурами, используемыми для кодирования выборок блоков кодирования. Видеокодер 20 может разбить блок кодирования у CU на один или несколько блоков предсказания. Блок предсказания может быть прямоугольным (то есть квадратным или неквадратным) блоком выборок, к которому применяется одинаковое предсказание. Единица предсказания (PU) у CU может быть блоком предсказания из выборок яркости, двумя соответствующими блоками предсказания из выборок цветности изображения и синтаксическими структурами, используемыми для предсказания выборок блока предсказания. Видеокодер 20 может формировать блоки яркости, Cb и Cr с предсказанием для блоков предсказания яркости, Cb и Cr у каждой PU в CU.

Видеокодер 20 может использовать внутреннее предсказание или межкадровое предсказание для формирования (например, определения) блоков с предсказанием для PU. Если видеокодер 20 использует внутреннее предсказание для формирования блоков с предсказанием в PU, то видеокодер 20 может формировать блоки с предсказанием в PU на основе декодированных выборок изображения, ассоциированного с PU.

Если видеокодер 20 использует межкадровое предсказание для формирования (например, определения) блоков с предсказанием в PU, то видеокодер 20 может формировать блоки с предсказанием в PU на основе декодированных выборок одного или нескольких изображений помимо изображения, ассоциированного с PU. Видеокодер 20 может использовать однонаправленное предсказание или двунаправленное предсказание для формирования блоков с предсказанием в PU. Когда видеокодер 20 использует однонаправленное предсказание для формирования блоков с предсказанием для PU, PU может иметь один вектор движения (MV). Когда видеокодер 20 использует двунаправленное предсказание для формирования блоков с предсказанием для PU, PU может иметь два MV.

После того, как видеокодер 20 формирует блоки яркости, Cb и Cr с предсказанием для одной или нескольких PU в CU, видеокодер 20 может сформировать остаточный блок яркости для CU. Каждая выборка в остаточном блоке яркости у CU указывает разность между выборкой яркости в одном из блоков яркости с предсказанием у CU и соответствующей выборкой в исходном блоке кодирования яркости у CU. К тому же видеокодер 20 может формировать остаточный блок Cb для CU. Каждая выборка в остаточном блоке Cb у CU может указывать разность между выборкой Cb в одном из блоков Cb с предсказанием у CU и соответствующей выборкой в исходном блоке кодирования Cb у CU. Видеокодер 20 также может формировать остаточный блок Cr для CU. Каждая выборка в остаточном блоке Cr у CU может указывать разность между выборкой Cr в одном из блоков Cr с предсказанием у CU и соответствующей выборкой в исходном блоке кодирования Cr у CU.

Кроме того, видеокодер 20 может использовать разбиение квадродерева для разложения остаточных блоков яркости, Cb и Cr в CU на один или несколько блоков преобразований яркости, Cb и Cr. Блок преобразований может быть прямоугольным блоком выборок, к которому применяется одинаковое преобразование. Единица преобразования (TU) в CU может быть блоком преобразований из выборок яркости, двумя соответствующими блоками преобразований из выборок цветности и синтаксическими структурами, используемыми для преобразования выборок блока преобразований. Таким образом, каждая TU в CU может ассоциироваться с блоком преобразований яркости, блоком преобразований Cb и блоком

преобразований Cr. Блок преобразований яркости, ассоциированный с TU, может быть субблоком остаточного блока яркости у CU. Блок преобразований Cb может быть субблоком остаточного блока Cb у CU. Блок преобразований Cr может быть субблоком остаточного блока Cr у CU.

Видеокодер 20 может применить одно или несколько преобразований к блоку преобразований яркости у TU, чтобы сформировать блок коэффициентов яркости для TU. Блок коэффициентов может быть двумерным массивом коэффициентов преобразования. Коэффициент преобразования может быть скалярной величиной. Видеокодер 20 может применить одно или несколько преобразований к блоку преобразований Cb у TU, чтобы сформировать блок коэффициентов Cb для TU. Видеокодер 20 может применить одно или несколько преобразований к блоку преобразований Cr у TU, чтобы сформировать блок коэффициентов Cr для TU. Как описано подробнее, данное раскрытие изобретения описывает примерные способы, которыми видеокодер 20 определяет преобразования для использования в формировании блоков коэффициентов.

После формирования блока коэффициентов (например, блока коэффициентов яркости, блока коэффициентов Cb или блока коэффициентов Cr) видеокодер 20 может квантовать блок коэффициентов. Квантование в целом относится к процессу, в котором коэффициенты преобразования квантуются, чтобы уменьшить по возможности объем данных, используемый для представления коэффициентов преобразования, обеспечивая дополнительное сжатие. После того, как видеокодер 20 квантует блок коэффициентов, видеокодер 20 может энтропийно кодировать синтаксические элементы, указывающие квантованные коэффициенты преобразования. Например, видеокодер 20 может выполнить контекстно-адаптивное двоичное арифметическое кодирование (CABAC) над синтаксическими элементами, указывающими квантованные коэффициенты преобразования. Видеокодер 20 может вывести энтропийно кодированные синтаксические элементы в поток двоичных сигналов.

Видеокодер 20 может вывести поток двоичных сигналов, который включает в себя энтропийно кодированные синтаксические элементы. Поток двоичных сигналов может включать в себя последовательность разрядов, которая образует представление кодированных изображений, и ассоциированные данные. Поток двоичных сигналов может содержать последовательность единиц на уровне абстракции сети (NAL). Каждая из единиц NAL включает в себя заголовок единицы NAL и заключает в себя полезную нагрузку необработанной последовательности байтов (RBSP). Заголовок единицы NAL может включать в себя синтаксический элемент, который указывает код типа единицы NAL. Код типа единицы NAL, заданный заголовком единицы NAL у единицы NAL, указывает тип единицы NAL. RBSP может быть синтаксической структурой, содержащей целое число байтов, которая заключается в единицу NAL. В некоторых случаях RBSP включает в себя нулевые разряды.

Разные типы единиц NAL могут заключать в себя разные типы RBSP. Например, первый тип единицы NAL может заключать в себя RBSP для набора параметров изображения (PPS), второй тип единицы NAL может заключать в себя RBSP для кодированной секции, третий тип единицы NAL может заключать в себя RBSP для SEI, и так далее. Единицы NAL, которые заключают в себя RBSP для данных кодирования видео (в отличие от RBSP для наборов параметров и сообщений SEI), могут называться единицами NAL на уровне видеокодирования (VCL).

Видеодекодер 30 может принять поток двоичных сигналов, сформированный видеокодером 20. К тому же видеодекодер 30 может проанализировать поток двоичных сигналов, чтобы декодировать синтаксические элементы из этого потока двоичных сигналов. Видеодекодер 30 может восстановить изображения в видеоданных по меньшей мере частично на основе синтаксических элементов, декодированных из потока двоичных сигналов. Процесс для восстановления видеоданных в целом может быть обратным процессу, выполненному видеокодером 20. Например, видеодекодер 30 может использовать MV в PU для определения блоков с предсказанием для PU в текущей CU. К тому же видеодекодер 30 может обратно квантовать блоки коэффициентов преобразования, ассоциированные с TU в текущей CU.

Видеодекодер 30 может выполнить обратные преобразования над блоками коэффициентов преобразования, чтобы восстановить блоки преобразований, ассоциированные с TU в текущей CU. Данное раскрытие изобретения описывает примерные методики для способа, которым видеодекодер 30 определяет преобразования, которые используются для выполнения обратных преобразований над блоками коэффициентов преобразования.

Видеодекодер 30 может восстановить блоки кодирования в текущей CU путем сложения выборок блоков с предсказанием для PU в текущей CU с соответствующими выборками блоков преобразования у TU в текущей CU. Путем восстановления блоков кодирования для каждой CU изображения видеодекодер 30 может восстановить изображение.

Как описано выше, CU включает в себя одну или несколько TU. Нижеследующее описывает схему преобразования на основе остаточного квадродерева в HEVC. Чтобы приспособиться к различным характеристикам остаточных блоков, в HEVC применяется структура кодирования с преобразованием, использующая остаточное квадродерево (RQT) , которая кратко описывается в

http://www.hhi.fraunhofer.de/fields-of-competence/image-processing/research-groups/image-video-coding/hevc-high-efficiency-video-coding/transform-coding-using-the-residual-quadtree-rqt.html.

Как описано выше, каждое изображение разделяется на CTU, которые кодируются в порядке рас-

трового сканирования для определенного фрагмента или секции. CTU является квадратным блоком и представляет собой корень квадродерева, то есть дерева кодирования. Размер CTU может меняться от 8×8 до 64×64 выборок яркости, но обычно используется 64×64. Каждая CTU может дополнительно разбиваться на меньшие квадратные блоки, называемые единицами кодирования (CU). После того, как CTU рекурсивно разбивается на CU, каждая CU дополнительно разделяется на единицы предсказания (PU) и единицы преобразования (TU). Разбиение CU на TU осуществляется рекурсивно на основе подхода с квадродеревом, поэтому разностный сигнал каждой CU кодируется по древовидной структуре, а именно остаточному квадродереву (RQT). RQT допускает размеры TU от 4×4 до 32×32 выборок яркости.

Фиг. 3 показывает пример, где CU включает в себя 10 TU, обозначенных буквами от "a" до "j", и соответствующее разбиение блоков. Каждый узел RQT фактически является единицей преобразования (TU). Отдельные TU обрабатываются в порядке обхода дерева в глубину, который иллюстрируется на фиг. 3 в виде алфавитного порядка, который придерживается рекурсивного Z-сканирования с обходом в глубину. Подход с квадродеревом дает возможность адаптации преобразования к меняющимся пространственно-частотным характеристикам разностного сигнала. Как правило, большие размеры блоков преобразований, которые обладают большей пространственной поддержкой, обеспечивают лучшее разрешение по частоте. Однако меньшие размеры блоков преобразований, которые обладают меньшей пространственной поддержкой, обеспечивают лучшее пространственное разрешение. Компромисс между двумя разрешениями, пространственным и по частоте, выбирается с помощью решения о режиме кодера (например, видеокодером 20), например, на основе методики оптимизации искажения в зависимости от скорости. Методика оптимизации искажения в зависимости от скорости вычисляет взвешенную сумму разрядов кодирования и искажения от восстановления, то есть цену искажения в зависимости от скорости, для каждого режима кодирования (например, определенной структуры разбиения RQT), и выбирает режим кодирования с наименьшей ценой искажения в зависимости от скорости в качестве лучшего режима.

В RQT задаются три параметра: максимальная глубина дерева, минимальный разрешенный размер преобразования и максимальный разрешенный размер преобразования. Минимальный и максимальный размеры преобразования могут меняться в диапазоне от 4×4 до 32×32 выборок, что соответствует поддерживаемым преобразованиям блоков, упомянутым в предыдущем абзаце. Максимальная разрешенная глубина RQT ограничивает количество TU. Равная нулю максимальная глубина означает, что CB (блок кодирования) нельзя больше разбивать, если каждый включенный TB (блок преобразований) достигает максимального разрешенного размера преобразования, например 32×32.

Все эти параметры взаимодействуют и влияют на структуру RQT. Рассмотрим случай, в котором размер корневого CB равен 64×64, максимальная глубина равна нулю, а максимальный размер преобразования равен 32×32. В этом случае CB нужно разбить по меньшей мере один раз, поскольку в противном случае это привело бы к TB 64×64, что не разрешено. В HEVC преобразования большего размера, например, преобразования 64×64, не приняты преимущественно из-за их ограниченной пользы в общем и относительно высокой сложности для видео с относительно меньшим разрешением.

Параметры RQT, то есть максимальная глубина RQT, минимальный и максимальный размер преобразования, передаются в потоке двоичных сигналов на уровне набора параметров последовательности. Что касается глубины RQT, разные значения могут задаваться и сигнализироваться для CU с внутренним и межкадровым кодированием (то есть кодированных CU с внутренним предсказанием или декодированных CU с межкадровым предсказанием либо кодированных CU с внутренним предсказанием или CU с межкадровым предсказанием).

Преобразование квадродерева применяется для внутренних и внешних остаточных блоков. Как правило, для остаточного блока применяется преобразование DCT-II с одинаковым размером разбиения текущего остаточного квадродерева. Однако, если блоком текущего остаточного квадродерева является 4×4, и он формируется с помощью внутреннего предсказания, то применяется вышеупомянутое преобразование DST-VII 4×4.

Нижеследующее описывает кодирование коэффициентов в HEVC. Независимо от размера TU остаток единицы преобразования кодируется с помощью неперекрывающихся групп коэффициентов (CG), и каждая содержит коэффициенты блока 4×4 в TU. Например, TU 32×32 содержит в целом 64 CG, а TU 16×16 содержит в целом 16 CG. CG внутри TU кодируются в соответствии с некоторым предопределенным порядком сканирования. При кодировании каждой CG коэффициенты внутри текущей CG сканируются и кодируются в соответствии с некоторым предопределенным порядком сканирования для блока 4×4. Фиг. 4 иллюстрирует сканирование коэффициентов для TU 8×8, содержащей 4 CG.

Для каждой цветовой компоненты может прежде всего сигнализироваться один признак для указания, содержит ли текущая единица преобразования по меньшей мере один ненулевой коэффициент. Если есть по меньшей мере один ненулевой коэффициент, то положение последнего значимого коэффициента в порядке сканирования коэффициентов в единице преобразования явно кодируется с координацией относительно верхнего левого угла единицы преобразования. Вертикальная или горизонтальная компонента координации представляется префиксом и суффиксом, где префикс преобразуется в двоичную форму

по усеченному коду Райса (TR), а суффикс преобразуется в двоичную форму с фиксированной длиной.

last_sig_coeff_x_prefix задает префикс положения столбца последнего значимого коэффициента в порядке сканирования в блоке преобразований. Значения last_sig_coeff_x_prefix должны быть в диапазоне от 0 до (log2TrafoSize <<1) - 1 включительно.

last sig coeff y prefix задает префикс положения строки последнего значимого коэффициента в порядке сканирования в блоке преобразований. Значения last_sig_coeff_y_prefix должны быть в диапазоне от 0 до (log2TrafoSize <<1) - 1 включительно.

last_sig_coeff_x_suffix задает суффикс положения столбца последнего значимого коэффициента в порядке сканирования в блоке преобразований. Значения last_sig_coeff_x_suffix должны быть в диапазоне от 0 до (1 <<((last_sig_coeff_x_prefix >>1)-1))-1 включительно.

Положение столбца последнего значимого коэффициента в порядке сканирования в блоке преобразований, LastSignificantCoeffX, выводится следующим образом:

Если last_sig_coeff_x suffix отсутствует, то применяется следующее:

LastSignificantCoeffX=last_sig_coeff_x_prefix

В противном случае (last_sig_coeff_x_suffix присутствует) применяется следующее:

LastSignificantCoeffX=(1<<((last_sig_coeff_x_prefix >>1)-1))*

(2+(last_sig_coeff_x_prefix & 1))+last_sig_coeff_x suffix

last_sig_coeff_y_suffix задает суффикс положения строки последнего значимого коэффициента в порядке сканирования в блоке преобразований. Значения last sig coeff y suffix должны быть в диапазоне от 0 до (1<<((last_sig_coeff_y_prefix >>1)-1))-1 включительно.

Положение строки последнего значимого коэффициента в порядке сканирования в блоке преобразований, LastSignificantCoeffY, выводится следующим образом:

Если last sig coeff y suffix отсутствует, то применяется следующее:

LastSignificantCoeffY=last_sig_coeff_yprefix

В противном случае (last_sig_coeff_y_suffix присутствует) применяется следующее:

LastSignificantCoeffY=(1<<((last_sig_coeff_y_prefix>>1)-1))*

(2+(last_sig_coeff_y_prefix & 1))+last_sig_coeff_y_suffix

Когда scanIdx равен 2, координаты меняются следующим образом:

(LastSignificantCoeffX, LastSignificantCoeffY)=Swap(LastSignificantCoeffX, LastSignificantCoeffY)

При таком кодированном положении, а также порядке сканирования коэффициентов CG дополнительно сигнализируется один признак для CG за исключением последней CG (в порядке сканирования), который указывает, содержит ли группа ненулевые коэффициенты. Для тех CG, которые могут содержать ненулевые коэффициенты, могут дополнительно кодироваться значимые признаки, абсолютные значения коэффициентов и знаковая информация для каждого коэффициента в соответствии с предопределенным порядком сканирования коэффициентов 4×4.

Как описано выше, описанные в данном раскрытии изобретения методики описывают способы определения преобразования, которое видеокодер 20 применяет для преобразования блока преобразований в блок коэффициентов, и способы определения преобразования, которое видеодекодер 30 применяет (например, в качестве обратного преобразования) для преобразования блока коэффициентов в блок преобразований.

Нижеследующее описывает множественное преобразование для остатка внутреннего и межкадрового предсказания (например, разные типы преобразований, когда остаточный блок формируется из внутреннего предсказания, и когда остаточный блок формируется из межкадрового предсказания).

В некоторых случаях несмотря на то, что Тип-VII DST может эффективно повысить эффективность внутреннего кодирования по сравнению с традиционным Типом-II DCT, эффективность преобразования сравнительно ограничена, потому что остатки предсказания показывают различную статистику, и постоянное использование Типа-II DCT и Типа-VII DST не может эффективно приспособиться ко всем возможным случаям. Предложены некоторые методики для приспособления к разным случаям.

В S.-C. Lim, D.-Y. Kim, S. Jeong, J. S. Choi, H. Choi, and Y.-L. Lee, "Rate-distortion optimized adaptive transform coding", Opt. Eng., vol. 48, no. 8, pp. 087004-1-087004-14, Aug. 2009 предлагается новая схема преобразования, которая адаптивно применяет целочисленную версию DCT или DST для остатка предсказания, и для каждого блока сигнализируется, используется ли преобразование DCT или DST для остатка предсказания. В Y. Ye and M. Karczewicz, "Improved H.264 intra coding based on bidirectional intra prediction, directional transform, and adaptive coefficient scanning", in Proc. 15th IEEE Int. Conf. Image Process., Oct. 2008, pp. 2116-2119 предложено, что каждый режим внутреннего предсказания может отображаться в уникальную пару преобразования (C и R), предопределенную как пара KLT, чтобы применялось зависимое от режима преобразование (MDDT). Таким образом, разные преобразования KLT могут использоваться для разных режимов внутреннего предсказания; однако то, какое преобразование нужно использовать, предопределено и зависит от режима внутреннего предсказания.

Однако в X. Zhao, L. Zhang, S. W. Ma, and W. Gao, "Video coding with rate-distortion optimized transform", IEEE Trans. Circuits Syst. Video Technol., vol. 22, no. 1, pp. 138-151, Jan. 2012 может использоваться больше преобразований, и явно сигнализируется индекс к преобразованиям из предопределенного набо-

ра вариантов преобразований, которые выводятся из автономного процесса обучения. Аналогично MDDT, каждое направление внутреннего предсказания может обладать своим уникальным набором пар преобразований. Индекс сигнализируется для задания, какая пара преобразований выбирается из набора. Например, имеется вплоть до четырех вертикальных преобразований KLT и вплоть до четырех горизонтальных преобразований KLT для наименьших размеров блоков 4×4; поэтому можно выбрать 16 сочетаний. Для больших размеров блоков используется меньшее количество сочетаний. Предложенный способ в "Video coding with rate-distortion optimized transform" применяется к остатку внутреннего и межкадрового предсказания. Для остатка межкадрового предсказания можно выбрать вплоть до 16 сочетаний преобразований KLT, и для каждого блока сигнализируется индекс к одному из сочетаний (четыре для 4×4 и шестнадцать для 8×8).

В A. Saxena and F. Fernandes, "DCT/DST-based transform coding for intra prediction in image/video coding", IEEE Trans. Image Processing, и C. Yeo, Y. H. Tan, Z. Li, and S. Rahardja, "Mode-dependent transforms for coding directional intra prediction residuals", IEEE Trans. Circuits Syst. Video Technol., vol. 22, no. 4, pp. 545-554, 2012, используются множественные преобразования; однако вместо использования преобразований KLT (которые обычно нужно обучать), используется либо DCT (DCT-II), либо DST (DST-VII) для единицы преобразования (при этом левое и правое преобразования (например, C и R) одинаковы), и сигнализируемым признаком определяется, какое преобразование нужно использовать. В F. Zou, O. C. Au, C. Pang, J. Dai, and F. Lu, "Rate-Distortion Optimized Transforms Based on the Lloyd-Type Algorithm for Intra Block Coding", IEEE Journal of Selected Topics in Signal Processing, Volume:7, Issue: 6, Nov. 2013, используется несколько предопределенных пар преобразований KLT, и для единицы кодирования сигнализируется индекс к паре преобразований (вместо его выведения), чтобы каждая единица преобразования в единице кодирования использовала одинаковую пару преобразований.

В J. An, X. Zhao, X. Guo and S. Lei, "Non-CE7: Boundary-Dependent Transform for Inter-Predicted Residue", JCTVC-G281 множественные преобразования выбираются для внешне предсказанного остатка в TU в соответствии с их местоположениями в CU. Преобразования C и R выбираются из DST-VII и "перевернутой" версии DST-VII. Поэтому возможны вплоть до четырех сочетаний для TU в CU. Однако, поскольку сочетание полностью определяется местоположением PU, не нужно сигнализировать, какое сочетание используется.

С методиками, связанными с преобразованиями, могут быть некоторые проблемы для остатков (например, проблемы для остатков с внутренним предсказанием, которые получаются из внутреннего предсказания, но с тем же успехом могут применяться к остаткам с межкадровым предсказанием, которые получаются из межкадрового предсказания). Существующие способы могут использовать пару из преобразований DST или DCT для остатка с внутренним предсказанием. Однако те преобразования не могут охватить все возможные распределения разностного сигнала.

Например, в HEVC для остаточных блоков внутреннего предсказания, больше либо равных 8×8, применяется только Тип-II DCT, что нельзя приспособить к меняющейся статистике остатка внутреннего предсказания. В HEVC для остатка межкадрового предсказания применяется только Тип-II DCT, что нельзя приспособить к меняющейся статистике остатка межкадрового предсказания. Простой выбор преобразования в зависимости от размера блока преобразований или режимов внутреннего предсказания не очень эффективен, потому что статистика остатков по-прежнему может обладать большим разбросом даже при одном и том же режиме внутреннего предсказания или одном и том же размере преобразования.

Данное раскрытие изобретения описывает следующие методики. В некоторых примерах одна или несколько следующих методик могут решать одну или несколько вышеупомянутых проблем. Однако не является требованием, что следующие методики решают одну или несколько вышеупомянутых проблем. Нижеследующие методики могут применяться по отдельности. В некоторых случаях может применяться любое сочетание примерных методик. Например, видеокодер 20 и видеодекодер 30 могут применять методики по отдельности или применять в некоторых случаях любое сочетание из одной или нескольких методик.

В некоторых примерах в дополнение к основанному на DCT-II преобразованию, используемому в HEVC, для каждого остаточного блока, сформированного режимом внутреннего предсказания, видеокодер 20 и видеодекодер 30 могут выбирать преобразования из двух или более возможных преобразований из семейств DCT и DST. В качестве одного примера возможные преобразования могут принадлежать суммарно 16 преобразованиям на основе разных типов семейств DCT и DST и могут включать в себя, но не ограничиваются, DCT-I ~ DCT-VIII, DST-I ~ DST-VIII. В качестве альтернативы или в дополнение видеокодер 20 и видеодекодер 30 могут использовать другие синусоидальные унитарные преобразования, или даже могут использоваться другие преобразования KLT. Для каждой TU горизонтальное и вертикальное преобразования (например, правое и левое преобразования) могут быть одинакового типа. Например, возможными преобразованиями являются DST-VII, DCT-VIII, DST-I и DST-V.

Как описано выше, существует 16 преобразований (например, DCT-I - DCT-VIII и DST-I - DST-VIII). Одним из способов идентифицировать, какое преобразование использовать, является создание ви-

деокодером 20 и видеодекодером 30 списка этих 16 преобразований. Тогда видеокодер 20 может сигнализировать (например, формировать в потоке двоичных сигналов) первый индекс к списку, чтобы идентифицировать левое преобразование (например, преобразование C для уравнения $Y=C*X*R^T$, где X - блок преобразований, а Y - результирующий блок коэффициентов), и сигнализировать (например, формировать в потоке двоичных сигналов) второй индекс к списку, чтобы идентифицировать правое преобразование (например, преобразование R для уравнения $Y=C*X*R^T$). Тогда видеодекодер 30 принял бы первый индекс и второй индекс из потока двоичных сигналов и определил преобразования C и R, которые видеодекодер 30 должен использовать для обратного преобразования блока коэффициентов обратно в блок преобразований.

В этом примере значение первого индекса может меняться от 0 до 15, и значение второго индекса может меняться от 0 до 15. Вообще, кодирование больших чисел требует сигнализации большего количества разрядов, нежели кодирование меньших чисел (например, указание значения 15 индекса требует больше разрядов, чем указание значения 2 индекса). В случае, где список включает в себя все 16 преобразований, может быть служебная нагрузка сигнализации, которая потребляет больше полосы пропускания, чем желательно. Однако ограничение возможных вариантов того, какие преобразования можно использовать, как сделано в HEVC, может уменьшить служебную нагрузку сигнализации, но отрицательно повлиять на эффективность кодирования, так как лучшие преобразования не используются.

В методиках, описанных в данном раскрытии изобретения, видеокодер 20 и видеодекодер 30 могут уметь определять левое и правое преобразование из довольно большого количества возможных преобразований со слабым влиянием на служебную нагрузку сигнализации. В качестве одного примера видеокодер 20 и видеодекодер 30 могут определять множество поднаборов преобразований, где каждое поднабор преобразований идентифицирует множество возможных преобразований.

Например, видеокодер 20 и видеодекодер 30 могут создать три следующих поднаборы преобразований и сохранить эти поднаборы преобразований в запоминающем устройстве: поднабор 0 преобразований: {DST-VII, DCT-VIII}, поднабор 1 преобразований: {DST-VII, DST-I} и поднабор 2 преобразований: {DST-VII, DCT-V}. В некоторых примерах эти три преобразования могут заранее сохраняться в запоминающем устройстве видеокодера 20 и видеодекодера 30. В любом случае видеокодер 20 и видеодекодер 30 могут считаться определяющими три этих поднаборы преобразований, где каждое из трех поднаборов преобразований идентифицирует множество возможных преобразований (например, два преобразования в этом примере). Множество поднаборов преобразований может включать в себя больше или меньше трех поднаборов преобразований и, как правило, включает в себя два или более поднаборов преобразований. Каждое поднабор преобразований может включать в себя одно или несколько возможных преобразований, но по меньшей мере одно идентифицирует множество возможных преобразований. Например, некоторые поднаборы преобразований могут идентифицировать только одно преобразование, а другие могут идентифицировать два или более преобразований. В некоторых примерах каждое поднабор преобразований может идентифицировать сравнительно небольшое количество преобразований (например, меньше либо равное 5).

В методиках, описанных в данном раскрытии изобретения, видеокодер 20 и видеодекодер 30 могут определять соответствующие поднаборы преобразований. Например, если сохраненными поднаборами преобразований в видеокодере 20 являются поднабор 0 преобразований: {DST-VII, DCT-VIII}, поднабор 1 преобразований: {DST-VII, DST-I} и поднабор 2 преобразований: {DST-VII, DCT-V}, то видеодекодер 30 может хранить поднаборы обратных преобразований: поднабор 0 обратных преобразований: {IDST-VII, IDCT-VIII}, поднабор 1 обратных преобразований: {IDST-VII, IDST-I} и поднабор 2 обратных преобразований: {IDST-VII, IDCT-V}. В качестве другого примера видеодекодер 30 может хранить те же преобразования, что и видеокодер 20, и может инвертировать их перед применением обратного преобразования. В любом примере видеокодер 20 и видеодекодер 30 могут считаться хранящими соответствующие поднаборы преобразований (например, одинаковые поднаборы или поднаборы, имеющие обратные друг другу преобразования).

Видеокодер 20 и видеодекодер 30 могут использовать неявные методики для выбора поднаборов преобразований для левого и правого преобразований. Неявные методики означают, что видеокодеру 20 не нужно сигнализировать видеодекодеру 30 информацию, указывающую видеодекодеру 30, какие поднаборы преобразований выбирать. Видеокодер 20 и видеодекодер 30 могут конфигурироваться для выполнения одной и той же неявной методики для выбора поднаборов преобразований, что приводит к видеокодеру 20 и видеодекодеру 30, выбирающим одинаковые поднаборы преобразований без какого-либо увеличения количества информации, которую нужно сигнализировать.

В качестве одного примера, если блок преобразований формируется из внутреннего предсказания, то видеокодер 20 и видеодекодер 30 могут определить, какие поднаборы преобразований выбирать, на основе режима внутреннего предсказания. Например, видеокодер 20 и видеодекодер 30 могут хранить таблицу, которая отображает режим внутреннего предсказания в поднабор преобразований, из которого нужно определить левое преобразование, и в поднабор преобразований, из которого нужно определить правое преобразование.

В качестве примера видеокодер 20 может кодировать с внутренним предсказанием текущий блок в

режиме X внутреннего предсказания. В этом примере видеокодер 20 формирует блок преобразований из остаточного блока, сформированного из кодирования с внутренним предсказанием текущего блока в режиме X внутреннего предсказания. Видеокодер 20 может выбрать поднабор преобразований для левого преобразования на основе режима X внутреннего предсказания и выбрать поднабор преобразований для правого преобразования на основе режима X внутреннего предсказания. Видеокодер 20 может определить левое и правое преобразования из соответствующих выбранных поднаборов преобразований, как подробнее описано ниже, и применить преобразования для формирования блока коэффициентов.

Видеокодер 20 может сформировать поток двоичных сигналов видео, который включает в себя информацию, указывающую значения коэффициентов из блока коэффициентов, а также информацию, указывающую, что блок преобразований, который формируется из блока коэффициентов, предназначен для блока, который кодировался с внутренним предсказанием с использованием режима X внутреннего предсказания. Видеодекодер 30 может сформировать блок коэффициентов из сигнализированной информации и определить, что режимом внутреннего предсказания был режим X, также из сигнализированной информации. Видеодекодер 30 может выбрать поднабор преобразований для левого преобразования (которое в этом случае будет обратным преобразованию, применяемому видеокодером 20) и поднабор преобразований для правого преобразования (которое в этом случае будет обратным преобразованию, применяемому видеокодером 20) на основе режима внутреннего предсказания, являющегося режимом X.

Сохраненное отображение, указывающее, какие поднаборы преобразований отображаются в какой режим внутреннего предсказания, одинаково на стороне видеокодера 20 и стороне видеодекодера 30. Поэтому видеокодер 20 и видеодекодер 30 выбирают соответствующие поднаборы преобразований. Видеодекодер 30 может определить левое и правое преобразования из соответствующих выбранных поднаборов преобразований, как подробнее описано ниже, и применить преобразования для формирования блока преобразований.

Хотя вышеприведенный пример описывается относительно режимов внутреннего предсказания, описанные в данном раскрытии изобретения методики этим не ограничены. В некоторых примерах вместо режимов внутреннего предсказания видеокодер 20 и видеодекодер 30 могут выбирать соответствующие поднаборы преобразований на основе другой информации, например глубины RQT, квантованных коэффициентов и т.п.

Также, хотя вышеприведенный пример описывается для внутреннего предсказания, описанные в данном раскрытии изобретения методики с тем же успехом можно распространить на межкадровое предсказание. Например, аналогично вышеприведенному, видеокодер 20 и видеодекодер 30 могут определить множество поднаборов преобразований. Это множество поднаборов преобразований для случая межкадрового предсказания может быть таким же или отличным от множества поднаборов преобразований для случая внутреннего предсказания. В некоторых случаях множество поднаборов преобразований для случая межкадрового предсказания может быть таким же, как некоторые, но не все из множества поднаборов преобразований для случая внутреннего предсказания.

Для межкадрового предсказания видеокодер 20 и видеодекодер 30 могут хранить отображение между положением блока преобразований и PU, CU или LCU, с которой он ассоциируется.

Например, отображение может указывать, что если блок преобразований находится на левой границе PU, CU или LCU, то выбирается первая группа поднаборов преобразований (например, один поднабор преобразований для левого преобразования и один поднабор преобразований для правого преобразования). Если блок преобразований находится на правой границе PU, CU или LCU, то выбирается вторая группа поднаборов преобразований, и так далее для верхней и нижней границ, где в каждом случае видеокодер 20 и видеодекодер 30 выбирают один поднабор преобразований для левого преобразования и один поднабор преобразований для правого преобразования.

Видеокодер 20 и видеодекодер 30 могут кодировать и декодировать блоки изображения в конкретном порядке. Соответственно, на основе местоположения только что кодированного или декодированного блока видеокодер 20 и видеодекодер 30 могут определять местоположение блока преобразований в PU, CU или LCU. Опять с точки зрения видеодекодера 30, видеодекодер 30 формирует блок преобразований из блока коэффициентов. Однако на основе порядка декодирования видеодекодер 30 может уметь определять местоположение блока преобразований, который нужно сформировать из блока коэффициентов.

Таким образом, видеокодер 20 и видеодекодер 30 могут определять соответствующие поднаборы преобразований, из которых нужно определять левое преобразование и правое преобразование, без какого-либо увеличения количества информации, которую нужно сигнализировать. В некоторых примерах после того, как видеокодер 20 выбирает поднаборы преобразований, видеокодер 20 может сигнализировать информацию (например, сформировать информацию в потоке двоичных сигналов видео), указывающую, какое преобразование в выбранных поднаборах преобразований предназначено для левого преобразования, а какое преобразование предназначено для правого преобразования. Видеодекодер 30 принимает сигнализированную информацию и определяет левое и правое преобразования.

Например, видеокодер 20 может сигнализировать(например, сформировать в потоке двоичных сиг-

налов) индекс к поднабору преобразований, выбранному для левого преобразования, и сигнализировать (например, сформировать в потоке двоичных сигналов) индекс к поднабору преобразований, выбранному для правого преобразования. Видеодекодер 30 может принять соответствующие индексы к соответствующим поднаборам преобразований и определить левое и правое преобразование.

В этом примере может быть увеличение информации, которую нужно сигнализировать (например, сигнализируются индексы для определения левого и правого преобразований). Однако увеличение информации, которую нужно сигнализировать, может быть минимальным. Как описано выше, каждое из поднаборов преобразований может идентифицировать сравнительно небольшое количество преобразований. Поэтому диапазон значения индекса может быть относительно небольшим (например, от 0 до 4, если максимальное количество преобразований, которое идентифицирует каждое поднабор преобразований, равно 5).

Соответственно, для сравнительно небольшого увеличения служебной нагрузки сигнализации методики, описанные в данном раскрытии изобретения, допускают довольно большое увеличение количества преобразований, которые могут выбираться. Например, поскольку имеется множество поднаборов преобразований, причем каждое включает в себя одно или несколько преобразований, можно идентифицировать многие и, возможно, все из 16 примерных преобразований в одном или нескольких преобразованиях. Поскольку поднаборы преобразований выбираются с помощью неявных методик, отсутствует увеличение служебной нагрузки сигнализации, а поскольку каждое поднабор преобразований идентифицирует сравнительно небольшое количество преобразований, идентификация конкретного преобразования не сильно увеличивает служебную нагрузку сигнализации.

В некоторых примерах можно дополнительно уменьшить величину служебной нагрузки сигнализации. Например, в некоторых примерах видеокодер 20 и видеодекодер 30 могут выбирать поднаборы преобразований, как описано выше, но затем конфигурироваться для определения конкретного преобразования из каждого из соответствующих поднаборов преобразований на основе некоторых условий. В этом случае видеокодеру 20 может не требоваться сигнализировать, а видеодекодеру 30 может не требоваться принимать информацию, указывающую, какое преобразование использовать в выбранных поднаборах преобразований.

В качестве примера во время процесса кодирования видеокодер 20 может использовать конкретное преобразование из выбранного поднаборы преобразований (например, первое идентифицированное преобразование в выбранном поднаборе преобразований) и после того, как применяется преобразование, определить, что количество ненулевых коэффициентов в результирующем блоке коэффициентов меньше пороговой величины. В этом случае видеодекодер 30 может принять информацию, указывающую значения коэффициентов в блоке коэффициентов, и аналогичным образом определить, что количество ненулевых коэффициентов меньше пороговой величины. В некоторых примерах, если видеодекодер 30 определяет, что количество ненулевых коэффициентов в блоке коэффициентов меньше пороговой величины (например, 1 или 2), то видеодекодер 30 может определить, что видеодекодер 30 должен использовать конкретное преобразование из выбранного поднаборы преобразований (например, первое идентифицированное преобразование в выбранном поднаборе преобразований).

Например, предположим, что на основе режима внутреннего предсказания видеокодер 20 определил, что поднабором преобразований для левого преобразования является поднабор 0, а для правого преобразования - поднабор 1. В этом случае видеокодер 20 может определить, что если первое идентифицированное преобразование в поднаборе 0 используется в качестве левого преобразования, и если первое идентифицированное преобразование в поднаборе 1 используется в качестве правого преобразования, то количество ненулевых коэффициентов в результирующем блоке коэффициентов меньше порогового значения. В этом примере видеокодер 20 может не сигнализировать информацию, указывающую, что первое идентифицированное преобразование в поднаборе 0 и поднаборе 1 нужно использовать в качестве левого и правого преобразований соответственно. В иных случаях, если первое идентифицированное преобразование в поднаборе 0 (или поднаборе 1) не используется в качестве левого преобразования (или правого преобразования), то количество ненулевых коэффициентов в результирующем блоке коэффициентов меньше порогового значения. В этом примере видеокодер 20 добавляет ограничение, что идентифицированные преобразования в поднаборе 0 и поднаборе 1 нельзя использовать в качестве левого и правого преобразования.

Видеодекодер 30 может принять режим внутреннего предсказания и, как и видеокодер 20, определить на основе режима внутреннего предсказания, что поднабор 0 преобразований и поднабор 1 преобразований нужно выбрать для левого и правого преобразований соответственно. Также после формирования блока коэффициентов из информации, указывающей значения коэффициентов, видеодекодер 30 также может определить, что количество ненулевых коэффициентов в блоке коэффициентов меньше пороговой величины. Видеодекодер 30 может определить, что первое идентифицированное преобразование в поднаборе 0 и первое идентифицированное преобразование в поднаборе 1 нужно использовать в качестве левого и правого преобразований соответственно, без приема этой информации от видеокодера 20, потому что количество ненулевых коэффициентов меньше пороговой величины.

В вышеприведенных примерах поднаборы преобразований образуются из 16 преобразований (то

есть восьми DCT и восьми DST). Однако описанные в данном раскрытии изобретения методики этим не ограничиваются. Дополнительные примеры преобразований включают в себя преобразования KLT. Соответственно, поднаборы преобразований могут включать в себя одно или несколько преобразований из восьми DCT, восьми DST, преобразований KLT и других примеров преобразований. Исключительно для простоты описания примеры описываются по отношению к восьми DCT и восьми DST.

В качестве резюме в некоторых из примеров, описанных в данном раскрытии изобретения, выполняется предварительный выбор из трех или более возможных преобразований, чтобы сформулировать поднабор преобразований, и окончательное преобразование для использования для текущей TU выбирается из этого поднаборы преобразований. Например, поднабор преобразований может составлять поднабор левых преобразований и/или поднабор правых преобразований. Предварительный выбор для формулирования поднаборы преобразований (или поднаборы левых преобразований и поднаборы правых преобразований) может определяться уже декодированной информацией, например режимами внутреннего предсказания, глубиной RQT, квантованными коэффициентами и т.п.

Количество поднаборов преобразования может ограничиваться небольшим целым числом, например 1, 2, 3 или 4, и разные поднаборы преобразования содержат разный тип преобразований. В одном примере создаются три поднаборы преобразований, при этом каждое содержит два преобразования. На основе заданного режима внутреннего предсказания поднабор левых преобразований принимается за одно из трех поднаборов, и поднабор правых преобразований также принимается за одно из трех поднаборов (может быть или не быть таким же, как поднабор левых преобразований). В качестве примера тремя поднаборами преобразований являются: {DST-VII, DCT-VIII}, {DST-VII, DST-I} и {DST-VII, DCT-V}. Одним из вышеупомянутых трех поднаборов может быть либо поднабор левых преобразований, либо поднабор правых преобразований. Поэтому различные режимы внутреннего предсказания могут соответствовать вплоть до 9 разным сочетаниям поднаборов для левого и правого преобразований. В качестве альтернативы или дополнительно поднабор левых преобразований или поднабор правых преобразований содержит только одно преобразование. В качестве альтернативы или дополнительно поднабор левых преобразований и поднабор правых преобразований могут содержать только одно преобразование.

В описанных выше примерах поднаборы преобразований и преобразования, идентифицированные в поднаборах преобразований, могут быть одинаковыми независимо от размера TU, и количество преобразований в поднаборах преобразований может быть одинаковым для разных режимов внутреннего предсказания. Однако описанные в данном раскрытии изобретения методики этим не ограничиваются.

В некоторых примерах для разных размеров TU количество преобразований в поднаборе левых/правых преобразований может быть разным, типичное количество может быть равно 2, 3 и 4, но не только. Для разных режимов внутреннего предсказания количество преобразований в поднаборе левых/правых преобразований может быть разным; типичное количество преобразований может быть равно 2, 3 и 4, но не ограничивается этим.

Как описано выше, когда предварительно выбрано поднабор преобразований, окончательное преобразование для использования может сигнализироваться с помощью индекса к поднабору преобразований. Когда поднабор левых преобразований (или поднабор правых преобразований) содержит два или более преобразований, сигнализируется индекс для преобразования, принадлежащего поднабору левых преобразований (или поднабору правых преобразований). Это означает, что когда количество в поднаборе левых или правых преобразований равно 1, не нужно сигнализировать индекс преобразований.

Вышеприведенные примеры описывали случай, где видеокодер 20 и видеодекодер 30 могут предварительно выбирать для формулирования поднаборов преобразований. Однако описанные в данном раскрытии изобретения примеры этим не ограничиваются. В качестве альтернативы или дополнительно может быть не нужно выполнять предварительный выбор для формулирования поднаборов преобразований, и напрямую сигнализируется один индекс к двум или более возможным преобразованиям (в качестве полного набора) для указания левого или правого преобразования. Например, в видеокодере 20 можно ввести ограничение, что можно проверять только некоторые из преобразований в полном наборе, а другие преобразования не проверяются для уменьшения сложности кодера. То, какие преобразования выбирать, и индексы преобразований могут зависеть от режима внутреннего предсказания или другой информации.

В некоторых примерах для каждой TU может быть ограничено, что для левого преобразования (правого преобразования) видеокодер 20 и видеодекодер 30 могут выбирать левое и правое преобразование из поднаборы возможных преобразований. Например, только один поднабор преобразований содержит DST-VII, DCT-VIII и DCT-II, и левое преобразование для каждой TU всегда выбирается из {DST-VII, DCT-VIII и DCT-II}, а правое преобразование для каждой TU также всегда выбирается из {DST-VII, DCT-VIII и DCT-II}.

Как описано выше, примерные методики, описанные в данном раскрытии изобретения, могут применяться к внутреннему предсказанию и межкадровому предсказанию. В HEVC для блока преобразований, сформированного из межкадрового предсказания, было доступно только основанное на DCT-II преобразование. В некоторых примерах в дополнение к традиционному, основанному на DCT-II преобразованию, как в HEVC, для каждого остаточного блока, сформированного режимом межкадрового предска-

зания, видеокодер 20 и видеодекодер 30 могут выбирать преобразования из двух или более возможных способов преобразований из семейств DCT и DST или иных преобразований, например KLT, в дополнение к тому, что создаются поднабор левых преобразований и поднабор правых преобразований. Аналогично вышеприведенному примеру для внутреннего предсказания видеокодер 20 может сигнализировать (например, формировать в потоке двоичных сигналов), а видеодекодер 30 может принимать в потоке двоичных сигналов индекс к поднабору левых преобразований и индекс к поднабору правых преобразований для каждой TU, чтобы определить левое и правое преобразования.

В качестве одного примера два преобразования, например DST-VII и DCT-VIII, помещаются в поднабор левых преобразований и поднабор правых преобразований. Одноразрядный индекс к каждому из этих поднаборов определяет окончательные левое и правое преобразования текущей TU. Поднаборы могут быть либо {DST-VII, DCT-VIII}, либо {DST-VIII, DCT-VII}.

В качестве альтернативы или дополнительно выполняется предварительный выбор из трех или более возможных преобразований для формулирования поднаборы преобразований, и окончательное преобразование для использования для текущей TU выбирается из того поднаборы преобразований. Например, предварительный выбор для формулирования поднаборы преобразований (или поднаборы левых преобразований и поднаборы правых преобразований) может определяться взаимным расположением текущей TU и принадлежащей PU, то есть, располагается ли текущая TU на верхней границе, левой границе, правой границе, нижней границе или в другом положении принадлежащей PU.

В одном примере создаются три поднаборы преобразований, при этом каждое содержит два преобразования. На основе взаимного расположения текущей TU и принадлежащей PU поднабор левых преобразований принимается за одно из трех поднаборов, и поднабор правых преобразований также принимается за одно из трех поднаборов (может быть или не быть таким же, как поднабор левых преобразований). В качестве альтернативы или дополнительно поднабор левых преобразований или поднабор правых преобразований содержит только одно преобразование. В качестве альтернативы или дополнительно поднабор левых преобразований и поднабор правых преобразований могут содержать только одно преобразование.

В вышеприведенных примерах видеокодер 20 и видеодекодер 30 могут выбирать поднаборы преобразований для каждой TU в CU, а затем определять левое и правое преобразования для каждой TU, как описано выше. В этом примере определение того, какие преобразования использовать, считается относящимся к уровню TU. Однако описанные в данном раскрытии изобретения примерные методики этим не ограничиваются.

В некоторых случаях видеокодер 20 может определять, что левое и правое преобразования для каждой TU в CU должны быть одинаковым преобразованием по умолчанию (например, DCT-II в качестве одного примера, но с тем же успехом возможны другие типы преобразований). Также может быть преобразование по умолчанию для левого преобразования и преобразование по умолчанию для правого преобразования, либо преобразование по умолчанию для левого преобразования и правого преобразования может быть одинаковым. В нижеследующем описании термин "преобразование по умолчанию" следует интерпретировать включающим в себя случай, где преобразование по умолчанию для левого и правого преобразований отличается, и где преобразование по умолчанию для левого и правого преобразований одинаково. Например, преобразование по умолчанию для левого и правого преобразования (например, где отличается или одинаковое) может предварительно выбираться и быть одинаковым для видеокодера 20 и видеодекодера 30.

Если видеокодер 20 определяет, что каждая TU в CU должна иметь одинаковое преобразование по умолчанию, то видеокодер 20 может сигнализировать информацию, указывающую это как таковое (например, формировать в потоке двоичных сигналов видео информацию, указывающую это). В этом примере видеокодер 20 может не сигнализировать индексы к поднабору преобразований, что уменьшает количество информации, которую нужно сигнализировать, потому что видеодекодер 30 на основе принятой информации может определить, что преобразование по умолчанию нужно использовать для каждой TU в CU.

В качестве примера видеокодер 20 может сигнализировать (например, формировать в потоке двоичных сигналов) признак, указывающий, должна ли каждая TU в CU применять одинаковое преобразование по умолчанию. Если признак имеет первое значение (например, цифровое высокое), то к каждой TU в CU применялось одинаковое преобразование по умолчанию. Если признак имеет второе значение (например, цифровое низкое), то по меньшей мере к одной TU из CU применялось преобразование, отличное от преобразования по умолчанию. В случае, где по меньшей мере к одной TU в CU применялось иное преобразование, видеокодер 20 при необходимости может выбрать поднаборы преобразований и сигнализировать индексы в поднаборах преобразований (например, ненулевые коэффициенты больше пороговой величины), как описано выше. В случае, где к каждой TU в CU применялось одинаковое преобразование по умолчанию, видеокодер 20 может не сигнализировать никакие индексы ни в каком из поднаборов преобразований, так как видеодекодер 30 уже может определить, какое преобразование использовать.

Видеодекодер 30 может принять признак, указывающий, должна ли каждая TU в CU применять

одинаковое преобразование по умолчанию. Если признак имеет первое значение, то видеодекодер 30 может определить, что не должны выбираться никакие поднаборы преобразований и не должны анализироваться (например, приниматься) никакие индексы к поднаборам преобразований из потока двоичных сигналов. В этом случае видеодекодер 30 может применить преобразование по умолчанию к каждому блоку коэффициентов в CU. Если признак имеет второе значение, то видеодекодер 30 может определить, что нужно выбрать поднаборы преобразований, определить, нужно ли принимать индексы (например, на основе количества ненулевых коэффициентов), и принять индексы в выбранных поднаборах преобразований на основе определения, что индексы нужно принять.

В вышеприведенном примере признак, указывающий, должна ли каждая TU использовать одинаковое преобразование по умолчанию, относится к уровню CU (например, указывая, что каждая TU в CU использует одинаковое преобразование по умолчанию). В некоторых примерах вместо уровня CU признак может относиться к уровню CTU или уровню PU.

Например, видеокодер 20 может сигнализировать (например, формировать в потоке двоичных сигналов) признак, указывающий, все ли блоки преобразований в блоке преобразуются с использованием одинакового преобразования. В ответ на прием признака, указывающего, что не все блоки преобразований в блоке преобразуются с использованием одинакового преобразования, видеодекодер 30 может выбрать поднаборы преобразований и определить индексы в выбранных преобразованиях, как описано выше. В ответ на прием признака, указывающего, что все блоки преобразований в блоке преобразуются с использованием одинакового преобразования, видеодекодер 30 может использовать то преобразование для каждого из блоков преобразований в блоке. В этом примере "блок" может быть одной из CTU, CU или PU, в качестве нескольких примеров.

В качестве резюме сигнализация преобразования для использования для каждой TU может выполняться на уровне TU, когда текущая CU использует дополнительные преобразования, например, как описано выше. Например, видеокодер 20 может отправить один признак для каждой CU, указывающий, кодируются ли TU в ней с помощью дополнительных преобразований (например, с использованием преобразований помимо имеющихся в HEVC). В качестве альтернативы или дополнительно такое указание может сигнализироваться на уровне LCU (уровне CTU), уровне CU, уровне PU, TU или уровне любого другого блока.

Когда признак указывает, что никакая TU в CU не кодируется с помощью дополнительных преобразований, все TU кодируются с помощью одного преобразования по умолчанию. В одном примере преобразованием по умолчанию является DCT-II. В качестве альтернативы или дополнительно преобразование по умолчанию может зависеть от внутренних/внешних режимов, режима внутреннего предсказания, размера блока, положения TU в PU или любой другой статистики текущей TU. Например, как описано выше, видеокодер 20 и видеодекодер 30 могут определить одинаковое преобразование по умолчанию, и условие, для которого нужно использовать преобразование по умолчанию, может основываться на таких факторах, как внутренний/внешний режимы, режим внутреннего предсказания, размер блока, положение TU в PU или любая другая статистика текущей TU. Таким образом, при использовании преобразований по умолчанию можно уменьшить количество информации, которую нужно сигнализировать.

К тому же указания могут присутствовать в разных иерархиях. Например, видеокодер 20 может сначала сигнализировать (например, сформировать в потоке двоичных сигналов) одноразрядный признак на уровне LCU (CTU), если одноразрядный признак равен 0, то видеокодер 20 и видеодекодер 30 могут применять только DCT-II для каждой CU, в противном случае, если одноразрядный признак равен 1, то видеокодер 20 может сигнализировать другой признак на уровне CU, задающий, могут ли TU в CU использовать множественные преобразования или только преобразование по умолчанию.

В этом примере видеодекодер 30 на каждом иерархическом уровне может определять, все ли TU на конкретном иерархическом уровне используют преобразование по умолчанию. Например, если признак на уровне CTU указывает, что все TU в CTU должны использовать одинаковое преобразование по умолчанию, то видеодекодер 30 может использовать одинаковое преобразование по умолчанию для всех TU в CTU. Если признак на уровне CTU указывает, что не все TU в CTU должны использовать одинаковое преобразование по умолчанию, то видеодекодер 30 может выбрать поднаборы преобразований и определить преобразования для каждой TU в CTU, как описано выше.

В некоторых случаях вместо остановки на уровне CTU и определения преобразований для каждой TU может существовать другой признак для каждой из CU в CTU. Например, видеодекодер 30 может принимать признак для каждой CU в CTU, указывающий, все ли TU в CU используют одинаковое преобразование по умолчанию или не используют одинаковое преобразование по умолчанию. Если для CU видеодекодер 30 принимает признак, указывающий, что все TU в CU используют одинаковое преобразование по умолчанию, то видеодекодер 30 может применить преобразование по умолчанию. Если для CU видеодекодер 30 принимает признак, указывающий, что не все TU в CU используют одинаковое преобразование по умолчанию, то видеодекодер 30 может выбрать поднаборы преобразований и определить преобразования для каждой TU в CU, как описано выше.

В некоторых случаях вместо остановки на уровне CU и определения преобразований для каждой TU может существовать другой признак для каждой из PU в CU. Например, видеодекодер 30 может при-

нимать признак для каждой PU в CU, указывающий, все ли TU в PU используют одинаковое преобразование по умолчанию или не используют одинаковое преобразование по умолчанию. Если для PU видеодекодер 30 принимает признак, указывающий, что все TU в PU используют одинаковое преобразование по умолчанию, то видеодекодер 30 может применить преобразование по умолчанию. Если для PU видеодекодер 30 принимает признак, указывающий, что не все TU в PU используют одинаковое преобразование по умолчанию, то видеодекодер 30 может выбрать поднаборы преобразований и определить преобразования для каждой TU в PU, как описано выше.

В качестве альтернативы или дополнительно, когда признак блочной компенсации движения с перекрытием (OBMC) в CU сигнализируется как снятый, одноразрядный признак, который указывает, применяется ли только одно преобразование по умолчанию, не сигнализируется для текущей CU, а подразумевается значение по умолчанию (например, 0), которое указывает, что применяется преобразование по умолчанию (например, DCT-II). В качестве альтернативы или дополнительно моделирование контекста CABAC одноразрядного признака одного блока, который указывает, применяется ли только одно преобразование по умолчанию, зависит от признака OBMC текущего блока, когда OBMC разрешена для текущей секции (например, зависит от значения признака OBMC).

В одном примере, когда признак OBMC (либо выведенный неявно, либо явно сигнализированный) является истиной (то есть равен 1), видеокодер 20 и видеодекодер 30 могут использовать один набор моделей контекста для CABAC-кодирования или CABAC-декодирования одноразрядного признака. Когда признак OBMC является ложью (то есть равен 0), другой набор моделей контекста может использоваться для кодирования одноразрядного признака. Кроме того, в качестве альтернативы или дополнительно инициализированные вероятности двух наборов моделей контекста могут отличаться. В качестве альтернативы или дополнительно моделирование контекста CABAC одноразрядного признака одного блока, который указывает, применяется ли только одно преобразование по умолчанию, зависит от значения соответствующего одноразрядного признака пространственных соседних блоков (например, левого соседнего блока и/или верхнего соседнего блока) или временных соседних блоков (например, совмещенного блока в эталонном изображении).

Когда у CU есть дополнительные разрешенные преобразования (например, что означает больше ограниченных выборов HEVC) для каждой TU, видеокодер 20 может сигнализировать, а видеодекодер 30 может принимать индексы к преобразованиям из возможных преобразований (набора или поднабора), как описано выше. В качестве альтернативы или дополнительно видеокодер 20 может сигнализировать такую информацию, а видеодекодер 30 может принимать такую информацию на уровне LCU, уровне CU, уровне PU или уровне любого другого блока. Когда видеокодер 20 сигнализирует индикатор на уровне LCU, уровне CU, уровне PU или уровне любого другого блока, все включенные TU в тот уровень могут использовать одинаковую пару преобразований.

Например, видеокодер 20 и видеодекодер 30 могут выбирать поднаборы преобразований, как описано выше (например, на основе режима внутреннего предсказания или на основе местоположения TU для межкадрового предсказания). В некоторых примерах видеокодер 20 может сигнализировать индексы для каждого блока преобразований, а видеодекодер 30 может принимать индексы для каждого блока преобразований. Однако в некоторых примерах вместо приема индексов для каждого блока преобразований видеокодер 20 может сигнализировать один индекс для левого преобразования и один индекс для правого преобразования, в качестве нескольких примеров для всех TU в CTU, всех TU в CU или всех TU в PU. В этом примере для каждого поднаборы преобразований, которое видеодекодер 30 выбирает для правого и левого преобразований для TU, видеодекодер 30 может применять преобразование, идентифицированное индексом для всех TU блока (например, в CUT, CU или PU).

Другими словами, в некоторых случаях индексы к поднаборам преобразований могут считаться более "глобальными". Например, видеокодер 20 может сигнализировать индексы для левого преобразования и правого преобразования. В этом случае индексы могут быть глобальными в том смысле, что индексы одинаковы для каждой TU в блоке независимо от конкретных поднаборов преобразований, которые выбираются, где блоком является CTU, CU или PU. В таких примерах видеодекодер 30 может определить из этих глобальных индексов левое и правое преобразования из выбранных поднаборов преобразований. Например, видеодекодер 30 может не анализировать индексы для каждого выбранного поднаборы преобразований для каждого блока преобразований, а вместо этого идентифицировать преобразование для всех блоков преобразований в блоке (например, CTU, CU или PU) на основе глобальных индексов.

Как описано выше, в некоторых примерах видеокодер 20 может не сигнализировать индексы к выбранным поднаборам преобразований. Например, для некоторых TU можно пропустить сигнализацию дополнительных преобразований, если энергия разностного сигнала ограничивается, например, если отсутствует ненулевой коэффициент, переданный для текущей TU. Аналогичный пропуск сигнализации дополнительных преобразований можно применять к LCU, CU, PU или уровню любого другого блока.

В качестве альтернативы или дополнительно индикатор на уровне некоторого блока можно пропустить, если общее количество, или полная абсолютная сумма, или сумма квадратов ненулевых коэффициентов, переданных на уровне некоторого блока, меньше заданного порогового значения. Другими слова-

ми, видеокодер 20 может не сигнализировать индексы к выбранным поднаборам преобразований, если общее количество, или полная абсолютная сумма, или сумма квадратов ненулевых коэффициентов в блоке коэффициентов меньше порогового значения. В таких примерах, если видеодекодер 30 определяет, что общее количество, или полная абсолютная сумма, или сумма квадратов ненулевых коэффициентов меньше заданного порогового значения, то видеодекодер 30 может определить, что не нужно принимать (например, анализировать) индексы к выбранным поднаборам преобразований из потока двоичных сигналов.

В одном примере пороговое значение общего количества ненулевых коэффициентов равно 2. В качестве альтернативы или дополнительно пороговое значение для общего количества ненулевых коэффициентов может отличаться для разных размеров блоков или разных режимов внутреннего предсказания.

В некоторых примерах, когда размер LCU, CU, PU или блока больше либо меньше предопределенного порогового значения или находится в заданном диапазоне порогового значения, видеокодер 20 может пропустить сигнализацию индикатора (например, индексы к поднаборам преобразований), и видеокодер 20 и видеодекодер 30 могут применять только тип преобразования по умолчанию. В одном примере преобразованием по умолчанию является DCT-II. Кроме того, когда размер CU больше 32×32, видеокодер 20 может не сигнализировать индикатор, и видеокодер 20 и видеодекодер 30 могут применять только DCT-II для каждой TU.

Видеокодер 20 и видеодекодер 30 могут преобразовывать в двоичную форму индикаторы (например, индексы к поднаборам преобразований) с использованием, например, кода с фиксированной длиной, усеченного унарного кода или экспоненциального кода Голомба. Видеокодер 20 и видеодекодер 30 могут энтропийно кодировать (например, кодировать или декодировать соответственно) индикаторы с использованием CABC с контекстами, и для каждого элемента дискретизации применяется один контекст. В одном примере модель контекста выбирается на основе индекса элемента дискретизации. Кроме того, в другом примере при выборе моделей контекста также учитывается режим внутреннего предсказания, или размер TU, или глубина TU. В качестве альтернативы или дополнительно часть элементов дискретизации кодируется с помощью моделей контекста, а оставшиеся элементы дискретизации кодируются с режимом обхода. В качестве альтернативы или дополнительно индикаторы могут кодироваться с обходом, то есть не применяется никакое моделирование контекста.

В примерных методиках видеокодер 20 может сигнализировать различную информацию в потоке двоичных сигналов, а видеодекодер 30 может принимать такую информацию из потока двоичных сигналов. Видеокодер 20 может сигнализировать такую информацию, а видеодекодер 30 может принимать такую информацию из разных мест.

В качестве одного примера синтаксис, связанный с множественными преобразованиями, можно представлять в высокоуровневом синтаксисе. Видеокодер 20 может сигнализировать (например, формировать в потоке двоичных сигналов), а видеодекодер 30 может принимать количество возможных преобразований, как описано выше, по отношению к преобразованиям, выбираемым из двух или более возможных преобразований, для использования в наборе параметров изображения (PPS), наборе параметров последовательности (SPS) или любых других местах, включая даже заголовок секции. Видеокодер 20 может сигнализировать (например, формировать в потоке двоичных сигналов), а видеодекодер 30 может принимать количество возможных преобразований в каждом поднаборе, как описано выше, по отношению к предварительному выбору из трех или более возможных преобразований, в заголовке секции, наборе параметров изображения (PPS), наборе параметров последовательности (SPS) или любых других местах.

Признак или индекс можно сигнализировать в заголовке секции, PPS, SPS или любых других местах для указания, применяется ли вышеупомянутое множественное преобразование на уровне блока. Как описано выше, одон выделенное значение этого признака или индекса может указывать, что все TU кодируются с помощью одного преобразования по умолчанию. Дополнительно или в качестве альтернативы одно выделенное значение этого признака или индекса может указывать, что признак/индекс или признаки/индексы могут сигнализироваться на уровне блока для выбора преобразования на уровне блока. Также размер блока, для которого не применяются множественные преобразования (когда размер больше сигнализированного размера либо меньше сигнализированного размера, или находится в диапазоне двух сигнализированных размеров), может присутствовать в наборе параметров, например, наборе параметров изображения или наборе параметров последовательности.

Повторимся, что вышеприведенное описание использует термин "преобразование". Однако следует понимать, что видеокодер 20 использует преобразование для формирования блока преобразований значений коэффициентов преобразования из остаточного блока. С другой стороны видеодекодер 30 использует обратное преобразование для формирования остаточного блока остаточных значений из блока преобразований. Соответственно, в вышеприведенном описании следует понимать, что описание преобразования в равной степени применимо к видеодекодеру 3 0; однако видеодекодер 30 использует обратное преобразование.

Фиг. 5 - блок-схема, иллюстрирующая примерный видеокодер 20, который может реализовать методики из данного раскрытия изобретения. Фиг. 5 предоставляется с целью объяснения, и ее не следует

рассматривать как ограничивающую методики, которые в общих чертах иллюстрируются и описываются в данном раскрытии изобретения. С целью объяснения данное раскрытие изобретения описывает видеокодер 20 применительно к кодированию HEVC. Однако методики из данного раскрытия изобретения могут применяться к другим стандартам или способам кодирования. Например, видеокодер 20 может конфигурироваться для применения к блоку преобразований большего количества преобразований, чем ограниченные варианты, предусмотренные в HEVC.

В примере из фиг. 5 видеокодер 20 включает в себя модуль 100 обработки с предсказанием, память 101 видеоданных, модуль 102 формирования остатка, модуль 104 обработки с преобразованием, модуль 106 квантования, модуль 108 обратного квантования, модуль 110 обработки с обратным преобразованием, модуль 112 восстановления, модуль 114 фильтров, буфер 116 декодированных изображений и модуль 118 энтропийного кодирования. Модуль 100 обработки с предсказанием включает в себя модуль 120 обработки с межкадровым предсказанием и модуль 126 обработки с внутренним предсказанием. Модуль 120 обработки с межкадровым предсказанием включает в себя модуль оценки движения и модуль компенсации движения (не показаны). В других примерах видеокодер 20 может включать в себя больше, меньше компонентов или другие функциональные компоненты.

Память 101 видеоданных может хранить видеоданные для кодирования с помощью компонентов видеокодера 20. Видеоданные, сохраненные в памяти 101 видеоданных, можно получить, например, из источника 18 видео. Буфер 116 декодированных изображений может быть запоминающим устройством эталонных изображений, которое хранит эталонные видеоданные для использования при кодировании видеоданных видеокодером 20, например, в режимах внутреннего или межкадрового кодирования. Память 101 видеоданных и буфер 116 декодированных изображений могут быть образованы любым из ряда запоминающих устройств, например, динамическим оперативным запоминающим устройством (DRAM), включая синхронное DRAM (SDRAM), магниторезистивное RAM (MRAM), резистивное RAM (RRAM) или другие типы запоминающих устройств. Память 101 видеоданных и буфер 116 декодированных изображений могут предоставляться одним запоминающим устройством либо отдельными запоминающими устройствами. В различных примерах память 101 видеоданных может находиться на кристалле с другими компонентами видеокодера 20 или вне кристалла относительно тех компонентов.

Видеокодер 20 может принять видеоданные. Видеокодер 20 может кодировать каждую CTU в секции изображения в видеоданных. Каждая из CTU может ассоциироваться с блоками дерева кодирования (CTB) яркости одинакового размера и соответствующими CTB изображения. Модуль 100 обработки с предсказанием может выполнить разбиение квадродерева как часть кодирования CTU, чтобы разделить CTB в CTU на постепенно более мелкие блоки. Более мелкий блок может быть блоками кодирования в CU. Например, модуль 100 обработки с предсказанием может разбить CTB, ассоциированный с CTU, на четыре субблока одинакового размера, разбить один или несколько субблоков на четыре субблока одинакового размера, и так далее.

Видеокодер 20 может кодировать CU в CTU, чтобы сформировать кодированные представления CU (то есть кодированные CU). Как часть кодирования CU модуль 100 обработки с предсказанием может распределить блоки кодирования, ассоциированные с CU, между одной или несколькими PU в CU. Таким образом, каждая PU может ассоциироваться с блоком предсказания яркости и соответствующими блоками предсказания цветности. Видеокодер 20 и видеодекодер 30 могут поддерживать PU, имеющие различные размеры. Как указано выше, размер CU может относиться к размеру блока кодирования яркости в CU, а размер PU может относиться к размеру блока предсказания яркости в PU. Предполагая размер конкретной CU равным 2N×2N, видеокодер 20 и видеодекодер 30 могут поддерживать размеры PU в 2N×2N или N×N для внутреннего предсказания и симметричные размеры PU в 2N×2N, 2N×N, N×2N, N×N или аналогичных для межкадрового предсказания. Видеокодер 20 и видеодекодер 30 также могут поддерживать асимметричное разбиение для размеров PU в 2N×NU, 2N×ND, nL×2N и nR×2N для межкадрового предсказания.

Модуль 120 обработки с межкадровым предсказанием может формировать данные с предсказанием для PU путем выполнения межкадрового предсказания над каждой PU в CU. Данные с предсказанием для PU могут включать в себя блоки с предсказанием из PU и информацию о движении для PU. Модуль 121 межкадрового предсказания может выполнять разные операции для PU в CU в зависимости от того, находится ли PU в I-секции, P-секции или B-секции. В I-секции все PU имеют внутреннее предсказание. Поэтому, если PU находится в I-секции, то модуль 121 межкадрового предсказания не выполняет межкадровое предсказание над этой PU. Таким образом, для кодированных в I-режиме блоков предсказанный блок образуется с использованием пространственного предсказания из ранее кодированных соседних блоков в том же кадре.

Если PU находится в P-секции, то модуль оценки движения в модуле 120 обработки с межкадровым предсказанием может искать эталонную область для PU в эталонных изображениях в списке эталонных изображений (например, "RefPicList0").

Эталонная область для PU может быть областью в эталонном изображении, которая содержит выборочные блоки, которые точнее всего соответствуют выборочным блокам в PU. Модуль оценки движе-

ния может формировать индекс эталона, который указывает положение в RefPicList0 эталонного изображения, содержащего эталонную область для PU. К тому же модуль оценки движения может формировать MV, который указывает пространственное смещение между блоком кодирования в PU и эталонным местоположением, ассоциированным с эталонной областью. Например, MV может быть двумерным вектором, который предоставляет смещение от координат в текущем декодированном изображении к координатам в эталонном изображении. Модуль оценки движения может вывести индекс эталона и MV в качестве информации о движении PU. Модуль компенсации движения в модуле 120 обработки с межкадровым предсказанием может формировать блоки с предсказанием в PU на основе фактических или интерполированных выборок в эталонном местоположении, указанном вектором движения PU.

Если PU находится в B-секции, то модуль оценки движения в модуле 120 обработки с межкадровым предсказанием может выполнить однонаправленное предсказание или двунаправленное предсказание для PU. Чтобы выполнить однонаправленное предсказание для PU, модуль оценки движения может искать эталонную область для PU в эталонных изображениях из RefPicList0 или второго списка эталонных изображений ("RefPicList1"). Модуль оценки движения в качестве информации о движении PU может вывести индекс эталона, который указывает положение эталонного изображения, которое содержит эталонную область, в RefPicList0 или RefPicList1, MV, который указывает пространственное смещение между блоком предсказания в PU и эталонным местоположением, ассоциированным с эталонной областью, и один или несколько индикаторов направления предсказания, которые указывают, находится ли эталонное изображение в RefPicList0 либо в RefPicList1. Модуль компенсации движения в модуле 120 обработки с межкадровым предсказанием может формировать блоки с предсказанием в PU по меньшей мере частично на основе фактических или интерполированных выборок в эталонной области, указанной вектором движения PU.

Чтобы выполнить двунаправленное межкадровое предсказание для PU, модуль оценки движения может искать эталонную область для PU в эталонных изображениях в RefPicList0, а также может искать другую эталонную область для PU в эталонных изображениях в RefPicList1. Модуль оценки движения может сформировать индексы эталонных изображений, которые указывают положения эталонных изображений, которые содержат эталонные области, в RefPicList0 и RefPicList1. К тому же модуль оценки движения может формировать MV, которые указывают пространственные смещения между эталонным местоположением, ассоциированным с эталонными областями, и выборочным блоком в PU. Информация о движении PU может включать в себя индексы эталона и MV у PU. Модуль компенсации движения в модуле 120 обработки с межкадровым предсказанием может формировать блоки с предсказанием в PU по меньшей мере частично на основе фактических или интерполированных выборок в эталонных областях, указанных векторами движения PU.

Модуль 126 обработки с внутренним предсказанием может сформировать данные с предсказанием для PU путем выполнения внутреннего предсказания над PU. Данные с предсказанием для PU могут включать в себя блоки с предсказанием для PU и различные синтаксические элементы. Модуль 126 обработки с внутренним предсказанием может выполнять внутреннее предсказание над PU в I-секциях, P-секциях и B-секциях.

Чтобы выполнить внутреннее предсказание над PU, модуль 126 обработки с внутренним предсказанием может использовать несколько режимов внутреннего предсказания для формирования нескольких наборов данных с предсказанием для PU. Модуль 126 обработки с внутренним предсказанием может использовать выборки из выборочных блоков соседних PU для формирования блока с предсказанием для PU. Соседние PU могут находиться выше, выше и справа, выше и слева или слева от PU, предполагая порядок кодирования слева-направо, сверху-вниз для PU, CU и CTU. Модуль 126 обработки с внутренним предсказанием может использовать различные количества режимов внутреннего предсказания, например 35 режимов направленного внутреннего предсказания. В некоторых примерах количество режимов внутреннего предсказания может зависеть от размера области, ассоциированной с PU.

Модуль 100 обработки с предсказанием может выбирать данные с предсказанием для PU в CU из данных с предсказанием, сформированных для PU модулем 120 обработки с межкадровым предсказанием, или данных с предсказанием, сформированных для PU модулем 126 обработки с внутренним предсказанием. В некоторых примерах модуль 100 обработки с предсказанием выбирает данные с предсказанием для PU в CU на основе показателей искажения в зависимости от скорости передачи у наборов данных с предсказанием. Блоки с предсказанием в выбранных данных с предсказанием в этом документе могут называться выбранными блоками с предсказанием.

В описанных в данном раскрытии изобретения примерах методики применяются, когда видеоблок кодируется с внутренним предсказанием или с межкадровым предсказанием. Например, когда блок кодируется с внутренним предсказанием, режим внутреннего предсказания может использоваться для определения поднаборов преобразований. Когда блок кодируется с межкадровым предсказанием, его положение может использоваться для определения поднаборов преобразований. Соответственно, примерные методики применяются к видеоблоку, который кодируется с внутренним предсказанием в любом из режимов внутреннего предсказания или кодируется с межкадровым предсказанием однонаправленно или двунаправленно.

Кроме того, примерные методики не ограничиваются внутренним предсказанием или межкадровым предсказанием и с тем же успехом могут быть распространены на режим внутриблочного копирования (IBC). В режиме IBC блок с предсказанием находится в том же изображении, что и кодируемый видеоблок, и идентифицируется вектором блока. В режиме IBC поднаборы преобразований могут выбираться, в качестве нескольких примеров, из положения видеоблока, положения блока с предсказанием или вектора блока.

Модуль 102 формирования остатка на основе блока кодирования яркости, Cb и Cr в CU и выбранных блоков яркости, Cb и Cr с предсказанием у PU в CU может сформировать остаточные блоки яркости, Cb и Cr в CU. Например, модуль 102 формирования остатка может сформировать остаточные блоки в CU так, что каждая выборка в остаточных блоках имеет значение, равное разности между выборкой в блоке кодирования в CU и соответствующей выборкой в соответствующем выбранном блоке с предсказанием у PU в CU.

Модуль 104 обработки с преобразованием может выполнить разбиение квадродерева для разбиения остаточных блоков, ассоциированных с CU, на блоки преобразований, ассоциированные с TU в CU. Таким образом, TU может ассоциироваться с блоком преобразований яркости и двумя блоками преобразований цветности. Размеры и положения блоков преобразований яркости и цветности у TU в CU могут основываться или не основываться на размерах и положениях блоков предсказания у PU в CU. Структура квадродерева, известная как "остаточное квадродерево" (RQT), может включать в себя узлы, ассоциированные с каждой из областей. TU в CU могут соответствовать листам в RQT.

Модуль 104 обработки с преобразованием может сформировать блоки коэффициентов преобразования для каждой TU в CU путем применения одного или нескольких преобразований к блокам преобразований в TU. Модуль 104 обработки с преобразованием может применять различные преобразования к блоку преобразований, ассоциированному с TU. Например, модуль 104 обработки с преобразованием может применять к блоку преобразований дискретное косинусное преобразование (DCT), направленное преобразование или концептуально сходное преобразование. В некоторых примерах модуль 104 обработки с преобразованием не применяет преобразования к блоку преобразований. В таких примерах блок преобразований может рассматриваться в качестве блока коэффициентов преобразования.

В описанных в данном раскрытии изобретения методиках модуль 104 обработки с преобразованием может применять левое преобразование и правое преобразование к блоку преобразований в TU. В некоторых примерах модуль 100 обработки с предсказанием может определять, какие преобразования применять, с использованием описанных в данном раскрытии изобретения методик.

Например, модуль 100 обработки с предсказанием может определять множество поднаборов преобразований, при этом каждый поднабор идентифицирует одно или несколько возможных преобразований, где по меньшей мере один поднабор преобразований идентифицирует множество возможных преобразований. Возможные преобразования имеют разные типы преобразований, и в некоторых примерах модуль 100 обработки с предсказанием определяет множество поднаборов преобразований на основе размера кодируемого видеоблока.

В некоторых примерах память 101 видеоданных хранит множество поднаборов преобразований, и модуль 100 обработки с предсказанием может определять множество поднаборов преобразований из сохраненных поднаборов преобразований. В некоторых примерах память 101 видеоданных может хранить все преобразования, и модуль 100 обработки с предсказанием может создавать поднаборы преобразований предопределенным образом. Примеры возможных преобразований включают в себя DCT-I - DCT-VIII, DST-I - DST-VIII, преобразования KLT и т.п. В некоторых примерах множество поднаборов преобразований включает в себя три или более поднаборов преобразований.

Модуль 100 обработки с предсказанием может выбрать первый поднабор преобразований из множества поднаборов преобразований для левого преобразования для текущего блока преобразований в видеоблоке видеоданных и выбрать второй поднабор преобразований из множества поднаборов преобразований для правого преобразования для блока преобразований в видеоблоке видеоданных. Текущий блок преобразований может быть блоком преобразований, который модуль 104 обработки с преобразованием формирует и к которому модуль 104 обработки с преобразованием должен применить преобразования.

Модуль 100 обработки с предсказанием может определить левое преобразование из выбранного первого поднабора преобразований и определить правое преобразование из выбранного второго поднабора преобразований. Например, модуль 100 обработки с предсказанием может проверить каждое из преобразований в выбранных поднаборах преобразований и определить, какое преобразование обеспечивает наилучшее кодирование видео. Модуль 100 обработки с предсказанием может определить соответствующие преобразования, которые обеспечивают наилучшее кодирование видео, в качестве левого преобразования и правого преобразования.

Модуль 104 обработки с преобразованием может определять текущий блок коэффициентов на основе левого преобразования, правого преобразования и текущего блока преобразований. Например, модуль 104 обработки с преобразованием может решать следующее уравнение: $Y=C*X*R^T$, где C - левое преобразование, R - правое преобразование, X - текущий блок преобразований, а Y - результирующий

текущий блок коэффициентов.

Если видеоблок (например, CU или PU) кодируется с внутренним предсказанием, то модуль 100 обработки с предсказанием может определить режим внутреннего предсказания у видеоблока. Модуль 100 обработки с предсказанием может выбрать первый поднабор преобразований на основе определенного режима внутреннего предсказания и выбрать второй поднабор преобразований на основе определенного режима внутреннего предсказания.

Если видеоблок (например, CU или PU) кодируется с межкадровым предсказанием, то модуль 100 обработки с предсказанием может определить местоположение текущего блока преобразований в видеоблоке (например, определить, предназначен ли блок преобразований для остатка, сформированного из конкретного местоположения в видеоблоке). Модуль 100 обработки с предсказанием может выбрать первый поднабор преобразований на основе определенного местоположения текущего блока преобразований и выбрать второй поднабор преобразований на основе определенного местоположения текущего блока преобразований.

Для внутреннего предсказания или межкадровыпредсказания в некоторых примерах модуль 100 обработки с предсказанием может побудить модуль 118 энтропийного кодирования сигнализировать (например, сформировать в потоке двоичных сигналов) индекс первого поднабора преобразований к первому поднабору преобразований, чтобы идентифицировать преобразование в первом поднаборе преобразований, используемое для определения текущего блока коэффициентов, и сигнализировать (например, сформировать в потоке двоичных сигналов) индекс второго поднабора преобразований ко второму поднабору преобразований, чтобы идентифицировать преобразование во втором поднаборе преобразований, используемое для определения текущего блока коэффициентов. В некоторых примерах модуль 100 обработки с предсказанием может определить количество ненулевых коэффициентов в текущем блоке коэффициентов. В этих примерах модуль 100 обработки с предсказанием может побудить модуль 118 энтропийного кодирования сигнализировать индекс первого поднабора преобразований на основе количества ненулевых коэффициентов, которое больше пороговой величине, и сигнализировать индекс второго поднабора преобразований на основе количества ненулевых коэффициентов, которое больше пороговой величины. Если количество ненулевых коэффициентов меньше пороговой величины, то модуль 100 обработки с предсказанием может не побуждать модуль 118 энтропийного кодирования сигнализировать индексы в первом и втором поднаборах преобразований.

В некоторых примерах по меньшей мере одно из первого поднабора преобразований или второго поднабора преобразований включает в себя преобразование, которое отличается от дискретного косинусного преобразования (DCT)-II и дискретного синусного преобразования (DST)-VII. В некоторых примерах первый поднабор преобразований и второй поднабор преобразований включают в себя разные преобразования (например, по меньшей мере одно преобразование в первом поднаборе преобразований не находится во втором поднаборе преобразований, или наоборот).

Модуль 106 квантования может квантовать коэффициенты преобразования в блоке коэффициентов. Процесс квантования может уменьшить разрядную глубину, ассоциированную с некоторыми или всеми коэффициентами преобразования. Например, n-разрядный коэффициент преобразования во время квантования можно округлить в меньшую сторону до m-разрядного коэффициента преобразования, где n больше т. Модуль 106 квантования может квантовать блок коэффициентов, ассоциированный с TU в CU, на основе значения параметра квантования (QP), ассоциированного с CU. Видеокодер 20 может регулировать степень квантования, применяемого к блокам коэффициентов, ассоциированным с CU, путем регулирования значения QP, ассоциированного с CU. Квантование может привносить потерю информации; таким образом, квантованные коэффициенты преобразования могут обладать меньшей точностью, нежели исходные коэффициенты преобразования.

Модуль 108 обратного квантования и модуль 110 обработки с обратным преобразованием могут применить к блоку коэффициентов соответственно обратное квантование и обратные преобразования, чтобы восстановить остаточный блок из блока коэффициентов. Модуль 112 восстановления может сложить восстановленный остаточный блок с соответствующими выборками из одного или нескольких блоков с предсказанием, сформированных модулем 100 обработки с предсказанием, чтобы создать восстановленный блок преобразований, ассоциированный с TU. Путем восстановления таким образом блоков преобразований для каждой TU в CU видеокодер 20 может восстановить блоки кодирования в CU.

Модуль 114 фильтров может выполнить одну или несколько операций уменьшения блочности, чтобы уменьшить блочные артефакты в блоках кодирования, ассоциированных с CU. Буфер 116 декодированных изображений может сохранить восстановленные блоки кодирования после того, как модуль 114 фильтров выполняет одну или несколько операций уменьшения блочности над восстановленными блоками кодирования. Модуль 120 обработки с межкадровым предсказанием может использовать эталонное изображение, которое содержит восстановленные блоки кодирования, чтобы выполнить межкадровое предсказание над PU других изображений. К тому же модуль 126 обработки с внутренним предсказанием может использовать восстановленные блоки кодирования в буфере 116 декодированных изображений, чтобы выполнить внутреннее предсказание над другими PU в том же изображении, что и CU.

Модуль 118 энтропийного кодирования может принимать данные от других функциональных ком-

понентов видеокодера 20. Например, модуль 118 энтропийного кодирования может принимать блоки коэффициентов от модуля 106 квантования (например, информацию, указывающую коэффициенты текущего блока коэффициентов, используемые для восстановления видеоблока) и может принимать синтаксические элементы от модуля 100 обработки с предсказанием (например, индексы к первому и второму поднаборам преобразований). Модуль 118 энтропийного кодирования может выполнить одну или несколько операций энтропийного кодирования над данными, чтобы сформировать энтропийно кодированные данные. Например, модуль 118 энтропийного кодирования может выполнить операцию контекстно-адаптивного кодирования с переменной длиной (CAVLC), операцию CABAC, операцию кодирования переменной длины с неравномерным кодом (V2V), операцию синтаксического контекстно-адаптивного двоичного арифметического кодирования (SBAC), операцию энтропийного кодирования с разбиением на интервалы вероятности (PIPE), операцию экспоненциального кодирования Голомба или другой тип операции энтропийного кодирования над данными. Видеокодер 20 может вывести поток двоичных сигналов, который включает в себя энтропийно кодированные данные, сформированные модулем 118 энтропийного кодирования. Например, поток двоичных сигналов может включать в себя данные, которые представляют RQT для CU.

В примерных методиках модуль 100 обработки с предсказанием определяет блок с предсказанием и формирует в потоке двоичных сигналов видео, который выводит модуль 118 энтропийного кодирования, информацию, указывающую режим предсказания видеоблока, на основе блока с предсказанием. Режим предсказания указывает, кодируется ли видеоблок с внутренним предсказанием или с межкадровым предсказанием. Например, блок с предсказанием является блоком в том же изображении, что и видеоблок на основе видеоблока с внутренним предсказанием, либо в изображении, отличном от изображения, которое включает в себя видеоблок на основе видеоблока с межкадровым предсказанием. Модуль 102 формирования остатка может определить текущий блок преобразований как остаток между видеоблоком и блоком с предсказанием.

Фиг. 6 - блок-схема, иллюстрирующая примерный видеодекодер 30, который конфигурируется для реализации методик из данного раскрытия изобретения. Фиг. 6 предоставляется с целью объяснения и не ограничивает методики, которые в общих чертах иллюстрируются и описываются в данном раскрытии изобретения. С целью объяснения данное раскрытие изобретения описывает видеодекодер 30 применительно к кодированию HEVC. Однако методики из данного раскрытия изобретения могут применяться к другим стандартам или способам кодирования.

Видеодекодер 30 представляет собой пример устройства, которое может конфигурироваться для выполнения методик в соответствии с различными примерами, описанными в данном раскрытии изобретения. В примере из фиг. 6 видеодекодер 30 включает в себя модуль 150 энтропийного декодирования, память 151 видеоданных, модуль 152 обработки с предсказанием, модуль 154 обратного квантования, модуль 156 обработки с обратным преобразованием, модуль 158 восстановления, модуль 160 фильтров и буфер 162 декодированных изображений. Модуль 152 обработки с предсказанием включает в себя модуль 164 компенсации движения и модуль 166 обработки с внутренним предсказанием. В других примерах видеодекодер 30 может включать в себя больше, меньше компонентов или другие функциональные компоненты.

Память 151 видеоданных может хранить видеоданные, например кодированный поток двоичных сигналов видео, для декодирования с помощью компонентов видеодекодера 30. Видеоданные, сохраненные в памяти 151 видеоданных, можно получить, например, с машиночитаемого носителя 16, например, из локального источника видео, например камеры, посредством проводной или беспроводной сетевой передачи видеоданных или путем обращения к физическим носителям информации. Память 151 видеоданных может образовывать буфер кодированных изображений (CPB), который хранит кодированные видеоданные из кодированного потока двоичных сигналов видео. Буфер 162 декодированных изображений может быть запоминающим устройством эталонных изображений, которое хранит эталонные видеоданные для использования при декодировании видеоданных видеодекодером 30, например, в режимах внутреннего или межкадрового кодирования. Память 151 видеоданных и буфер 162 декодированных изображений могут быть образованы любым из ряда запоминающих устройств, например, динамическим оперативным запоминающим устройством (DRAM), включая синхронное DRAM (SDRAM), магниторезистивное RAM (MRAM), резистивное RAM (RRAM) или другие типы запоминающих устройств. Память 151 видеоданных и буфер 162 декодированных изображений могут предоставляться одним запоминающим устройством либо отдельными запоминающими устройствами. В различных примерах память 151 видеоданных может находиться на кристалле с другими компонентами видеодекодера 30 или вне кристалла относительно тех компонентов.

Буфер кодированных изображений (CPB) может принимать и хранить кодированные видеоданные (например, единицы NAL) в потоке двоичных сигналов. Модуль 150 энтропийного декодирования может принимать кодированные видеоданные (например, единицы NAL) из CPB и анализировать единицы NAL, чтобы декодировать синтаксические элементы. Модуль 150 энтропийного декодирования может энтропийно декодировать энтропийно кодированные синтаксические элементы в единицах NAL. Модуль 152 обработки с предсказанием, модуль 154 обратного квантования, модуль 156 обработки с обратным

преобразованием, модуль 158 восстановления и модуль 160 фильтров могут сформировать декодированные видеоданные на основе синтаксических элементов, извлеченных из потока двоичных сигналов.

Единицы NAL в потоке двоичных сигналов могут включать в себя единицы NAL кодированной секции. Модуль 150 энтропийного декодирования может извлечь и энтропийно декодировать синтаксические элементы из единиц NAL кодированной секции как часть декодирования потока двоичных сигналов. Каждая из кодированных секций может включать в себя заголовок секции и данные секции. Заголовок секции может содержать синтаксические элементы, имеющие отношение к секции. Синтаксические элементы в заголовке секции могут включать в себя синтаксический элемент, который идентифицирует PPS, ассоциированный с изображением, которое содержит секцию.

В дополнение к декодированию синтаксических элементов из потока двоичных сигналов видеодекодер 30 может выполнять операцию восстановления над не разбитой CU. Чтобы выполнить операцию восстановления над не разбитой CU, видеодекодер 30 может выполнить операцию восстановления над каждой TU в CU. Путем выполнения операции восстановления для каждой TU в CU видеодекодер 30 может восстановить остаточные блоки в CU.

Как часть выполнения операции восстановления над TU в CU модуль 154 обратного квантования может обратно квантовать, то есть деквантовать, блоки коэффициентов, ассоциированные с TU. Модуль 154 обратного квантования может использовать значение QP, ассоциированное с CU в TU, чтобы определить степень квантования, а также степень обратного квантования для применения модулем 154 обратного квантования. То есть можно управлять коэффициентом сжатия, то есть отношением количества разрядов, используемых для представления исходной последовательности и сжатой последовательности, путем регулирования значения QP, используемого при квантовании коэффициентов преобразования. Коэффициент сжатия также может зависеть от применяемого способа энтропийного кодирования.

После того, как модуль 154 обратного квантования обратно квантует блок коэффициентов, модуль 156 обработки с обратным преобразованием может применить одно или несколько обратных преобразований к блоку коэффициентов, чтобы сформировать остаточный блок, ассоциированный с TU. Например, модуль 156 обработки с обратным преобразованием может применить обратное DCT, обратное целочисленное преобразование, обратное преобразование Карунена - Лоэва (KLT), обратное вращательное преобразование, обратное направленное преобразование или другое обратное преобразование к блоку коэффициентов.

В описанных в данном раскрытии изобретения методиках модуль 152 обработки с предсказанием может определять левое и правое преобразования, которые должен применять модуль 156 обработки с обратным преобразованием. Например, модуль 152 обработки с предсказанием может определять множество поднаборов преобразований, при этом каждый поднабор идентифицирует одно или несколько возможных преобразований, где по меньшей мере один поднабор преобразований идентифицирует множество возможных преобразований. Возможные преобразования имеют разные типы преобразований, и в некоторых примерах модуль 152 обработки с предсказанием определяет множество поднаборов преобразований на основе размера декодируемого видеоблока.

В некоторых примерах память 151 видеоданных хранит множество поднаборов преобразований, и модуль 152 обработки с предсказанием может определять множество поднаборов преобразований из сохраненных поднаборов преобразований. В некоторых примерах память 151 видеоданных может хранить все преобразования, и модуль 152 обработки с предсказанием может создавать поднаборы преобразований предопределенным образом. В некоторых примерах модуль 152 обработки с предсказанием может принимать из потока двоичных сигналов информацию, идентифицирующую поднаборы преобразований. Примеры возможных преобразований включают в себя DCT-I - DCT-VIII, DST-I - DST-VIII, преобразования KLT и т.п. В некоторых примерах множество поднаборов преобразований включает в себя три или более поднаборов преобразований.

Модуль 152 обработки с предсказанием может выбрать первый поднабор преобразований из множества поднаборов преобразований для левого преобразования для текущего блока коэффициентов видеоданных и выбрать второй поднабор преобразований из множества поднаборов преобразований для правого преобразования для текущего блока коэффициентов видеоданных. Модуль 152 обработки с предсказанием может определить левое преобразование из выбранного первого поднабора преобразований и определить правое преобразование из выбранного второго поднабора преобразований.

Модуль 156 обработки с обратным преобразованием может определять текущий блок преобразований на основе левого преобразования, правого преобразования и текущего блока коэффициентов. Например, модуль 156 обработки с обратным преобразованием может решать обратное к следующему уравнению: $Y=C*X*R^T$, где Y - блок коэффициентов, C - левое преобразование, X блок преобразований, а R - правое преобразование. Опять в данном раскрытии изобретения следует понимать, что модуль 156 обработки с обратным преобразованием применяет обратное преобразование к тому, что применяет видеокодер 20, но для простоты видеодекодер 30 описывается как применяющий преобразование.

Модуль 152 обработки с предсказанием может восстановить (например, декодировать с внутренним предсказанием или межкадровым предсказанием) видеоблок на основе текущего блока преобразований и блока с предсказанием. Например, если PU кодируется с использованием внутреннего предсказания, то

модуль 166 обработки с внутренним предсказанием может выполнить внутреннее предсказание, чтобы сформировать блоки с предсказанием для PU. Модуль 166 обработки с внутренним предсказанием может использовать режим внутреннего предсказания, чтобы сформировать блоки яркости, Cb и Cr с предсказанием для PU на основе блоков предсказания пространственно-соседних PU. Модуль 166 обработки с внутренним предсказанием может определить режим внутреннего предсказания для PU на основе одного или нескольких синтаксических элементов, декодированных из потока двоичных сигналов.

Модуль 152 обработки с предсказанием может создать первый список эталонных изображений (RefPicList0) и второй список эталонных изображений (RefPicList1) на основе синтаксических элементов, извлеченных из потока двоичных сигналов. Кроме того, если PU кодируется с использованием межкадрового предсказания, то модуль 150 энтропийного декодирования может извлечь информацию о движении для PU. Модуль 164 компенсации движения на основе информации о движении PU может определить одну или несколько эталонных областей для PU. Модуль 164 компенсации движения на основе выборочных блоков в одном или нескольких эталонных блоках для PU может сформировать для PU блоки яркости, Cb и Cr с предсказанием.

В описанных в данном раскрытии изобретения примерах методики применяются, когда видеоблок кодируется с внутренним предсказанием или с межкадровым предсказанием. Например, когда блок кодируется с внутренним предсказанием, режим внутреннего предсказания может использоваться для определения поднаборов преобразований. Когда блок кодируется с межкадровым предсказанием, его положение может использоваться для определения поднаборов преобразований. Соответственно, примерные методики применяются к видеоблоку, который кодируется с внутренним предсказанием в любом из режимов внутреннего предсказания или кодируется с межкадровым предсказанием однонаправленно или двунаправленно.

Кроме того, примерные методики не ограничиваются внутренним предсказанием или межкадровым предсказанием и с тем же успехом могут быть распространены на режим внутриблочного копирования (IBC). В режиме IBC эталонный блок, используемый для образования блока с предсказанием, находится в том же изображении, что и кодируемый видеоблок, и идентифицируется вектором блока. В режиме IBC поднаборы преобразований могут выбираться, в качестве нескольких примеров, из положения видеоблока, положения эталонного блока или вектора блока.

Модуль 158 восстановления может использовать блоки преобразований яркости, Cb и Cr, ассоциированные с TU в CU, и блоки яркости, Cb и Cr с предсказанием у PU в CU, то есть данные внутреннего предсказания либо данные межкадрового предсказания, в зависимости от обстоятельств, чтобы восстановить блоки кодирования яркости, Cb и Cr в CU. Например, модуль 158 восстановления может сложить выборки блоков преобразований яркости, Cb и Cr с соответствующими выборками блоков яркости, Cb и Cr с предсказанием, чтобы восстановить блоки кодирования яркости, Cb и Cr в CU.

Модуль 160 фильтров может выполнить операцию уменьшения блочности, чтобы уменьшить блочные артефакты, ассоциированные с блоками кодирования яркости, Cb и Cr в CU. Видеодекодер 30 может сохранить в буфере 162 декодированных изображений блоки кодирования яркости, Cb и Cr в CU. Буфер 162 декодированных изображений может предоставлять эталонные изображения для последующей компенсации движения, внутреннего предсказания и показа на устройстве отображения, например устройстве 32 отображения из фиг. 1. Например, видеодекодер 30 на основе блоков яркости, Cb и Cr в буфере 162 декодированных изображений может выполнить операции внутреннего предсказания или межкадрового предсказания над PU других CU.

В некоторых примерах, где видеоблок должен декодироваться с внутренним предсказанием, модуль 152 обработки с предсказанием может определить режим внутреннего предсказания видеоблока. Модуль 152 обработки с предсказанием может выбрать первый поднабор преобразований на основе определенного режима внутреннего предсказания и выбрать второй поднабор преобразований на основе определенного режима внутреннего предсказания.

Там, где видеоблок нужно декодировать с межкадровым предсказанием, модуль 152 обработки с предсказанием может определить местоположение текущего блока преобразований в видеоблоке (например, определить, предназначен ли блок коэффициентов для остатка, сформированного из конкретного местоположения в видеоблоке). Модуль 152 обработки с предсказанием может выбрать первый поднабор преобразований на основе определенного местоположения текущего блока преобразований и выбрать второй поднабор преобразований на основе определенного местоположения текущего блока преобразований.

В некоторых примерах модуль 152 обработки с предсказанием может принять индекс первого поднабора преобразований к первому поднабору преобразований и принять индекс второго поднабора преобразований ко второму поднабору преобразований. В этих примерах модуль 152 обработки с предсказанием может определить левое преобразование на основе преобразования в первом поднаборе преобразований, идентифицированного с помощью индекса первого поднабора преобразований, и определить правое преобразование на основе преобразования во втором поднаборе преобразований, идентифицированного с помощью индекса второго поднабора преобразований.

Однако модулю 152 обработки с предсказанием может не требоваться принимать индексы в первом

и втором поднаборах преобразований. Например, модуль 152 обработки с предсказанием может определить, что количество ненулевых коэффициентов в текущем блоке коэффициентов меньше пороговой величины. В таких случаях модуль 152 обработки с предсказанием может в ответ на определение, что количество ненулевых коэффициентов в текущем блоке коэффициентов меньше пороговой величины, определить, что первое преобразование, идентифицированное в первом поднаборе преобразований, является левым преобразованием без приема индекса поднаборы преобразований к первому поднабору преобразований, и в ответ на определение, что количество ненулевых коэффициентов в текущем блоке коэффициентов меньше пороговой величины, определить, что первое преобразование, идентифицированное во втором поднаборе преобразований, является правым преобразованием без приема индекса поднаборы преобразований ко второму поднабору преобразований.

Также модулю 152 обработки с предсказанием не обязательно определять преобразования из поднаборов преобразований во всех случаях. В некоторых примерах модуль 152 обработки с предсказанием может принимать признак, указывающий, что не все блоки преобразований в блоке, который включает в себя текущий блок преобразований, преобразуются с использованием одинакового преобразования. В таких примерах модуль 152 обработки с предсказанием может выбрать первое и второе преобразования и определить левое и правое преобразования из соответствующих первого и второго преобразований в ответ на прием признака, указывающего, что не все блоки преобразований в блоке, который включает в себя текущий блок преобразований, преобразуются с использованием одинакового преобразования. Примеры блока включают в себя единицу дерева кодирования (CTU), единицу кодирования (CU) или единицу предсказания (PU).

В некоторых примерах по меньшей мере одно из первого поднабора преобразований или второго поднабора преобразований включает в себя преобразование, которое отличается от дискретного косинусного преобразования (DCT)-II и дискретного синусного преобразования (DST)-VII. В некоторых примерах первый поднабор преобразований и второй поднабор преобразований включают в себя разные преобразования (например, по меньшей мере одно преобразование в первом поднаборе преобразований не находится во втором поднаборе преобразований, или наоборот).

В примерных методиках видеодекодер 30 может принимать из потока двоичных сигналов информацию, указывающую режим предсказания (например, кодируется ли видеоблок с внутренним предсказанием или с межкадровым предсказанием), и принимать из потока двоичных сигналов информацию, указывающую коэффициенты в текущем блоке коэффициентов. Модуль 152 обработки с предсказанием может определить блок с предсказанием на основе режима предсказания, и модуль 156 обработки с обратным преобразованием или модуль 152 обработки с предсказанием могут создать блок коэффициентов на основе принятой информации, указывающей коэффициенты. Режим предсказания является одним из режима межкадрового предсказания или режима внутреннего предсказания, и текущий блок преобразований является остатком от видеоблока и блока с предсказанием.

Описанные выше методики могут выполняться видеокодером 20 (фиг. 4 и 5) и/или видеодекодером 30 (фиг. 4 и 6), которые оба в целом могут называться кодировщиком видео. Также кодирование видео может, в зависимости от обстоятельств, относиться к кодированию видео или декодированию видео. К тому же кодирование видео и декодирование видео могут в общем называться "обработкой" видеоданных.

В нижеследующих подразделах будут предоставлены примеры вышеупомянутых методик. На практике любое сочетание любой части примеров может использоваться в качестве новой примерной методики.

Нижеследующее описывает примеры создания дополнительного списка возможных преобразований. Кроме способа преобразования по умолчанию, который всегда применяет DCT-II для всех включенных TU, для каждой TU можно создать дополнительные способы возможных преобразований с учетом выбранных наборов преобразований. В одном примере дополнительный список возможных преобразований для остатков внутреннего и межкадрового предсказания создается следующим образом: Во-первых, набор преобразований задается в виде совокупностей типов преобразований, например, примерный набор преобразований можно задать в виде {DCT-II, DST-VII}, что включает в себя два типа преобразований, то есть DCT-II и DST-VII. На основе двух данных наборов преобразований можно сформировать разные способы преобразований путем выбора одного типа преобразования из первого набора преобразований в качестве горизонтального преобразования и другого типа преобразования из второго набора преобразований в качестве вертикального преобразования. Например, когда набор 0 преобразований {DCT-II, DST-VII} используется для горизонтального преобразования, а набор 1 преобразований {DCT-VIII, DST-VII} используется для вертикального преобразования, можно сформировать в общем четыре способа преобразований в виде:

Таблица 1. Четыре способа преобразований на основе набора преобразований {DCT-II, DST-VII}

	Способ 1 преобразования	Способ 2 преобразования	Способ 3 преобразования	Способ 4 преобразования
Горизонтальное преобразование	DCT-II	DCT-II	DST-VII	DST-VII
Вертикальное преобразование	DCT-VIII	DST-VII	DCT-VIII	DST-VII

Для остатка внутреннего предсказания задается в общем три поднаборы преобразований, включая: поднабор 0 преобразований: {DST-VII, DCT-VIII}, поднабор 1 преобразований: {DST-VII, DST-I} и поднабор 2 преобразований: {DST-VII, DCT-V}. Выбор набора преобразований для горизонтального и вертикального преобразований зависит от режима внутреннего предсказания, как показано в табл. 2 ниже:

Таблица 2. Таблица отображения между режимом внутреннего предсказания и набором преобразований

Внутренний режим	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34
RightTransSubsetIdx	2	1	0	1	0	1	0	1	0	0	0	0	0	1	0	1	0	1	0	1	0	1	0	1	2	2	2	2	2	1	0	1	0	1	0
LeftTransSubsetIdx	2	1	0	1	0	1	0	1	2	2	2	2	2	1	0	1	0	1	0	1	0	1	0	1	0	0	0	0	0	1	0	1	0	1	0

Например, для внутреннего режима 10 типы возможных преобразований для горизонтального (правого) преобразования идут из набора 0 преобразований, включающего в себя DST-VII и DCT-VIII, а типы возможных преобразований для вертикального (левого) преобразования идут из набора 2 преобразований, включающего в себя DST-VII и DCT-V. Поэтому дополнительный список возможных преобразований для внутреннего режима 10 в конечном счете создается, как показано в табл. 3, в которой в общем формируется четыре способа преобразований.

Таблица 3. Пример четырех способов преобразований для режима 10 внутреннего предсказания на основе табл.2

	Способ 1	Способ 2	Способ 3	Способ 4
	преобразования	преобразования	преобразования	преобразования
Горизонтальное преобразование	DST-VII	DST-VII	DCT-VIII	DCT-VIII
Вертикальное преобразование	DST-VII	DCT-V	DST-VII	DCT-V

В соответствии с табл. 2 для каждой TU, принимая во внимание режим внутреннего предсказания, можно сформировать в общем четыре способа преобразований. Для остатка межкадрового предсказания одинаковый набор преобразований {DST-VII, DCT-VIII} используется для горизонтального и вертикального преобразований. Поэтому для каждой TU создается дополнительный список возможных преобразований, как показано в табл.4.

Таблица 4. Четыре способа преобразований для остатка межкадрового предсказания

	Способ 1 преобразования	Способ 2 преобразования	Способ 3 преобразования	Способ 4 преобразования
Горизонтальное преобразование	DST-VII	DST-VII	DCT-VIII	DCT-VIII
Вертикальное преобразование	DST-VII	DCT-VIII	DST-VII	DCT-VIII

Нижеследующее описание является примером сигнализации индикаторов. Чтобы адаптивно приспосабливаться к разному содержимому, сигнализируются индикаторы использования преобразований на CU-уровне и TU-уровне. Индикатор CU-уровня является одноразрядным признаком, указывающим, применяется ли DCT-II по умолчанию для всех TU, включенных в текущую CU. Если одноразрядный признак равен 0, то может применяться только DCT-II по умолчанию для всех TU, включенных в текущую CU; в противном случае для каждой TU может дополнительно сигнализироваться двухразрядный индикатор TU-уровня, и первый разряд задает, какой тип преобразования из заданного набора преобразований применяется в качестве горизонтального преобразования, а второй разряд задает, какой тип преобразования из заданного набора преобразований применяется в качестве вертикального преобразования.

Для остатка внутреннего предсказания индикаторы TU-уровня сигнализируются после коэффици-

ентов, и индикатор TU-уровня не сигнализируется и выводится равным 0, когда общее количество ненулевых коэффициентов в TU не больше 2. В противном случае индикатор TU-уровня сигнализируется явно. Для остатка межкадрового предсказания индикаторы TU-уровня могут сигнализироваться либо до, либо после коэффициентов, и индикатор TU-уровня не сигнализируется, когда в TU нет ненулевых коэффициентов.

Ниже предоставляются синтаксис и семантика предложенных примеров на основе HEVC. В нижеприведенном синтаксисе затенение элементов используется для указания возможных изменений в синтаксисе или иного содействия пониманию.

Синтаксис дерева преобразований

transform_tree(x0, y0, xBase, yBase, log2TrafoSize, trafoDepth, blkIdx) {	**Descriptor**
if(log2TrafoSize <= Log2MaxTrafoSize && log2TrafoSize > Log2MinTrafoSize && trafoDepth < MaxTrafoDepth && !(IntraSplitFlag && (trafoDepth == 0)))	
split_transform_flag[x0][y0][trafoDepth]	ae(v)
if(log2TrafoSize > 2) {	
if(trafoDepth == 0 || cbf_cb[xBase][yBase][trafoDepth − 1])	
cbf_cb[x0][y0][trafoDepth]	ae(v)
if(trafoDepth == 0 || cbf_cr[xBase][yBase][trafoDepth − 1])	
cbf_cr[x0][y0][trafoDepth]	ae(v)
}	
if(split_transform_flag[x0][y0][trafoDepth]) {	
x1 = x0 + (1 << (log2TrafoSize − 1))	
y1 = y0 + (1 << (log2TrafoSize − 1))	
if(trafoDepth == 0)	
add_multi_transform_flag[x0][y0]	**ae(v)**
transform_tree(x0, y0, x0, y0, log2TrafoSize − 1, trafoDepth + 1, 0)	
transform_tree(x1, y0, x0, y0, log2TrafoSize − 1, trafoDepth + 1, 1)	
transform_tree(x0, y1, x0, y0, log2TrafoSize − 1, trafoDepth + 1, 2)	
transform_tree(x1, y1, x0, y0, log2TrafoSize − 1, trafoDepth + 1, 3)	
} else {	
if(CuPredMode[x0][y0] == MODE_INTRA || trafoDepth != 0 || cbf_cb[x0][y0][trafoDepth] || cbf_cr[x0][y0][trafoDepth])	
cbf_luma[x0][y0][trafoDepth]	ae(v)
if(cbf_luma[x0][y0][trafoDepth] && trafoDepth == 0)	
add_multi_transform_flag[x0][y0]	**ae(v)**
transform_unit(x0, y0, xBase, yBase, log2TrafoSize, trafoDepth, blkIdx)	
}	
}	

В качестве альтернативы add_multi_transform_flag может сигнализироваться без зависимости от cbf_luma.

transform_tree(x0, y0, xBase, yBase, log2TrafoSize, trafoDepth, blkIdx) {	**Descriptor**
if(trafoDepth == 0)	
add_multi_transform_flag[x0][y0]	**ae(v)**
if(log2TrafoSize <= Log2MaxTrafoSize && log2TrafoSize > Log2MinTrafoSize && trafoDepth < MaxTrafoDepth && !(IntraSplitFlag && (trafoDepth == 0)))	
split_transform_flag[x0][y0][trafoDepth]	ae(v)
…	
}	

Это эквивалентно отправке признака в единице кодирования.

coding_unit(x0, y0, log2CbSize) {	**Descriptor**
if(transquant_bypass_enabled_flag)	
cu_transquant_bypass_flag	ae(v)
if(slice_type != I)	
cu_skip_flag[x0][y0]	ae(v)
nCbS = (1 << log2CbSize)	
if(cu_skip_flag[x0][y0])	
prediction_unit(x0, y0, nCbS, nCbS)	
else {	
if(slice_type != I)	
pred_mode_flag	ae(v)
if(CuPredMode[x0][y0] != MODE_INTRA || log2CbSize == MinCbLog2SizeY)	
part_mode	ae(v)
if(CuPredMode[x0][y0] == MODE_INTRA) {	
if(PartMode == PART_2Nx2N && pcm_enabled_flag && log2CbSize >= Log2MinIpcmCbSizeY && log2CbSize <= Log2MaxIpcmCbSizeY)	
pcm_flag[x0][y0]	ae(v)
if(pcm_flag[x0][y0]) {	
while(!byte_aligned( ))	
pcm_alignment_zero_bit	f(1)
pcm_sample(x0, y0, log2CbSize)	
} else {	
pbOffset = (PartMode == PART_NxN) ? (nCbS / 2) : nCbS	
for(j = 0; j < nCbS; j = j + pbOffset)	
for(i = 0; i < nCbS; i = i + pbOffset)	
prev_intra_luma_pred_flag[x0 + i][y0 + j]	ae(v)
for(j = 0; j < nCbS; j = j + pbOffset)	
for(i = 0; i < nCbS; i = i + pbOffset)	
if(prev_intra_luma_pred_flag[x0 + i][y0 + j])	
mpm_idx[x0 + i][y0 + j]	ae(v)
Else	
rem_intra_luma_pred_mode[x0 + i][y0 + j]	ae(v)
intra_chroma_pred_mode[x0][y0]	ae(v)
}	
} else {	
if(PartMode == PART_2Nx2N)	
prediction_unit(x0, y0, nCbS, nCbS)	
else if(PartMode == PART_2NxN) {	
prediction_unit(x0, y0, nCbS, nCbS / 2)	
prediction_unit(x0, y0 + (nCbS / 2), nCbS, nCbS / 2)	
} else if(PartMode == PART_Nx2N) {	
prediction_unit(x0, y0, nCbS / 2, nCbS)	
prediction_unit(x0 + (nCbS / 2), y0, nCbS / 2, nCbS)	
} else if(PartMode == PART_2NxnU) {	
prediction_unit(x0, y0, nCbS, nCbS / 4)	
prediction_unit(x0, y0 + (nCbS / 4), nCbS, nCbS * 3 / 4)	
} else if(PartMode == PART_2NxnD) {	
prediction_unit(x0, y0, nCbS, nCbS * 3 / 4)	
prediction_unit(x0, y0 + (nCbS * 3 / 4), nCbS, nCbS / 4)	
} else if(PartMode == PART_nLx2N) {	
prediction_unit(x0, y0, nCbS / 4, nCbS)	
prediction_unit(x0 + (nCbS / 4), y0, nCbS * 3 / 4, nCbS)	
} else if(PartMode == PART_nRx2N) {	
prediction_unit(x0, y0, nCbS * 3 / 4, nCbS)	
prediction_unit(x0 + (nCbS * 3 / 4), y0, nCbS / 4, nCbS)	
} else { /* PART_NxN */	
prediction_unit(x0, y0, nCbS / 2, nCbS / 2)	
prediction_unit(x0 + (nCbS / 2), y0, nCbS / 2, nCbS / 2)	
prediction_unit(x0, y0 + (nCbS / 2), nCbS / 2, nCbS / 2)	
prediction_unit(x0 + (nCbS / 2), y0 + (nCbS / 2), nCbS / 2, nCbS / 2)	
}	
}	
if(!pcm_flag[x0][y0]) {	
if(CuPredMode[x0][y0] != MODE_INTRA && !(PartMode == PART_2Nx2N && merge_flag[x0][y0]))	
rqt_root_cbf	ae(v)
if(rqt_root_cbf) {	
add_multi_transform_flag	ae(v)
MaxTrafoDepth = (CuPredMode[x0][y0] == MODE_INTRA ? (max_transform_hierarchy_depth_intra + IntraSplitFlag) : max_transform_hierarchy_depth_inter)	
transform_tree(x0, y0, x0, y0, log2CbSize, 0, 0)	
}	
}	
}	
}	

Семантика дерева преобразований add_multi_transform_flag[x0][y0] задает, применяется ли улучшенное множественное преобразование для каждой TU, включенной в текущую CU, когда add multi transform flag[x0][y0] равен 0, всегда применяется DCT-II для каждой TU, включенной в текущую CU, в противном случае могут дополнительно сигнализироваться left_transform_flag и right_transform_flag для каждой TU, чтобы задавать левое преобразование и правое преобразование, применяемые для TU, принадлежащей текущему дереву преобразований. Когда add multi transform_flag[x0][y0] отсутствует, он подразумевается равным 0.

Синтаксис кодирования остатков

residual_coding(x0, y0, log2TrafoSize, cIdx) {	**Descriptor**
if(transform_skip_enabled_flag && !cu_transquant_bypass_flag && (log2TrafoSize == 2))	
transform_skip_flag[x0][y0][cIdx]	ae(v)
last_sig_coeff_x_prefix	ae(v)
last_sig_coeff_y_prefix	ae(v)
if(last_sig_coeff_x_prefix > 3)	
last_sig_coeff_x_suffix	ae(v)
if(last_sig_coeff_y_prefix > 3)	
last_sig_coeff_y_suffix	ae(v)
lastScanPos = 16	
lastSubBlock = (1 << (log2TrafoSize − 2)) * (1 << (log2TrafoSize − 2)) − 1	
do {	
if(lastScanPos == 0) {	
lastScanPos = 16	
lastSubBlock− −	
}	
lastScanPos− −	
xS = ScanOrder[log2TrafoSize − 2][scanIdx][lastSubBlock][0]	
yS = ScanOrder[log2TrafoSize − 2][scanIdx][lastSubBlock][1]	
xC = (xS << 2) + ScanOrder[2][scanIdx][lastScanPos][0]	
yC = (yS << 2) + ScanOrder[2][scanIdx][lastScanPos][1]	
} while((xC != LastSignificantCoeffX) || (yC != LastSignificantCoeffY))	
for(i = lastSubBlock; i >= 0; i− −) {	
xS = ScanOrder[log2TrafoSize − 2][scanIdx][i][0]	
yS = ScanOrder[log2TrafoSize − 2][scanIdx][i][1]	
inferSbDcSigCoeffFlag = 0	
if((i < lastSubBlock) && (i > 0)) {	
coded_sub_block_flag[xS][yS]	ae(v)
inferSbDcSigCoeffFlag = 1	
}	
for(n = (i == lastSubBlock) ? lastScanPos − 1 : 15; n >= 0; n− −) {	
xC = (xS << 2) + ScanOrder[2][scanIdx][n][0]	
yC = (yS << 2) + ScanOrder[2][scanIdx][n][1]	
if(coded_sub_block_flag[xS][yS] && (n > 0 || !inferSbDcSigCoeffFlag)) {	
sig_coeff_flag[xC][yC]	ae(v)
if(sig_coeff_flag[xC][yC])	
inferSbDcSigCoeffFlag = 0	
}	
}	
firstSigScanPos = 16	
lastSigScanPos = −1	
numGreater1Flag = 0	
lastGreater1ScanPos = −1	
for(n = 15; n >= 0; n− −) {	
xC = (xS << 2) + ScanOrder[2][scanIdx][n][0]	
yC = (yS << 2) + ScanOrder[2][scanIdx][n][1]	
if(sig_coeff_flag[xC][yC]) {	
if(numGreater1Flag < 8) {	
coeff_abs_level_greater1_flag[n]	ae(v)
numGreater1Flag++	
if(coeff_abs_level_greater1_flag[n] && lastGreater1ScanPos == −1)	
lastGreater1ScanPos = n	
}	
if(lastSigScanPos == −1)	
lastSigScanPos = n	
firstSigScanPos = n	

}	
}	
signHidden = (lastSigScanPos − firstSigScanPos > 3 && !cu_transquant_bypass_flag)	
if(lastGreater1ScanPos != −1)	
coeff_abs_level_greater2_flag[lastGreater1ScanPos]	ae(v)
for(n = 15; n >= 0; n− −) {	
xC = (xS << 2) + ScanOrder[2][scanIdx][n][0]	
yC = (yS << 2) + ScanOrder[2][scanIdx][n][1]	
if(sig_coeff_flag[xC][yC] && (!sign_data_hiding_enabled_flag || !signHidden || (n != firstSigScanPos)))	
coeff_sign_flag[n]	ae(v)
}	
numSigCoeff = 0	
sumAbsLevel = 0	
for(n = 15; n >= 0; n− −) {	
xC = (xS << 2) + ScanOrder[2][scanIdx][n][0]	
yC = (yS << 2) + ScanOrder[2][scanIdx][n][1]	
if(sig_coeff_flag[xC][yC]) {	
baseLevel = 1 + coeff_abs_level_greater1_flag[n] + coeff_abs_level_greater2_flag[n]	
if(baseLevel == ((numSigCoeff < 8) ? ((n == lastGreater1ScanPos) ? 3 : 2) : 1))	
coeff_abs_level_remaining[n]	ae(v)
TransCoeffLevel[x0][y0][cIdx][xC][yC] = (coeff_abs_level_remaining[n] + baseLevel) * (1 − 2 * coeff_sign_flag[n])	
if(sign_data_hiding_enabled_flag && signHidden) {	
sumAbsLevel += (coeff_abs_level_remaining[n] + baseLevel)	
if((n == firstSigScanPos) && ((sumAbsLevel % 2) == 1))	
TransCoeffLevel[x0][y0][cIdx][xC][yC] = −TransCoeffLevel[x0][y0][cIdx][xC][yC]	
}	
numSigCoeff++	
}	
}	
}	
if(add_multi_transform_flag[x0][y0]) {	
if(numSigCoeff>2 || (CuPredMode[x0][y0] != MODE_INTRA && numSigCoeff>0)) {	
left_transform_flag[x0][y0]	ae(v)
right_transform_flag[x0][y0]	ae(v)
}	
}	
}	

Семантика кодирования остатков

left_transform_f_lag[x0][y0] задает индекс преобразования, применяемый для левого преобразования текущей TU, когда не присутствует, left_transform_flag[x0][y0] подразумевается равным 0.

right_transform_flag[x0][y0] задает индекс преобразования, применяемый для правого преобразования текущей TU, когда не присутствует, right_transform_flag[x0][y0] подразумевается равным 0.

Процесс декодирования для выведения левого преобразования и правого преобразования

Если CuPredMode [x0][y0]==MODE_INTRA, то, учитывая значение внутреннего режима IntraPredModeY[xPb][yPb], значение LeftTransSubsetIdx и RightTransSubsetIdx выводится на основе следующей таблицы:

Внутренний режим	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34
RightTransSubsetIdx	2	1	0	1	0	1	0	1	0	0	0	0	0	1	0	1	0	1	0	1	0	1	0	1	2	2	2	2	2	1	0	1	0	1	0
LeftTransSubsetIdx	2	1	0	1	0	1	0	1	2	2	2	2	2	1	0	1	0	1	0	1	0	1	0	1	0	0	0	0	0	1	0	1	0	1	0

Учитывая значение LeftTransSubsetIdx и left_tranform_flag, левое преобразование выводится с использованием следующей таблицы:

		LeftTransSubsetIdx		
		0	1	2
left_transform_flag	0	DST-7	DST-7	DST-7
	1	DCT-8	DST-1	DCT-5

Учитывая значение RightTransSubsetIdx и right tranform flag, левое преобразование выводится с использованием следующей таблицы:

		RightTransSubsetIdx		
		0	1	2
right_transform_flag	0	DST-7	DST-7	DST-7
	1	DCT-8	DST-1	DCT-5

В противном случае (CuPredMode[x0][y0]!= MODE_INTRA) выполняются следующее:

Учитывая значение left tranform flag, левое преобразование выводится с использованием следующей таблицы:

left_transform_flag	0	DCT-8
	1	DST-7

Учитывая значение right_tranform_flag, правое преобразование выводится с использованием следующей таблицы:

right_transform_flag	0	DCT-8
	1	DST-7

Нижеследующее является примерами применения улучшенного множественного преобразования и большого преобразования. Для процесса кодирования, выполняемого видеокодером 20, в одном примере для каждой CU меньше (либо равной) 32×32 текущая CU кодируется дважды. В первом проходе текущая CU кодируется с использованием только DCT-II. Цена искажения в зависимости от скорости для кодирования всей CU записывается в виде RDcost_CU_DCT; и цена искажения в зависимости от скорости для кодирования каждого режима внутреннего предсказания у PU записывается в виде RDcost_PU_DCT[i][p], где i указывает индекс режима внутреннего предсказания внутри текущей CU, а p указывает индекс PU внутри текущей CU. Оптимальный режим внутреннего предсказания на основе оптимизации искажения в зависимости от скорости для текущей PU, индексированной по p, обозначается в виде IPM[p].

Во втором проходе текущая CU опять кодируется с использованием множественных преобразований, описываемых ниже. Для каждой включенной PU, индексированной по p', в текущей CU выполняется следующее. Для каждого возможного режима i' внутреннего предсказания, если RDcost_PU_DCT [i'] [p']>RDcost_PU_DCT[IPM[p']][p'], то режим i' внутреннего предсказания пропускается и не выбирается в качестве оптимального режима внутреннего предсказания для текущей PU. В противном случае для каждой TU, включенной в текущую PU, выполняется следующее.

Для каждой TU внутри текущей PU, учитывая текущий возможный режим внутреннего предсказания, в соответствии с вышеприведенным примером выбираются 2 возможных правых (R) преобразования и 2 возможных левых (L) преобразования, в целом 4 разных сочетания преобразований R и L. Затем каждое возможное сочетание преобразований R и L проверяется с использованием цены искажения в зависимости от скорости.

Во время этого процесса, если одно сочетание преобразований R и L формирует нулевой коэффициент (например, нулевое значение или отсутствие коэффициентов), то оставшиеся сочетания преобразований L и R пропускаются и не выбираются в качестве оптимального сочетания преобразований R и L. Сочетание преобразований R и L с наименьшей ценой искажения в зависимости от скорости выбирается в качестве фактических преобразований для кодирования текущего остаточного блока. Кроме того, во время вышеупомянутого процесса выбора сочетания преобразований R и L, если одно возможное сочетание преобразований R и L формирует не более 2 ненулевых коэффициентов, то оно не выбирается в качестве оптимального сочетания преобразований, пока оба преобразования R и L не будут преобразованием DST-VII.

После того, как вышеупомянутый процесс выполнен для всех PU внутри текущей CU, цена искажения в зависимости от скорости для кодирования всей CU записывается в виде RDcost_CU_EMT. Если RDcost_CU_DCT меньше RDcost_CU_EMT, то один признак add multi transform flag условно сигнализируется как 0, как описано в вышеприведенном примере, и все включенные TU кодируются с использованием DCT-II. В противном случае add_multi_transform_flag условно сигнализируется как 1, как описано в вышеприведенном примере, и для каждой включенной TU признак left_transform_flag и другой признак right transform flag условно сигнализируются после того, как сигнализируются коэффициенты, как описано в вышеприведенном примере, чтобы указать, какое левое преобразование и правое преобразование выбираются для кодирования текущей TU.

Нижеследующее описывает примеры процесса декодирования, выполняемого видеодекодером 30. В одном примере для каждой CU меньше (либо равной) 32×32 условно сигнализируется одноразрядный признак add_multi_transform_flag, как описано в вышеприведенном примере. Этот признак не сигнализируется, только когда глубина преобразования равна 0, и значение признака кодированного блока (CBF) у компоненты яркости равно 0, в противном случае признак сигнализируется всегда.

Если add multi transform flag равен 0, то только DCT-2 применяется для всех включенных TU, в противном случае выполняется следующее. Для каждой TU одноразрядный признак left_transform_flag и другой признак right_transform_flag условно сигнализируются после того, как сигнализируются коэффициенты, как описано в вышеприведенном примере. Условие того, сигнализируются ли left_transform_flag и right transform flag, описывается ниже.

Когда текущая CU кодируется с внутренним предсказанием, когда общее количество ненулевых коэффициентов меньше (либо равно) 2, left_transform_flag и right transform flag не сигнализируются. В противном случае left_transform_flag и right_transform_flag сигнализируются.

В противном случае, когда текущая CU не кодируется с внутренним предсказанием, когда отсутст-

вует ненулевой коэффициент, left transform flag и right transform flag не сигнализируются. В противном случае left_transform_flag и right transform flag сигнализируются.

Для каждой TU, учитывая сигнализированные left transform flag и right transform flag, левое и правое преобразования выводятся, как описано в вышеприведенном примере. Когда текущая CU больше 32×2, для каждой TU с таким же размером 64×64 преобразование выполняется, как подробнее описано ниже по отношению к примерам для TU большего размера.

Нижеследующее является примерами другого варианта создания списка дополнительных преобразований на основе режима внутреннего предсказания. Кроме способа преобразования по умолчанию, который всегда применяет DCT-II для всех включенных TU, для каждой TU дополнительное возможное преобразование DST-VII может применяться следующим образом. Учитывая режим внутреннего предсказания у текущей TU, обозначенный как IPM, левое и правое преобразование, применяемое к этой TU, задается как представлено ниже. Если (IPM & 1) равно 1, то DCT-II применяется как левое и правое преобразование для текущей TU; в противном случае ((IPM &1) равно 0) DST-VII применяется как левое и правое преобразование для текущей TU.

Вышеприведенное описывает примерные методики для определения преобразований для использования. Нижеследующее описывает примеры для поддержки преобразования большего размера. Например, поддержка преобразования 64×64 выгодна, в особенности для кодирования видео с большими разрешениями, например 1080p и 4K. Поддержка преобразования 64×64, ограничивая при этом сложность для видеокодера 20 и видеодекодер 30, является важной, и хотя для достижения этого есть различные способы, может быть доступно решение лучше.

Фактические применяемые матрицы преобразований N×N могут быть целочисленным приближением после масштабирования исходных матриц преобразований с плавающей запятой, и масштабирование может быть больше $64*\log_2 N$, включая, но не только, $s*\log_2 N$, где s может быть 128 или 256. В одном примере результирующие коэффициенты преобразования после горизонтального и вертикального преобразований остаются в 16-разрядном представлении путем применения дополнительных операций сдвига вправо. Дополнительные операции сдвига вправо включают в себя, но не ограничиваются, сдвиг вправо результирующих коэффициентов преобразования после вертикального и горизонтального обратного/прямого преобразования на дополнительные $\log_2(s/64)$ разрядов.

К остаточным блокам могут применяться размеры преобразования больше 32-точечного преобразования, включая, но не только, 64-точечное, 128-точечное, 256-точечное. Когда M-точечное и N-точечное преобразования применяются для горизонтального и вертикального преобразований соответственно, где M и N - целые числа, и M могло бы быть таким же или отличаться от N, сигнализируются только верхние левые X×Y низкочастотных коэффициентов внутри результирующего блока коэффициентов M×N, где X<M и Y<N, а оставшиеся коэффициенты не сигнализируются и выводятся как 0.

Положение последнего ненулевого коэффициента внутри результирующего блока коэффициентов M×N может кодироваться путем повторного использования той же логики кодирования положения последнего ненулевого коэффициента, используемой для блоков S×T, где X≤S≤M и Y≤T≤N, с точки зрения моделирования контекста. Чтобы обнулить коэффициенты за пределами X×Y, можно ввести ограничение для LastSignificantCoeffY и LastSignificantCoef fX. Например, значение LastSignificantCoeffX (LastSignificantCoeffY) может быть меньше X (Y). Значение X и Y может быть постоянным, например 32, или зависеть от размера преобразования, например, X=M/2, Y=N/2. Типичным значением X и Y является 32 для 64-точечного преобразования.

В одном примере может предопределяться и использоваться один или несколько порядков сканирования CG (группы кодирования) для M×N. Однако для CG вне верхней левой области X×Y в результирующем блоке коэффициентов M×N пропускаются и не кодируются признаки CG-уровня, сигнализирующие, имеется ли по меньшей мере один ненулевой коэффициент в каждой CG. В качестве альтернативы или дополнительно для всех CG W×H, где типичным значением W и H является 4, CG кодируются, придерживая Cb порядка сканирования для областей X×Y, который включает в себя, но не только, диагональный (M/W)×(N/H), зигзагообразный, горизонтальный или вертикальный порядок сканирования. В качестве альтернативы или дополнительно все CG группируются в единицы W'×H', где W кратное W, а H' - кратное H, единицы W'×H' кодируются, придерживая Cb порядка сканирования, включающего в себя, но не только, диагональный (M/W') × (N/H'), зигзагообразный, горизонтальный или вертикальный порядок сканирования, и CG внутри каждой единицы W×H кодируются с порядком сканирования, включающим в себя, но не только, диагональный (W'/W) × (H'/H), зигзагообразный, горизонтальный или вертикальный порядок сканирования.

Для поддержки кодирования глубины RQT, соответствующей размеру преобразования 64×64, 128×128 или 256×256, контексты CABAC, используемые для кодирования признаков разбиения RQT, которые зависят от размеров преобразования, могут совместно использоваться для значений глубины RQT, соответствующих размеру преобразования больше 32×32. Например, для некоторых случаев глубины RQT, включая, но не только, значения глубины RQT, соответствующие размеру 64×64 и 32×32

преобразования, один и тот же контекст CABAC может применяться для кодирования признака разбиения RQT.

Нижеследующее является примерами выполнения обнуления 32x32 для преобразования 64×64. Может быть ограничение для последнего положения, где last_sig_coeff_x_suffix задает суффикс положения столбца последнего значимого коэффициента в порядке сканирования в блоке преобразований. Значения last_sig_coeff_x_suffix должны быть в диапазоне от 0 до (1<<((last_sig_coeff_x_prefix >>1)-1))-1 включительно. Положение столбца последнего значимого коэффициента в порядке сканирования в блоке преобразований, LastSignificantCoeffX, выводится следующим образом. Если last_sig_coeff_x_suffix отсутствует, то применяется следующее: LastSignificantCoeffX=last_sig_coeff_x_prefix, в противном случае (last_sig_coeff_x suffix присутствует) применяется следующее LastSignificantCoeffX=(1<<((last_sig_coeff_xprefix>>1)-1))*(2+(last_sig_coeff_x_prefix&1))+last_sig_coeff_x_suffix.

Синтаксический элемент last sig coeff y suffix задает суффикс положения строки последнего значимого коэффициента в порядке сканирования в блоке преобразований. Значения last_sig_coeff_ysuffix должны быть в диапазоне от 0 до (1«((last_sig_coeff_y_prefix»1)-1))-1 включительно. Положение строки y последнего значимого коэффициента в порядке сканирования в блоке преобразований. LastSignificant CoeffY выводится следующим образом. Если last sig coeff y suffix отсутствует, то применяется следующее LastSignificantCoeffY=last_sig_coeff_y_prefix, в противном случае (last_sig_coeff_ysuffix присутствует) применяется следующее LastSignificantCoeffY=(1<<((last_sig_coeff_y_prefix»1)-1))*(2+(last_sig_coeff_y_prefix&1))+last_sig_coeff_y_suffix.

Когда scanIdx равен 2, координаты меняются следующим образом (LastSignificantCoeffX, LastSignificantCoeffY)=Swap(LastSignificantCoeffX, LastSignificantCoeffY). Значение LastSignificantCoeffX или LastSignificantCoeffY должно быть меньше 32.

Нижеследующее является условиями для сигнализации значимой CG и значимых коэффициентов.

Синтаксис кодирования остатков

residual_coding(x0, y0, log2TrafoSize, cIdx) {	**Descriptor**
if(transform_skip_enabled_flag && !cu_transquant_bypass_flag && (log2TrafoSize == 2))	
transform_skip_flag[x0][y0][cIdx]	ae(v)
last_sig_coeff_x_prefix	ae(v)
last_sig_coeff_y_prefix	ae(v)
if(last_sig_coeff_x_prefix > 3)	
last_sig_coeff_x_suffix	ae(v)
if(last_sig_coeff_y_prefix > 3)	
last_sig_coeff_y_suffix	ae(v)
lastScanPos = 16	
lastSubBlock = (1 << (log2TrafoSize − 2)) * (1 << (log2TrafoSize − 2)) − 1	
do {	
if(lastScanPos == 0) {	
lastScanPos = 16	
lastSubBlock− −	
}	
lastScanPos− −	
xS = ScanOrder[log2TrafoSize − 2][scanIdx][lastSubBlock][0]	
yS = ScanOrder[log2TrafoSize − 2][scanIdx][lastSubBlock][1]	

xC = (xS << 2) + ScanOrder[2][scanIdx][lastScanPos][0]	
yC = (yS << 2) + ScanOrder[2][scanIdx][lastScanPos][1]	
} while((xC != LastSignificantCoeffX) || (yC != LastSignificantCoeffY))	
for(i = lastSubBlock; i >= 0; i− −) {	
xS = ScanOrder[log2TrafoSize − 2][scanIdx][i][0]	
yS = ScanOrder[log2TrafoSize − 2][scanIdx][i][1]	
inferSbDcSigCoeffFlag = 0	
if((i < lastSubBlock) && (i > 0) && (xS<32 || yS<32)) {	
coded_sub_block_flag[xS][yS]	ae(v)
inferSbDcSigCoeffFlag = 1	
}	
for(n = (i == lastSubBlock) ? lastScanPos − 1 : 15; n >= 0; n− −) {	
xC = (xS << 2) + ScanOrder[2][scanIdx][n][0]	
yC = (yS << 2) + ScanOrder[2][scanIdx][n][1]	
if(coded_sub_block_flag[xS][yS] && (n > 0 || !inferSbDcSigCoeffFlag)) {	
sig_coeff_flag[xC][yC]	ae(v)
if(sig_coeff_flag[xC][yC])	
inferSbDcSigCoeffFlag = 0	
}	
}	
firstSigScanPos = 16	
lastSigScanPos = −1	
numGreater1Flag = 0	
lastGreater1ScanPos = −1	
for(n = 15; n >= 0; n− −) {	
xC = (xS << 2) + ScanOrder[2][scanIdx][n][0]	
yC = (yS << 2) + ScanOrder[2][scanIdx][n][1]	
if(sig_coeff_flag[xC][yC]) {	
if(numGreater1Flag < 8) {	
coeff_abs_level_greater1_flag[n]	ae(v)
numGreater1Flag++	
if(coeff_abs_level_greater1_flag[n] && lastGreater1ScanPos == −1)	
lastGreater1ScanPos = n	
}	
if(lastSigScanPos == −1)	
lastSigScanPos = n	
firstSigScanPos = n	
}	

}	
signHidden = (lastSigScanPos − firstSigScanPos > 3 && !cu_transquant_bypass_flag)	
if(lastGreater1ScanPos != −1)	
coeff_abs_level_greater2_flag[lastGreater1ScanPos]	ae(v)
for(n = 15; n >= 0; n− −) {	
xC = (xS << 2) + ScanOrder[2][scanIdx][n][0]	
yC = (yS << 2) + ScanOrder[2][scanIdx][n][1]	
if(sig_coeff_flag[xC][yC] && (!sign_data_hiding_enabled_flag || !signHidden || (n != firstSigScanPos)))	
coeff_sign_flag[n]	ae(v)
}	
numSigCoeff = 0	
sumAbsLevel = 0	
for(n = 15; n >= 0; n− −) {	
xC = (xS << 2) + ScanOrder[2][scanIdx][n][0]	
yC = (yS << 2) + ScanOrder[2][scanIdx][n][1]	
if(sig_coeff_flag[xC][yC]) {	
baseLevel = 1 + coeff_abs_level_greater1_flag[n] + coeff_abs_level_greater2_flag[n]	
if(baseLevel == ((numSigCoeff < 8) ? ((n == lastGreater1ScanPos) ? 3 : 2) : 1))	
coeff_abs_level_remaining[n]	ae(v)
TransCoeffLevel[x0][y0][cIdx][xC][yC] = (coeff_abs_level_remaining[n] + baseLevel) * (1 − 2 * coeff_sign_flag[n])	
if(sign_data_hiding_enabled_flag && signHidden) {	
sumAbsLevel += (coeff_abs_level_remaining[n] + baseLevel)	
if((n == firstSigScanPos) && ((sumAbsLevel % 2) == 1))	
TransCoeffLevel[x0][y0][cIdx][xC][yC] = −TransCoeffLevel[x0][y0][cIdx][xC][yC]	
}	
numSigCoeff++	
}	
}	
}	
}	

Нижеследующее описывает пример блочной компенсации движения с перекрытием (OBMC). OBMC была предложена при разработке H.263 IT-UT. См. Video Coding for Low Bitrate Communication, документ Rec. H.263, ITU-T, апрель 1995 г. OBMC выполняется над блоком 8×8, и векторы движения двух соединенных соседних блоков 8×8 используются для текущего блока, как показано на фиг. 8A и 8B. Например, для первого блока 8×8 в текущем макроблоке, кроме своего вектора движения, для формирования двух дополнительных блоков предсказания также применяется верхний и левый соседний вектор движения. Таким образом, каждый пиксель в текущем блоке 8×8 имеет три значения предсказания, и среднее взвешенное этих трех значений предсказания используется в качестве окончательного предсказания для соответствующего пикселя.

Когда соседний блок не кодируется или кодируется как внутренний, то есть соседний блок не имеет доступного вектора движения, вектор движения текущего блока 8x8 используется в качестве соседнего вектора движения. Между тем для третьего и четвертого блока 8×8 в текущем макроблоке (как показано на фиг. 7) нижний соседний блок можно не использовать (например, всегда не использовать или использовать не всегда). Другими словами, в некоторых примерах для каждого MB никакая информация о движении из MB ниже его не будет использоваться для восстановления пикселей текущего MB во время OBMC.

Нижеследующее описывает OBMC, которая предложена в HEVC. В HEVC OBMC также была предложена для сглаживания границы PU в предварительной заявке США № 61/561783, поданной 18 ноября 2011 г., заявке США № 13/678329, поданной 15 ноября 2012 г., предварительной заявке США № 61/431480, поданной 10 января 2011 г., предварительной заявке США № 61/450532, поданной 8 марта 2011 г., и заявке США № 13/311834, поданной 6 декабря 2011 г. Пример способа, предложенного в HEVC, показан на фиг. 8A и 8B, где белая область является первой единицей предсказания (PU), обозначенной PU0, а серая область является второй PU, обозначенной PU1. Когда CU содержит две (или более) PU, линии/столбцы возле границы PU сглаживаются с помощью OBMC. Для пикселей, помеченных "A" или "B" в PU0 или PU1, формируются два значения предсказания, то есть путем применения векторов движения у PU0 и PU1 соответственно, и их среднее взвешенное используется в качестве окончательного предсказания.

Кроме того, в предварительной заявке США № 62/107964, поданной 26 января 2015 г., и предварительной заявке США № 62/116631, поданной 16 февраля 2015 г., предложен признак CU-уровня, указывающий, применяется ли OBMC для текущей CU, а именно - признак OBMC.

Было отмечено, что когда OBMC не применяется к одной единице кодирования (например, сигнализированным признаком является 0), не эффективны преобразования кроме DCT-II. Поэтому дополнительная сигнализация для указания использования множественных преобразований избыточна.

Как описано, видеокодер 20 может сигнализировать (например, формировать в потоке двоичных сигналов) признак CU-уровня для указания, задействована ли OBMC для текущей CU. В некоторых примерах, когда этот признак OBMC сигнализирован как 1 (указывая, что OBMC задействована для текущей CU), только DCT-II по умолчанию используется для каждой TU, и поэтому видеокодер 20 может не сигнализировать ничего для выбора преобразования, то есть не сигнализируется ни признак CU-уровня, ни индекс TU-уровня.

Нижеследующее описывает примеры оптимизации видеокодера 20. Например, следующие примеры могут быть предназначены для видеокодера 20. Однако видеодекодер 30 может выполнять аналогичные методики.

В кодере (например, видеокодере 20), когда предложенные множественные преобразования применяются для текущей TU, для размеров преобразования больше либо равных MxN вычисляются только низкочастотные коэффициенты M'×N', а другие коэффициенты устанавливаются в 0, где (M' <= M, и N' <= N, и M'*N' < M*N). В одном примере значение каждого из M и N равно 32, а значение каждого из M' и N' равно 16. В этом примере коэффициенты, расположенные в местоположениях больше M' и/или больше N', могут считаться высокочастотными коэффициентами. Вообще, коэффициенты ближе к правой части TU и ближе к нижней части TU могут считаться высокочастотными коэффициентами.

В кодере для некоторого режима кодирования, если цена кодирования с использованием преобразования по умолчанию, например DCT-II, больше текущей наименьшей цены кодирования, умноженной на заданное пороговое значение, то предложенные множественные преобразования пропускаются. Цена кодирования может быть ценой искажения в зависимости от скорости, суммой абсолютного остатка предсказания, суммой квадрата остатка предсказания или суммой абсолютной разности преобразований. Пороговое значение может зависеть от размера блока кодирования. В одном примере значением пороговой величины является 1,1.

В кодере для некоторого направления внутреннего предсказания, если цена кодирования с использованием преобразования по умолчанию, например DCT-II, больше цены кодирования у лучшего направления внутреннего предсказания, умноженной на заданное пороговое значение, то предложенные множественные преобразования не применяются и пропускаются для этого режима внутреннего предсказания. Цена кодирования может быть ценой искажения в зависимости от скорости, суммой абсолютного остатка предсказания, суммой квадрата остатка предсказания или суммой абсолютной разности

преобразований. Пороговые значения могут зависеть от размера блока кодирования. В одном примере значением для пороговых величин являются 1,47, 1,28, 1,12 и 1,06 для размеров блоков 4×4, 8×8, 16×16 и 32×32 соответственно.

В кодере, если цена кодирования у внутреннего разбиения PU N×N с использованием преобразования по умолчанию, например DCT-II, больше цены кодирования у внутреннего разбиения PU 2N×2N, умноженной на заданное пороговое значение, то предложенные множественные преобразования не применяются и пропускаются для внутреннего разбиения PU N×N. Цена кодирования может быть ценой искажения в зависимости от скорости, суммой абсолютного остатка предсказания, суммой квадрата остатка предсказания или суммой абсолютной разности преобразований. Пороговые значения могут зависеть от размера блока кодирования. В одном примере значением для пороговой величины является 1,2.

В кодере, если цена кодирования в режиме внутреннего разбиения PU 2N×2N с использованием преобразования по умолчанию, например DCT-II, больше цены кодирования в лучших режимах внутреннего кодирования, умноженной на заданное пороговое значение, то предложенные множественные преобразования не применяются и пропускаются для внутренних режимов PU. Цена кодирования может быть ценой искажения в зависимости от скорости, суммой абсолютного остатка предсказания, суммой квадрата остатка предсказания или суммой абсолютной разности преобразований. Пороговые значения могут зависеть от размера блока кодирования. В одном примере значением для пороговой величины является 1,4.

В кодере, если использование одного из вариантов множественных преобразований формирует все нулевые коэффициенты для текущего блока, то оставшиеся варианты преобразований не применяются и пропускаются для текущего блока. В качестве альтернативы или дополнительно, если использование преобразования по умолчанию (например, DCT-II) формирует все нулевые коэффициенты для текущего блока, то варианты множественных преобразований не применяются и пропускаются для текущего блока, и для кодирования текущего блока используется только преобразование по умолчанию (например, DCT-II).

В кодере, когда признак OBMC сигнализирован и указывает, что OBMC отключена, одноразрядный признак, указывающий, применяется ли только одно преобразование по умолчанию, по-прежнему сигнализируется в виде значения по умолчанию (например, 0), которое указывает, что применяется преобразование по умолчанию (например, DCT-II), и варианты множественных преобразований не применяются и пропускаются для текущего блока.

Фиг. 9 - блок-схема алгоритма, иллюстрирующая примерный способ декодирования видеоданных. Видеодекодер 30 может определить множество поднаборов преобразований, при этом каждый поднабор идентифицирует одно или несколько возможных преобразований, где по меньшей мере один поднабор преобразований идентифицирует множество возможных преобразований (200).

Например, модуль 152 обработки с предсказанием может извлечь множество поднаборов преобразований из поднаборов преобразований, сохраненных в памяти 151 видеоданных. Множество поднаборов преобразований может заранее сохраняться в памяти 151 видеоданных, либо от видеокодера 20 может приниматься информация, идентифицирующая, как создавать поднаборы преобразований. Видеодекодер 30 может выбрать первый поднабор преобразований из множества поднаборов преобразований для левого преобразования для текущего блока коэффициентов видеоданных и выбрать второй поднабор преобразований из множества поднаборов преобразований для правого преобразования для текущего блока коэффициентов видеоданных (202). Например, модуль 152 обработки с предсказанием может, в качестве нескольких примерных способов определения поднаборов преобразований, выбрать первый и второй поднаборы преобразований на основе информации о режиме внутреннего предсказания, сигнализированной в потоке двоичных сигналов видео, или на основе положения декодируемого видеоблока.

Видеодекодер 30 может определить левое преобразование из выбранного первого поднабора преобразований и определить правое преобразование из выбранного второго поднабора преобразований (204). Например, модуль 152 обработки с предсказанием может принять информацию в потоке двоичных сигналов, например индексы к выбранным поднаборам преобразований, или может неявно определить преобразования на основе количества ненулевых коэффициентов. Видеодекодер 30 может определить текущий блок преобразований на основе левого преобразования, правого преобразования и текущего блока коэффициентов (206). Например, модуль 156 обработки с обратным преобразованием может определить текущий блок преобразований путем применения левого преобразования и правого преобразования к блоку коэффициентов, выведенному модулем 154 обратного квантования. Видеодекодер 30 может восстановить (например, декодировать с внутренним предсказанием или межкадровым предсказанием) видеоблок на основе текущего блока преобразований и блока с предсказанием (208). Например, модуль 158 восстановления может сложить текущий блок преобразований (который является остатком между видеоблоком и блоком с предсказанием) с блоком с предсказанием, чтобы восстановить видеоблок.

Фиг. 10 - блок-схема алгоритма, иллюстрирующая примерный способ кодирования видеоданных. Видеокодер 20 может определить множество поднаборов преобразований, при этом каждый поднабор идентифицирует одно или несколько возможных преобразований, где по меньшей мере один поднабор

преобразований идентифицирует множество возможных преобразований (300). Например, модуль 100 обработки с предсказанием может извлечь множество поднаборов преобразований из поднаборов преобразований, сохраненных в памяти 101 видеоданных. Множество поднаборов преобразований может заранее сохраняться в памяти 101 видеоданных. Видеокодер 20 может выбрать первый поднабор преобразований из множества поднаборов преобразований для левого преобразования для текущего блока преобразований в видеоблоке видеоданных и выбрать второй поднабор преобразований из множества поднаборов преобразований для правого преобразования для блока преобразований в видеоблоке видеоданных (302). Например, модуль 100 обработки с предсказанием может, в качестве нескольких примерных способов определения поднаборов преобразований, выбрать первый и второй поднаборы преобразований на основе информации о режиме внутреннего предсказания, которую модуль 118 энтропийного кодирования формирует в потоке двоичных сигналов видео, или на основе положения кодируемого видеоблока.

Видеокодер 20 может определить левое преобразование из выбранного первого поднабора преобразований и определить правое преобразование из выбранного второго поднабора преобразований (304). Например, модуль 100 обработки с предсказанием может проверить различные определенные преобразования, чтобы идентифицировать преобразование, которое обеспечивает хорошее качество кодирования видео. Видеокодер 20 может определить текущий блок коэффициентов на основе левого преобразования, правого преобразования и текущего блока преобразований (306). Например, модуль 104 обработки с преобразованием может определить текущий блок коэффициентов путем применения левого преобразования и правого преобразования к блоку преобразований, выведенному модулем 102 восстановления. Видеокодер 20 может сформировать поток двоичных сигналов видео с информацией (например, сигнализировать информацию), указывающей коэффициенты текущего блока коэффициентов, используемые для восстановления видеоблока (308). Например, модуль 118 энтропийного кодирования может вывести информацию, которую видеодекодер 30 использует для восстановления видеоблока.

Следует понимать, что все описанные в этом документе методики можно использовать по отдельности или вместе. Данное раскрытие изобретения включает в себя несколько способов сигнализации, которые могут меняться в зависимости от некоторых факторов, например размера блока, типа секции и т.п. Такое изменение сигнализации или выведения синтаксических элементов может быть заранее известно кодеру и декодеру или может сигнализироваться явно в наборе параметров видео (VPS), наборе параметров последовательности (SPS), наборе параметров изображения (PPS), заголовке секции, на уровне фрагмента или где-либо еще.

Нужно признать, что в зависимости от примера некоторые действия или события в любой из описанных в этом документе методик можно выполнять в иной последовательности, можно добавлять, объединять или полностью пропускать (например, не все описанные действия или события необходимы для применения методик на практике). Кроме того, в некоторых примерах действия или события могут выполняться одновременно, например, посредством многопоточной обработки, обработки прерываний или нескольких процессоров, а не последовательно. К тому же, хотя некоторые аспекты данного раскрытия изобретения описываются для ясности как выполняемые одним модулем или блоком, следует понимать, что методики в данном раскрытии изобретения могут выполняться сочетанием блоков или модулей, ассоциированных с кодировщиком видео.

Хотя выше описываются конкретные сочетания различных аспектов методик, эти сочетания предоставляются всего лишь для иллюстрации примеров методик, описанных в данном раскрытии изобретения. Соответственно, методики в данном раскрытии изобретения не следует ограничивать этими примерными сочетаниями, и они могут включать в себя любое возможное сочетание различных аспектов методик, описанных в данном раскрытии изобретения.

В одном или нескольких примерах описываемые функции могут быть реализованы в аппаратных средствах, программном обеспечении, микропрограммном обеспечении или любом их сочетании. Если они реализованы в программном обеспечении, то функции могут храниться или передаваться в виде одной или нескольких команд либо кода на машиночитаемом носителе и исполняться аппаратным модулем обработки. Машиночитаемые носители могут включать в себя машиночитаемые носители информации, которые соответствуют материальному носителю, например носителям информации. Таким образом, машиночитаемые носители в целом могут соответствовать материальным машиночитаемым носителям информации, которые не меняются со временем (постоянны). Носители информации могут быть любыми доступными носителями, к которым можно обращаться с помощью одного или нескольких компьютеров либо одного или нескольких процессоров для извлечения команд, кода и/или структур данных для реализации методик, описанных в данном раскрытии изобретения. Компьютерный программный продукт может включать в себя машиночитаемый носитель. В качестве примера, а не ограничения, такие машиночитаемые носители информации могут быть выполнены в виде RAM, ROM, EEPROM, компакт-диска или другого накопителя на оптических дисках, накопителя на магнитных дисках или других магнитных запоминающих устройств, флэш-памяти или любого другого носителя, который может использоваться для хранения нужного программного кода в виде команд или структур данных и к которому [носителю] можно обращаться с помощью компьютера. Однако следует понимать, что машиночитаемые носители информации и носители информации не включают в себя соединения, несущие, сигналы или другие

кратковременные носители, а вместо этого ориентированы на долговременные, материальные носители информации. Диски при использовании в данном документе включают в себя компакт-диск (CD), лазерный диск, оптический диск, цифровой универсальный диск (DVD), гибкий диск и диск Blu-ray, где диски (disks) обычно воспроизводят данные магнитным способом, тогда как диски (discs) воспроизводят данные оптически с помощью лазеров. Сочетания вышеперечисленного также следует включить в область машиночитаемых носителей.

Команды могут исполняться одним или несколькими процессорами, например одним или несколькими цифровыми процессорами сигналов (DSP), универсальными микропроцессорами, специализированными интегральными схемами (ASIC), программируемыми пользователем логическими матрицами (FPGA) или другими эквивалентными интегральными либо дискретными логическими схемами. Соответственно, термин "процессор" при использовании в данном документе может относиться к любой вышеупомянутой структуре или к любой другой структуре, подходящей для реализации описанных в этом документе методик. К тому же в некоторых аспектах функциональные возможности, описанные в этом документе, могут быть предоставлены в специализированных аппаратных и/или программных модулях, сконфигурированных для кодирования и декодирования, или встроены в объединенный кодек. Также методики можно было бы полностью реализовать в одной или нескольких схемах или логических элементах.

Методики из данного раскрытия изобретения могут быть реализованы в широком спектре устройств, включая беспроводную телефонную трубку, интегральную схему (ИС) или набор ИС (например, набор микросхем). Различные компоненты, модули или блоки описываются в данном раскрытии изобретения для подчеркивания функциональных аспектов устройств, сконфигурированных для выполнения раскрытых методик, но не обязательно требуют реализации с помощью разных аппаратных модулей. Точнее, как описано выше, различные модули могут объединяться в аппаратный модуль кодека или предоставляться с помощью совокупности взаимодействующих аппаратных модулей, включающих в себя один или несколько процессоров, как описано выше, в сочетании с подходящим программным обеспечением и/или микропрограммным обеспечением.

Описаны различные примеры. Эти и другие примеры входят в объем нижеследующей формулы изобретения.

ФОРМУЛА ИЗОБРЕТЕНИЯ

1. Способ HEVC декодирования видеоданных, содержащий этапы, на которых

определяют множество поднаборов преобразований для преобразования текущего блока коэффициентов в видеоблоке, кодированном в соответствии с одним из множества режимов предсказания, при этом режимы предсказания содержат множество режимов внутреннего предсказания и множество режимов межкадрового предсказания, каждый поднабор идентифицирует одно или несколько возможных преобразований, где по меньшей мере один поднабор преобразований идентифицирует множество возможных преобразований, где этап, на котором определяют множество поднаборов преобразований, содержит этап, на котором определяют

либо первый один поднабор преобразований из множества поднаборов преобразований, причем этот первый поднабор преобразований является таким же для каждого из режимов внутреннего предсказания;

либо второй один поднабор преобразований из множества поднаборов преобразований, причем этот второй поднабор преобразований является таким же для каждого из режимов межкадрового предсказания;

выбирают первый поднабор преобразований из определенного множества поднаборов преобразований для левого преобразования для текущего блока коэффициентов видеоданных;

выбирают второй поднабор преобразований из определенного множества поднаборов преобразований для правого преобразования для текущего блока коэффициентов видеоданных;

определяют левое преобразование из выбранного первого поднабора преобразований;

определяют правое преобразование из выбранного второго поднабора преобразований;

определяют текущий блок преобразований на основе левого преобразования, правого преобразования и текущего блока коэффициентов; и

восстанавливают видеоблок на основе текущего блока преобразований и блока с предсказанием.

2. Способ HEVC кодирования видеоданных, содержащий этапы, на которых

определяют множество поднаборов преобразований для преобразования текущего блока преобразований в видеоблоке, кодированном в соответствии с одним из множества режимов предсказания, при этом режимы предсказания содержат множество режимов внутреннего предсказания и множество режимов межкадрового предсказания, каждый поднабор идентифицирует одно или несколько возможных преобразований, где по меньшей мере один поднабор преобразований идентифицирует множество возможных преобразований, где этап, на котором определяют множество поднаборов преобразований, содержит этап, на котором определяют

либо первый один поднабор преобразований из множества поднаборов преобразований, причем

этот первый поднабор преобразований является таким же для каждого из режимов внутреннего предсказания;

либо второй один поднабор преобразований из множества поднаборов преобразований, причем этот второй поднабор

преобразований является таким же для каждого из режимов межкадрового предсказания;

выбирают первый поднабор преобразований из определенного множества поднаборов преобразований для левого преобразования для текущего блока преобразований в видеоблоке видеоданных;

выбирают второй поднабор преобразований из определенного множества поднаборов преобразований для правого преобразования для текущего блока преобразований в видеоблоке видеоданных;

определяют левое преобразование из выбранного первого поднабора преобразований;

определяют правое преобразование из выбранного второго поднабора преобразований;

определяют текущий блок коэффициентов на основе левого преобразования, правого преобразования и текущего блока преобразований; и

формируют поток двоичных сигналов видео, который включает в себя информацию, указывающую коэффициенты текущего блока коэффициентов, используемые для восстановления видеоблока.

3. Способ по п.1 или 2, дополнительно содержащий этап, на котором

определяют режим внутреннего предсказания видеоблока, где этап, на котором выбирают первый поднабор преобразований, содержит этап, на котором выбирают первый поднабор преобразований на основе определенного режима внутреннего предсказания, и

где этап, на котором выбирают второй поднабор преобразований, содержит этап, на котором выбирают второй поднабор преобразований на основе определенного режима внутреннего предсказания.

4. Способ по п.1 или 2, дополнительно содержащий этап, на котором

определяют местоположение текущего блока преобразований в видеоблоке на основе видеоблока, кодированного с межкадровым предсказанием,

где этап, на котором выбирают первый поднабор преобразований, содержит этап, на котором выбирают первый поднабор преобразований на основе определенного местоположения текущего блока преобразований, и

где этап, на котором выбирают второй поднабор преобразований, содержит этап, на котором выбирают второй поднабор преобразований на основе определенного местоположения текущего блока преобразований.

5. Способ по п.1 или 2, дополнительно содержащий этапы, на которых

формируют в потоке двоичных сигналов видео индекс первого поднабора преобразований к первому поднабору преобразований, чтобы идентифицировать преобразование в первом поднаборе преобразований, используемое для определения текущего блока коэффициентов; и

формируют в потоке двоичных сигналов видео индекс второго поднабора преобразований к второму поднабору преобразований, чтобы идентифицировать преобразование во втором поднаборе преобразований, используемое для определения текущего блока коэффициентов, предпочтительно способ дополнительно содержит этап, на котором

определяют количество ненулевых коэффициентов в текущем блоке коэффициентов,

где этап, на котором сигнализируют индекс первого поднабора преобразований, содержит этап, на котором сигнализируют индекс первого поднабора преобразований на основе количества ненулевых коэффициентов, которое больше пороговой величины, и

где этап, на котором сигнализируют индекс второго поднабора преобразований, содержит этап, на котором сигнализируют индекс второго поднабора преобразований на основе количества ненулевых коэффициентов, которое больше пороговой величины.

6. Способ по п.1 или 2, в котором по меньшей мере одно из первого поднабора преобразований или второго поднабора преобразований включает в себя преобразование, которое отличается от дискретного косинусного преобразования (DCT)-II и дискретного синусного преобразования (DST)-VII.

7. Способ по п.1 или 2, в котором первый поднабор преобразований и второй поднабор преобразований включают в себя разные типы преобразований.

8. Способ по п.1 или 2, в котором множество поднаборов преобразований содержит три или более поднаборов преобразований.

9. Способ по п.1 или 2, в котором возможные преобразования являются разными типами преобразований.

10. Способ по п.1 или 2, в котором этап, на котором определяют множество поднаборов преобразований, содержит этап, на котором определяют множество поднаборов преобразований на основе размера видеоблока.

11. Способ по п.1 или 2, дополнительно содержащий этапы, на которых

определяют блок с предсказанием; и

на основе блока с предсказанием формируют в потоке двоичных сигналов видео информацию, указывающую режим предсказания видеоблока,

причем блок с предсказанием предпочтительно является блоком в том же изображении, что и ви-

деоблок на основе видеоблока с режимом внутреннего предсказания, либо в изображении, отличном от изображения, которое включает в себя видеоблок на основе видеоблока с межкадровым предсказанием.

12. Способ по п.1 или 2, дополнительно содержащий этап, на котором

определяют текущий блок преобразований как остаток между видеоблоком и блоком с предсказанием.

13. Устройство для HEVC декодирования видеоданных, содержащее

средство для определения множества поднаборов преобразований для преобразования текущего блока коэффициентов в видеоблоке, кодированном в соответствии с одним из множества режимов предсказания, при этом режимы предсказания содержат множество режимов внутреннего предсказания и множество режимов межкадрового предсказания, каждый поднабор идентифицирует одно или несколько возможных преобразований, где по меньшей мере один поднабор преобразований идентифицирует множество возможных преобразований, где средство для определения множества поднаборов преобразований содержит средство для определения

либо первого одного поднабора преобразований из множества поднаборов преобразований, причем этот первый поднабор преобразований является таким же для каждого из режимов внутреннего предсказания;

либо второго одного поднабора преобразований из множества поднаборов преобразований, причем этот второй поднабор преобразований является таким же для каждого из режимов межкадрового предсказания;

средство для выбора первого поднабора преобразований из определенного множества поднаборов преобразований для левого преобразования для текущего блока коэффициентов видеоданных;

средство для выбора второго поднабора преобразований из определенного множества поднаборов преобразований для правого преобразования для текущего блока коэффициентов видеоданных;

средство для определения левого преобразования из выбранного первого поднабора преобразований;

средство для определения правого преобразования из выбранного второго поднабора преобразований;

средство для определения текущего блока преобразований на основе левого преобразования, правого преобразования и текущего блока коэффициентов; и

средство для восстановления видеоблока на основе текущего блока преобразований и блока с предсказанием.

14. Устройство для HEVC кодирования видеоданных, содержащее:

средство для определения множества поднаборов преобразований для преобразования текущего блока преобразований в видеоблоке, кодированном в соответствии с одним из множества режимов предсказания, при этом режимы предсказания содержат множество режимов внутреннего предсказания и множество режимов межкадрового предсказания, каждый поднабор идентифицирует одно или несколько возможных преобразований, где по меньшей мере один поднабор преобразований идентифицирует множество возможных преобразований, где средство для определения множества поднаборов преобразований содержит средство для определения

либо первого одного поднабора преобразований из множества поднаборов преобразований, причем этот первый поднабор преобразований является таким же для каждого из режимов внутреннего предсказания;

либо второго одного поднабора преобразований из множества поднаборов преобразований, причем этот второй поднабор преобразований является таким же для каждого из режимов межкадрового предсказания;

средство для выбора первого поднабора преобразований из определенного множества поднаборов преобразований для левого преобразования для текущего блока преобразований в видеоблоке видеоданных;

средство для выбора второго поднабора преобразований из определенного множества поднаборов преобразований для правого преобразования для текущего блока преобразований в видеоблоке видеоданных;

средство для определения левого преобразования из выбранного первого поднабора преобразований;

средство для определения правого преобразования из выбранного второго поднабора преобразований;

средство для определения текущего блока коэффициентов на основе левого преобразования, правого преобразования и текущего блока преобразований; и

средство для формирования потока двоичных сигналов видео, который включает в себя информацию, указывающую коэффициенты текущего блока коэффициентов, используемые для восстановления видеоблока.

15. Машиночитаемый носитель информации, содержащий команды, под управлением которых видеокодер/декодер осуществляет способ по любому из пп.1-12.

4x4 DST-VII:

{29, 55, 74, 84}
{74, 74, 0,-74}
{84,-29,-74, 55}
{55,-84, 74,-29}

Фиг. 1A

4-ТОЧЕЧНОЕ DCT-II:

{64, 64, 64, 64}
{83, 36,-36,-83}
{64,-64,-64, 64}
{36,-83, 83,-36}

Фиг. 1B

8-ТОЧЕЧНОЕ DCT-II:

{64, 64, 64, 64, 64, 64, 64, 64}
{89, 75, 50, 18,-18,-50,-75,-89}
{83, 36,-36,-83,-83,-36, 36, 83}
{75,-18,-89,-50, 50, 89, 18,-75}
{64,-64,-64, 64, 64,-64,-64, 64}
{50,-89, 18, 75,-75,-18, 89,-50}
{36,-83, 83,-36,-36, 83,-83, 36}
{18,-50, 75,-89, 89,-75, 50,-18}

Фиг. 1C

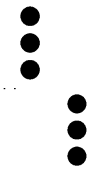
16-ТОЧЕЧНОЕ DCT-II:

{64, 64, 64, 64, 64, 64, 64, 64, 64, 64, 64, 64, 64, 64, 64, 64}
{90, 87, 80, 70, 57, 43, 25, 9, -9,-25,-43,-57,-70,-80,-87,-90}
{89, 75, 50, 18,-18,-50,-75,-89,-89,-75,-50,-18, 18, 50, 75, 89}
{87, 57, 9,-43,-80,-90,-70,-25, 25, 70, 90, 80, 43, -9,-57,-87}
{83, 36,-36,-83,-83,-36, 36, 83, 83, 36,-36,-83,-83,-36, 36, 83}
{80, 9,-70,-87,-25, 57, 90, 43,-43,-90,-57, 25, 87, 70, -9,-80}
{75,-18,-89,-50, 50, 89, 18,-75,-75, 18, 89, 50,-50,-89,-18, 75}
{70,-43,-87, 9, 90, 25,-80,-57, 57, 80,-25,-90, -9, 87, 43,-70}
{64,-64,-64, 64, 64,-64,-64, 64, 64,-64,-64, 64, 64,-64,-64, 64}
{57,-80,-25, 90, -9,-87, 43, 70,-70,-43, 87, 9,-90, 25, 80,-57}
{50,-89, 18, 75,-75,-18, 89,-50,-50, 89,-18,-75, 75, 18,-89, 50}
{43,-90, 57, 25,-87, 70, 9,-80, 80, -9,-70, 87,-25,-57, 90,-43}
{36,-83, 83,-36,-36, 83,-83, 36, 36,-83, 83,-36,-36, 83,-83, 36}
{25,-70, 90,-80, 43, 9,-57, 87,-87, 57, -9,-43, 80,-90, 70,-25}
{18,-50, 75,-89, 89,-75, 50,-18,-18, 50,-75, 89,-89, 75,-50, 18}
{9, -25, 43,-57, 70,-80, 87,-90, 90,-87, 80,-70, 57,-43, 25, -9}

Фиг. 1D

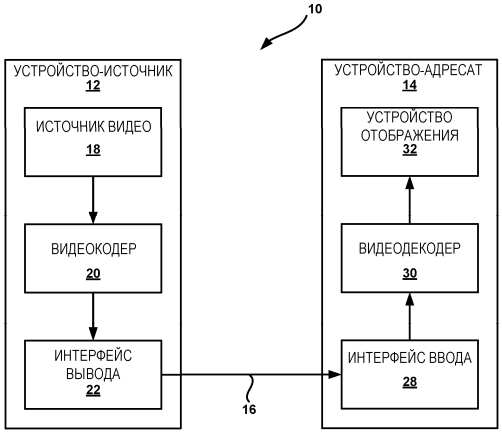
32-ТОЧЕЧНОЕ DCT-II:

{64,64,64,64,64,64,64,64,64,64,64,64,64,64,64,64,64,64,64,64,64,64,64,64,64,64,64,64,64,64,64,64}
{90,90,88,85,82,78,73,67,61,54,46,38,31,22,13,4,-4,-13,-22,-31,-38,-46,-54,-61,-67,-73,-78,-82,-85,-88,-90,-90}
{90,87,80,70,57,43,25,9,-9,-25,-43,-57,-70,-80,-87,-90,-90,-87,-80,-70,-57,-43,-25,-9,9,25,43,57,70,80,87,90}
{90,82,67,46,22,-4,-31,-54,-73,-85,-90,-88,-78,-61,-38,-13,13,38,61,78,88,90,85,73,54,31,4,-22,-46,-67,-82,-90}
{89,75,50,18,-18,-50,-75,-89,-89,-75,-50,-18,18,50,75,89,89,75,50,18,-18,-50,-75,-89,-89,-75,-50,-18,18,50,75,89}
{88,67,31,-13,-54,-82,-90,-78,-46,-4,38,73,90,85,61,22,-22,-61,-85,-90,-73,-38,4,46,78,90,82,54,13,-31,-67,-88}
{87,57,9,-43,-80,-90,-70,-25,25,70,90,80,43,-9,-57,-87,-87,-57,-9,43,80,90,70,25,-25,-70,-90,-80,-43,9,57,87}
{85,46,-13,-67,-90,-73,-22,38,82,88,54,-4,-61,-90,-78,-31,31,78,90,61,4,-54,-88,-82,-38,22,73,90,67,13,-46,-85}
{83,36,-36,-83,-83,-36,36,83,83,36,-36,-83,-83,-36,36,83,83,36,-36,-83,-83,-36,36,83,83,36,-36,-83,-83,-36,36,83}
{82,22,-54,-90,-61,13,78,85,31,-46,-90,-67,4,73,88,38,-38,-88,-73,-4,67,90,46,-31,-85,-78,-13,61,90,54,-22,-82}
{80,9,-70,-87,-25,57,90,43,-43,-90,-57,25,87,70,-9,-80,-80,-9,70,87,25,-57,-90,-43,43,90,57,-25,-87,-70,9,80}
{78,-4,-82,-73,13,85,67,-22,-88,-61,31,90,54,-38,-90,-46,46,90,38,-54,-90,-31,61,88,22,-67,-85,-13,73,82,4,-78}
{75,-18,-89,-50,50,89,18,-75,-75,18,89,50,-50,-89,-18,75,75,-18,-89,-50,50,89,18,-75,-75,18,89,50,-50,-89,-18,75}
{73,-31,-90,-22,78,67,-38,-90,-13,82,61,-46,-88,-4,85,54,-54,-85,4,88,46,-61,-82,13,90,38,-67,-78,22,90,31,-73}
{70,-43,-87,9,90,25,-80,-57,57,80,-25,-90,-9,87,43,-70,-70,43,87,-9,-90,-25,80,57,-57,-80,25,90,9,-87,-43,70}

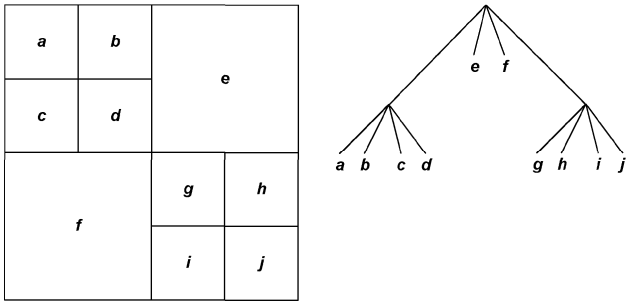


{67,-54,-78,38,85,-22,-90,4,90,13,-88,-31,82,46,-73,-61,61,73,-46,-82,31,88,-13,-90,-4,90,22,-85,-38,78,54,-67}
{64,-64,-64,64,64,-64,-64,64,64,-64,-64,64,64,-64,-64,64,64,-64,-64,64,64,-64,-64,64,64,-64,-64,64,64,-64,-64,64}
{61,-73,-46,82,31,-88,-13,90,-4,-90,22,85,-38,-78,54,67,-67,-54,78,38,-85,-22,90,4,-90,13,88,-31,-82,46,73,-61}
{57,-80,-25,90,-9,-87,43,70,-70,-43,87,9,-90,25,80,-57,-57,80,25,-90,9,87,-43,-70,70,43,-87,-9,90,-25,-80,57}
{54,-85,-4,88,-46,-61,82,13,-90,38,67,-78,-22,90,-31,-73,73,31,-90,22,78,-67,-38,90,-13,-82,61,46,-88,4,85,-54}
{50,-89,18,75,-75,-18,89,-50,-50,89,-18,-75,75,18,-89,50,50,-89,18,75,-75,-18,89,-50,-50,89,-18,-75,75,18,-89,50}
{46,-90,38,54,-90,31,61,-88,22,67,-85,13,73,-82,4,78,-78,-4,82,-73,-13,85,-67,-22,88,-61,-31,90,-54,-38,90,-46}
{43,-90,57,25,-87,70,9,-80,80,-9,-70,87,-25,-57,90,-43,-43,90,-57,-25,87,-70,-9,80,-80,9,70,-87,25,57,-90,43}
{38,-88,73,-4,-67,90,-46,-31,85,-78,13,61,-90,54,22,-82,82,-22,-54,90,-61,-13,78,-85,31,46,-90,67,4,-73,88,-38}
{36,-83,83,-36,-36,83,-83,36,36,-83,83,-36,-36,83,-83,36,36,-83,83,-36,-36,83,-83,36,36,-83,83,-36,-36,83,-83,36}
{31,-78,90,-61,4,54,-88,82,-38,-22,73,-90,67,-13,-46,85,-85,46,13,-67,90,-73,22,38,-82,88,-54,-4,61,-90,78,-31}
{25,-70,90,-80,43,9,-57,87,-87,57,-9,-43,80,-90,70,-25,-25,70,-90,80,-43,-9,57,-87,87,-57,9,43,-80,90,-70,25}
{22,-61,85,-90,73,-38,-4,46,-78,90,-82,54,-13,-31,67,-88,88,-67,31,13,-54,82,-90,78,-46,4,38,-73,90,-85,61,-22}
{18,-50,75,-89,89,-75,50,-18,-18,50,-75,89,-89,75,-50,18,18,-50,75,-89,89,-75,50,-18,-18,50,-75,89,-89,75,-50,18}
{13,-38,61,-78,88,-90,85,-73,54,-31,4,22,-46,67,-82,90,-90,82,-67,46,-22,-4,31,-54,73,-85,90,-88,78,-61,38,-13}
{9,-25,43,-57,70,-80,87,-90,90,-87,80,-70,57,-43,25,-9,-9,25,-43,57,-70,80,-87,90,-90,87,-80,70,-57,43,-25,9}
{4,-13,22,-31,38,-46,54,-61,67,-73,78,-82,85,-88,90,-90,90,-90,88,-85,82,-78,73,-67,61,-54,46,-38,31,-22,13,-4}

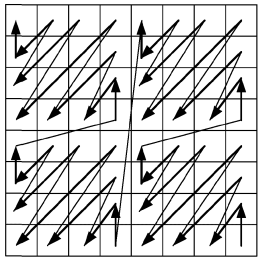
Фиг. 1F



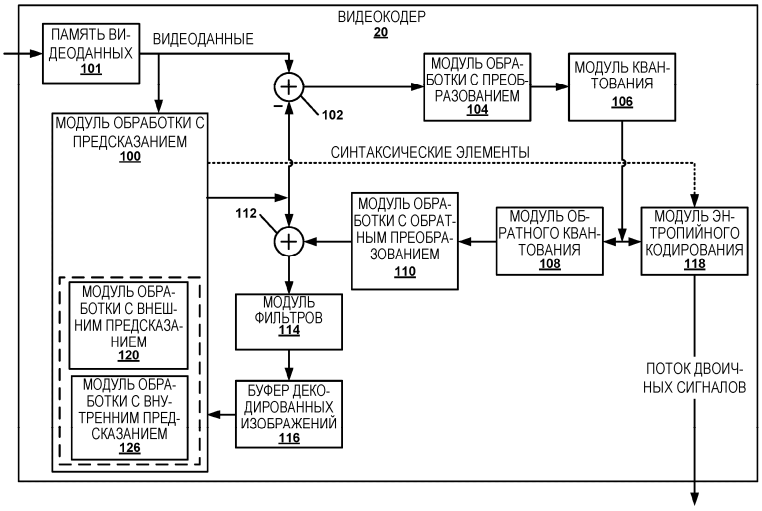
Фиг. 2



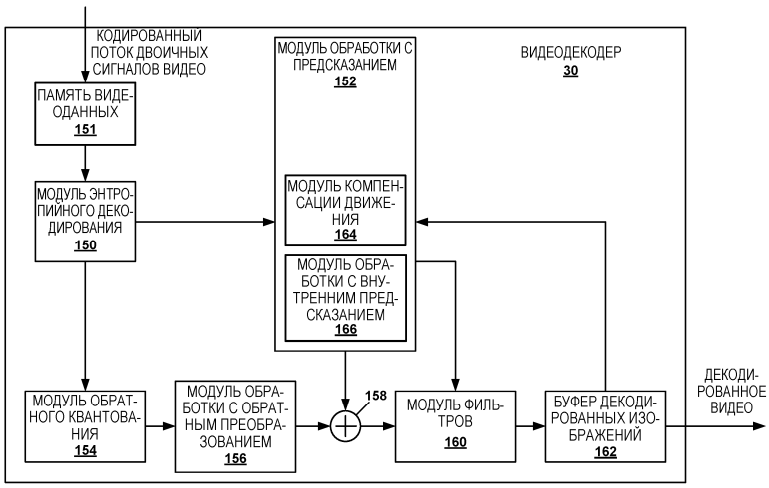
Фиг. 3



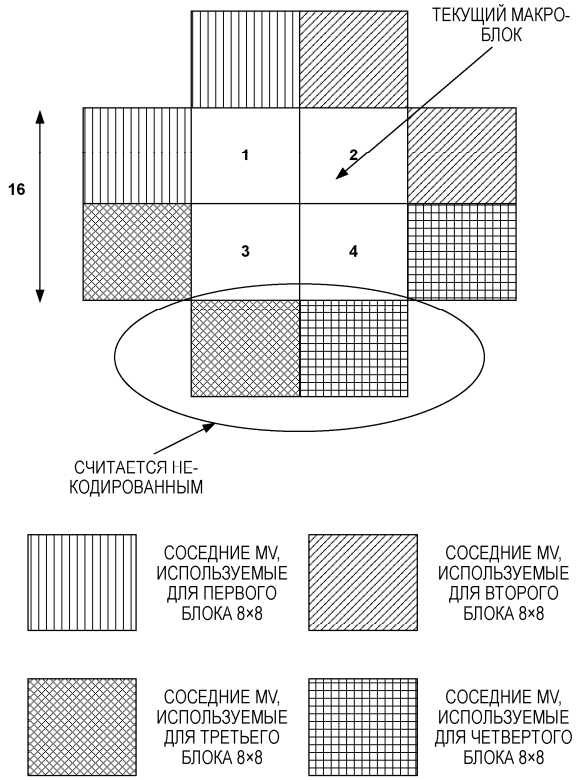
Фиг. 4



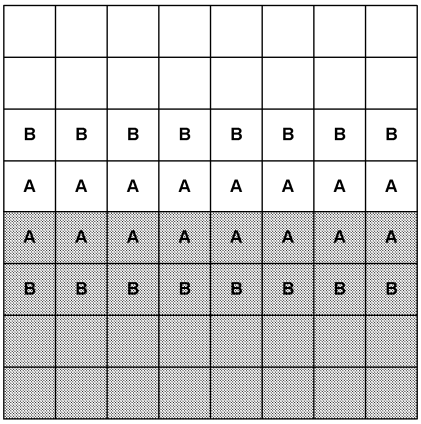
Фиг. 5



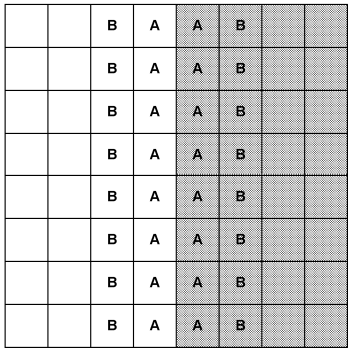
Фиг. 6



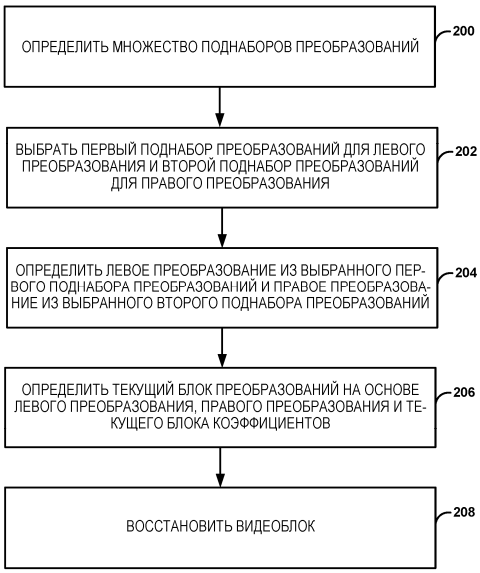
Фиг. 7



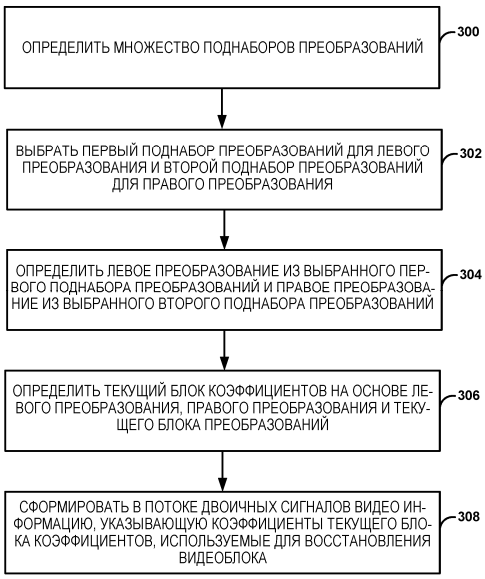
Фиг. 8A



Фиг. 8B



Фиг. 9



Фиг. 10