(19) Евразийское патентное ведомство

(11) 037461 (13) B1

(12) ОПИСАНИЕ ИЗОБРЕТЕНИЯ К ЕВРАЗИЙСКОМУ ПАТЕНТУ

(45) Дата публикации и выдачи патента
2021.03.30

(21) Номер заявки
201791564

(22) Дата подачи заявки
2016.02.10

(51) Int. Cl. ***H04N 19/176*** (2014.01)
H04N 19/119 (2014.01)
H04N 19/11 (2014.01)
H04N 19/93 (2014.01)
H04N 19/436 (2014.01)

(54) ДЕАКТИВАЦИЯ КОДИРОВАНИЯ В ПАЛИТРОВОМ РЕЖИМЕ ДЛЯ БЛОКА ВИДЕОДАННЫХ ОПРЕДЕЛЕННОГО РАЗМЕРА

(31) **62/114,537; 15/019,086**
(32) **2015.02.10; 2016.02.09**
(33) **US**
(43) **2017.12.29**
(86) **PCT/US2016/017258**
(87) **WO 2016/130622 2016.08.18**

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

(72) Изобретатель:
Серегин Вадим, Джоши Раджан Лаксман, Пу Вэй, Карчевич Марта (US)

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

(56) WANG W. ET AL.: "Non-CE5: CU dependent color palette maximum size", 19. JCT-VC MEETING; 17-10-2014 - 24-10-2014; STRASBOURG (JOINT COLLABORATIVE TEAM ON VIDEO CODING OF ISO/IEC JTC1/SC29/WG11 AND ITU-T SG.16); URL: HTTP://WFTP3.ITU.INT/AV-ARCH/JCTVC-SITE/, no. JCTVC-S0201, 9 October 2014 (2014-10-09), XP030116986, Section 1

T.-D. CHUANG ET AL.: "CE1-related: Index map scan for 64×64 palette coding block", 20. JCT-VC MEETING; 10-2-2105 - 18-2-2015; GENEVA (JOINT COLLABORATIVE TEAM ON VIDEO CODING OF ISO/IEC JTC1/SC29/WG11 AND ITU-T SG.16); URL: HTTP://WFTP3.ITU.INT/AV-ARCH/JCTVC-SITE/, no. JCTVC-T0058 - Version 2, 31 January 2015 (2015-01-31), XP030117178, Abstract, Section 1

GENHUA JIN ET AL.: "A Parallel and Pipelined Execution of H.264/AVC Intra Prediction", COMPUTER AND INFORMATION TECHNOLOGY, 2006. CIT '06. THE SIXTH IEE E INTERNATIONAL CONFERENCE ON, IEEE, PI, 1 September 2006 (2006-09-01), pages 246-246, XP031021802, ISBN: 978-0-7695-2687-4, Section 3.2 and figure 3

T.-D. CHUANG ET AL.: "CE1-related: Index map scan for 64×64 palette coding block", 20. JCT-VC MEETING; 10-2-2105 - 18-2-2015; GENEVA (JOINT COLLABORATIVE TEAM ON VIDEO CODING OF ISO/IEC JTC1/SC29/WG11 AND ITU-T SG.16); URL: HTTP://WFTP3.ITU.INT/AV-ARCH/JCTVC-SITE/, no. JCTVC-T0058 - Version 4, 12 February 2015 (2015-02-12), XP030117179, Abstract, Sections 1 and 2.2

CHI MAI LUONG: "Introduction to computer vision and image processing", INTERNET CITATION, 2008, pages 1-179, XP008118836, Retrieved from the Internet: URL: http://www.web.archive.org/web/*/http: //www.netnam.vn/unescocourse//computervision/computer.htm [retrieved on 2010-04-08], Section 10.2, pages 141-143, Section 10.8, especially page 168 first two paragraphs

"Run-length Colour Encoding; T.45 (02/00)", ITU-T STANDARD, INTERNATIONAL TELECOMMUNICATION UNION, GENEVA; CH, no. T.45 (02/00), 10 February 2000 (2000-02-10), pages 1-11, XP017462323 [retrieved on 2001-08-15], Section 2.1.2

HAN W.J.: "CE1: Grouped & run-length VLC of respred & cbp", 19. JVT MEETING; 31-03-2006 - 07-04-2006; GENEVA, CH (JOINT VIDEOTEAM OF ISO/IEC JTC1/SC29/WG11 AND ITU-T SG.16), no. JVT-S029, 28 March 2006 (2006-03-28), XP030006408, ISSN: 0000-0409, Section 3.1 with figures 3 and 4

LAROCHE G. ET AL.: "Non-RCE4: Run coding for Palette mode", 16. JCT-VC MEETING; 9-1-2014 - 17-1-2014; SAN JOSE (JOINT COLLABORATIVE TEAM ON VIDEO CODING OF ISO/IEC JTC1/SC29/WG11 AND ITU-T SG.16); URL: HTTP://WFTP3.ITU.INT/AV-ARCH/JCTVC-SITE/, no. JCTVC-P0113, 3 January 2014 (2014-01-03), XP030115608, Sections 1 and 2.2

(57) В изобретении способ кодирования видеоданных может включать в себя прием блока видеоданных, имеющего размер. Способ может включать в себя определение размера блока видеоданных. Способ может включать в себя деактивацию кодирования в палитровом режиме для блока видеоданных на основе определенного размера блока видеоданных.

037461 B1

037461 B1

Данная заявка притязает на приоритет предварительной заявки на патент (США) № 62/114537, поданной 10 февраля 2015 года, которая настоящим полностью содержится в данном документе по ссылке.

Область техники, к которой относится изобретение

Данное раскрытие сущности относится к кодированию и декодированию контента, а более конкретно, к кодированию и декодированию контента согласно режиму кодирования на основе палитр.

Уровень техники

Поддержка цифрового видео может быть включена в широкий диапазон устройств, включающих в себя цифровые телевизионные приемники, системы цифровой прямой широковещательной передачи, беспроводные широковещательные системы, персональные цифровые устройства (PDA), переносные или настольные компьютеры, планшетные компьютеры, устройства для чтения электронных книг, цифровые камеры, цифровые записывающие устройства, цифровые мультимедийные проигрыватели, устройства видеоигр, консоли для видеоигр, сотовые или спутниковые радиотелефоны, так называемые "смартфоны", устройства видеоконференц-связи, устройства потоковой передачи видео и т.п. Цифровые видеоустройства реализуют такие технологии сжатия видео, как технологии сжатия видео, описанные в стандартах, заданных посредством разрабатываемых в настоящее время стандартов MPEG-2, MPEG-4, ITU-T H.263, ITU-T H.264/MPEG-4, часть 10, усовершенствованное кодирование видео (AVC), стандарта высокоэффективного кодирования видео (HEVC), и расширений таких стандартов. Видеоустройства могут передавать, принимать, кодировать, декодировать и/или сохранять цифровую видеоинформацию более эффективно посредством реализации таких технологий сжатия видео.

Технологии сжатия видео выполняют пространственное (внутрикадровое) прогнозирование и/или временное (межкадровое) прогнозирование для того, чтобы уменьшать или удалять избыточность, внутренне присущую в видеопоследовательностях. Для кодирования видео на основе блоков серия последовательных видеомакроблоков (т.е. видеокадр или часть видеокадра) может быть сегментирована на видеоблоки. Видеоблоки в серии внутренне кодированных последовательных (I-) макроблоков изображения кодируются с использованием пространственного прогнозирования относительно опорных выборок в соседних блоках в идентичном изображении. Видеоблоки в серии взаимно кодированных последовательных (P- или B-) макроблоков изображения могут использовать пространственное прогнозирование относительно опорных выборок в соседних блоках в идентичном изображении или временное прогнозирование относительно опорных выборок в других опорных изображениях. Изображения могут упоминаться как кадры, и опорные изображения могут упоминаться как опорные кадры.

Пространственное или временное прогнозирование приводит в результате к прогнозирующему блоку для блока, который должен быть кодирован. Остаточные данные представляют пиксельные разности между исходным блоком, который должен быть кодирован, и прогнозирующим блоком. Взаимно кодированный блок кодируется согласно вектору движения, который указывает на блок опорных выборок, формирующих прогнозный блок, и остаточные данные указывают разность между кодированным блоком и прогнозным блоком. Внутренне кодированный блок кодируется согласно режиму внутреннего кодирования и остаточным данным. Для дополнительного сжатия остаточные данные могут быть преобразованы из пиксельной области в область преобразования, приводя к остаточным коэффициентам, которые затем могут быть квантованы. Квантованные коэффициенты, первоначально размещаемые в двумерном массиве, могут сканироваться для того, чтобы формировать одномерный вектор коэффициентов, и может применяться энтропийное кодирование с тем, чтобы достигать еще большего сжатия.

Контент, такой как изображение, может кодироваться и декодироваться с использованием палитрового режима. Обычно палитровый режим представляет собой технологию, предусматривающую использование палитры значений цвета для того, чтобы представлять контент. Контент может кодироваться таким образом, что контент представляется посредством карты индексов, которая включает в себя значения, соответствующие значениям цвета в палитре. Карта индексов может декодироваться, чтобы получать значения цвета, чтобы восстанавливать контент.

Сущность изобретения

Технологии этого раскрытия сущности относятся к кодированию контента на основе палитр. Например, при кодировании контента на основе палитр кодер контента (например, кодер контента, такой как видеокодер или видеодекодер) может формировать "палитру" в качестве таблицы цветов для представления видеоданных конкретной области (например, данного блока). Кодирование контента на основе палитр, например, может быть особенно полезным для кодирования областей видеоданных, имеющих относительно небольшое число цветов. Вместо кодирования фактических пиксельных значений (или их остатков), кодер контента может кодировать индексы палитр (например, значения индекса) для одного или более пикселов, которые связывают пикселы с записями в палитре, представляющей цвета пикселов. Технологии, описанные в этом раскрытии сущности, могут включать в себя технологии для различных комбинаций одного или более из передачи в служебных сигналах режимов кодирования на основе палитр, передачи палитр, извлечения палитр, извлечения значения непередаваемых синтаксических элементов, передачи карт кодирования на основе палитр и других синтаксических элементов, прогнозирования записей палитры, кодирования серий индексов палитр, энтропийного кодирования информации палитры и различные другие технологии палитрового кодирования.

В одном примере это раскрытие сущности описывает способ, содержащий прием блока видеоданных, имеющего размер; определение размера блока видеоданных; и деактивацию кодирования в палитровом режиме для блока видеоданных на основе определенного размера блока видеоданных.

В одном примере это раскрытие сущности описывает устройство, содержащее запоминающее устройство, выполненное с возможностью сохранять видеоданные; и видеокодер, поддерживающий связь с запоминающим устройством, причем видеокодер выполнен с возможностью: принимать блок видеоданных, имеющий размер, из запоминающего устройства; определять размер блока видеоданных; и деактивировать кодирование в палитровом режиме для блока видеоданных на основе определенного размера блока видеоданных.

В одном примере это раскрытие сущности описывает устройство, содержащее средство для приема блока видеоданных, имеющего размер; средство для определения размера блока видеоданных; и средство для деактивации кодирования в палитровом режиме для блока видеоданных на основе определенного размера блока видеоданных.

В одном примере это раскрытие сущности описывает долговременный считываемый компьютером носитель хранения данных, сохраняющий инструкции, которые при выполнении инструктируют одному или более процессоров: принимать блок видеоданных, имеющий размер; определять размер блока видеоданных; и деактивировать кодирование в палитровом режиме для блока видеоданных на основе определенного размера блока видеоданных.

Подробности одного или более примеров данного раскрытия сущности изложены на прилагаемых чертежах и в нижеприведенном описании. Другие признаки, цели и преимущества раскрытия сущности должны становиться очевидными из описания и чертежей, а также из формулы изобретения.

Краткое описание чертежей

Фиг. 1 является блок-схемой, иллюстрирующей примерную систему кодирования видео, которая может использовать технологии, описанные в этом раскрытии сущности;

фиг. 2 - блок-схемой, иллюстрирующей примерный видеокодер, который может выполнять технологии, описанные в этом раскрытии сущности;

фиг. 3 - блок-схемой, иллюстрирующей примерный видеодекодер, который может выполнять технологии, описанные в этом раскрытии сущности;

фиг. 4 - концептуальной схемой, иллюстрирующей пример определения палитры для кодирования видеоданных, в соответствии с технологиями этого раскрытия сущности;

фиг. 5 - концептуальной схемой, иллюстрирующей примеры определения индексов для палитры для видеоблока, в соответствии с технологиями этого раскрытия сущности;

фиг. 6 - концептуальной схемой, иллюстрирующей пример определения максимальной длины серии с копированием сверху, при условии порядка растрового сканирования, в соответствии с технологиями этого раскрытия сущности;

фиг. 7 - блок-схемой последовательности операций способа, иллюстрирующей примерный процесс для обработки видеоданных в соответствии с технологиями для кодирования видео на основе палитр этого раскрытия сущности.

Подробное описание изобретения

Аспекты этого раскрытия сущности направлены на технологии для кодирования контента (например, кодирования видео) и сжатия данных контента (например, сжатия видеоданных). В частности, это раскрытие сущности описывает технологии для кодирования на основе палитр данных контента (например, видеоданных). В различных примерах этого раскрытия сущности, технологии этого раскрытия сущности могут быть направлены на процессы прогнозирования или кодирования блока в палитровом режиме, чтобы повышать эффективность кодирования и/или уменьшать сложность кодека, как подробнее описано ниже. Например, раскрытие сущности описывает технологии, связанные с ограничением размера палитрового блока для палитрового режима.

При использовании в данном документе примеры термина "контент" могут изменяться на термин "видео", и примеры термина "видео" могут изменяться на термин "контент". Это является истинным независимо от того, используются термины "контент" или "видео" в качестве прилагательного, существительного или другой части речи. Например, ссылка на "кодер контента" также включает в себя ссылку на "видеокодер", и ссылка на "видеокодер" также включает в себя ссылку на "кодер контента". Аналогично, ссылка на "контент" также включает в себя ссылку на "видео", и ссылка на "видео" также включает в себя ссылку на "контент".

При использовании в данном документе "контент" означает любой тип контента. Например, "контент" может означать видео, экранный контент, изображение, любой графический контент, любой отображаемый контент или любые данные, соответствующие означенному (например, видеоданные, данные экранного контента, данные изображений, данные графического контента, данные отображаемого контента и т.п.).

При использовании в данном документе термин "видео" может означать экранный контент, движущийся контент, множество изображений, которые могут представляться в последовательности, или любые данные, соответствующие означенному (например, данные экранного контента, данные движущего-

ся контента, видеоданные, данные изображений и т.п.).

При использовании в данном документе, термин "изображение" может означать одно изображение, одно или более изображений, одно или более изображений из множества изображений, соответствующих видео, одно или более изображениям из множества изображений, не соответствующих видео, множество изображений, соответствующих видео (например, все изображения, соответствующие видео, или не все изображения, соответствующие видео), подчасть одного изображения, множество подчастей одного изображения, множество подчастей, соответствующих множеству изображений, один или более графических примитивов, данные изображений, графические данные и т.п.

При традиционном кодировании видео изображения предположительно имеют непрерывный тон и пространственную плавность. На основе этих допущений разработаны различные инструментальные средства, такие как преобразования на основе блоков, фильтрация и другие инструментальные средства кодирования, и такие инструментальные средства демонстрируют хорошую производительность для видео естественного контента. Тем не менее, в таких вариантах применения, как удаленный рабочий стол, совместная работа и беспроводной дисплей, машиногенерируемый экранный контент может представлять собой доминирующий контент, который должен сжиматься. Этот тип экранного контента имеет тенденцию иметь дискретный тон, резкие линии и высококонтрастные границы объектов. Допущение относительно непрерывного тона и плавности более не может применяться, и в силу этого традиционные технологии кодирования видео могут быть неэффективными при сжатии контента (например, экранного контента).

В одном примере кодирования видео на основе палитр видеокодер может кодировать блок видеоданных посредством определения палитры для блока (например, явного кодирования палитры, прогнозирования палитры или комбинации вышеозначенного), нахождения записи в палитре, чтобы представлять значение одного или более пикселов, и кодирования как палитры, так и блока со значениями индекса, которые указывают запись в палитре, используемую для того, чтобы представлять пиксельные значения блока. В некоторых примерах, видеокодер может передавать в служебных сигналах палитру и/или значения индекса в кодированном потоке битов. В свою очередь, видеодекодер может получать, из кодированного потока битов, палитру для блока, а также значения индекса для отдельных пикселов блока. Видеодекодер может связывать значения индекса пикселов с записями палитры, чтобы восстанавливать различные пиксельные значения блока.

Например, конкретная область видеоданных предположительно может иметь относительно небольшое число цветов. Видеокодер (например, видеокодер или видеодекодер) может кодировать (например, кодировать или декодировать) так называемую "палитру", чтобы представлять видеоданные конкретной области. Палитра может выражаться как таблица цветов или пиксельных значений, представляющих видеоданные конкретной области (например, данного блока). Видеокодер может кодировать индекс, который связывает одно или более пиксельных значений с надлежащим значением в палитре. Каждый пиксел может быть ассоциирован с записью в палитре, которая представляет цвет пиксела. Например, палитра может включать наиболее доминирующие пиксельные значения в данный блок. В некоторых случаях наиболее доминирующие пиксельные значения могут включать в себя одно или более пиксельных значений, которые возникают наиболее часто в блоке. Дополнительно, в некоторых случаях видеокодер может применять пороговое значение, чтобы определять то, должно или нет пиксельное значение быть включено в качестве одного из наиболее доминирующих пиксельных значений в блок. Согласно различным аспектам кодирования на основе палитр, видеокодер может кодировать значения индекса, указывающие одно или более пиксельных значений текущего блока, вместо кодирования фактических пиксельных значений или их остатков для текущего блока видеоданных. В контексте кодирования на основе палитр значения индекса указывают соответствующие записи в палитре, которые используются для того, чтобы представлять отдельные пиксельные значения текущего блока. Вышеприведенное описание имеет намерение предоставлять общее описание кодирования видео на основе палитр.

Кодирование на основе палитр, которое может быть, в частности, подходящим для кодирования экраногенерируемого контента или другого контента, при котором одно или более традиционных инструментальных средств кодирования являются неэффективными. Технологии для кодирования на основе палитр видеоданных могут использоваться с одной или более других технологий кодирования, таких как технологии для взаимного или внутреннего прогнозирующего кодирования. Например, как подробнее описано ниже, кодер или декодер, или комбинированный кодер-декодер (кодек) может быть выполнен с возможностью осуществлять взаимное и внутреннее прогнозирующее кодирование, а также кодирование на основе палитр.

В некоторых примерах технологии кодирования на основе палитр могут быть выполнены с возможностью использования с одним или более стандартов кодирования видео. Например, стандарт высокоэффективного кодирования видео (HEVC) представляет собой новый стандарт кодирования видео, разрабатываемый посредством Объединенной группы для совместной работы над видеостандартами (JCT-VC) Экспертной группы в области кодирования видео (VCEG) ITU-T и Экспертной группы по киноизображению (MPEG) ISO/IEC. Окончательный документ по HEVC-стандарту опубликован как "ITU-T H.265, SERIES H: AUDIOVISUAL AND MULTIMEDIA SYSTEMS Infrastructure of audiovisual services -

Coding of moving video - High efficiency video coding", Telecommunication Standardization Sector of International Telecommunication Union (ITU), апрель 2013 г.

Чтобы предоставлять более эффективное кодирование экраногенерируемого контента, JCT-VC разрабатывает расширение для HEVC-стандарта, называемое "стандартом HEVC-кодирования экранного контента (SCC)". Недавний рабочий проект HEVC SCC-стандарта, называемый "Проектом 2 SCC HEVC" или "WD2", описывается в документе JCTVC-S1005, R. Joshi и J. Xu, "HEVC screen content coding draft text 2", Joint Collaborative Team on Video Coding (JCT-VC) ITU-T SG 16 WP 3 и ISO/IEC JTC 1/SC 29/WG 11, 19th Meeting: Страсбург, FR, 17-24 октября 2014 года.

Относительно инфраструктуры HEVC в качестве примера технологии кодирования на основе палитр могут быть выполнены с возможностью использоваться в качестве режима на основе единиц кодирования (CU). В других примерах, технологии кодирования на основе палитр могут быть выполнены с возможностью использоваться в качестве режима на основе единиц прогнозирования (PU) в инфраструктуре HEVC. Соответственно, все следующие раскрытые процессы, описанные в контексте CU-режима, дополнительно или альтернативно, могут применяться к PU. Тем не менее, эти примеры на основе HEVC не должны считаться ограничением или недочетом технологий кодирования на основе палитр, описанных в данном документе, поскольку технологии в данном документе могут применяться, чтобы работать независимо либо в качестве части других существующих или будущих разрабатываемых систем/стандартов. В этих случаях единица для палитрового кодирования может представлять собой квадратные блоки, прямоугольные блоки или даже области непрямоугольной формы.

В некоторых примерах палитра может извлекаться для одной или более CU, PU или любой области данных (например, любого блока данных). Например, палитра может содержать (и может состоять из) наиболее доминирующие пиксельные значения в текущей CU, при этом CU представляет собой область данных для этого конкретного примера. Размер и элементы палитры сначала передаются из видеокодера в видеодекодер. Размер и/или элементы палитры для кодируемой текущей CU могут непосредственно кодироваться или прогнозирующе кодироваться с использованием размера и/или элементов палитры в CU, которые граничат с текущей CU, т.е. в соседних CU (например, при этом соседняя CU может включать в себя CU выше текущей CU и/или слева от текущей CU). После этого пиксельные значения в CU кодируются на основе палитры согласно определенному порядку сканирования. Для каждого пиксельного местоположения в CU флаг, например palette_flag, сначала передается, чтобы указывать то, включено или нет пиксельное значение в палитру. Для тех пиксельных значений, которые увязываются с записью в палитре, индекс палитры, ассоциированный с этой записью, передается в служебных сигналах для данного пиксельного местоположения в CU. Для тех пиксельных значений, которые не существуют в палитре, специальный индекс может назначаться пикселу, и фактическое пиксельное значение передается для данного пиксельного местоположения в CU. Эти пикселы упоминаются как "пикселы с управляющим кодом". Пиксел с управляющим кодом может кодироваться с использованием любого существующего способа энтропийного кодирования, такого как кодирование фиксированной длины, унарное кодирование и т.д.

Выборки в блоке видеоданных могут обрабатываться (например, сканироваться) с использованием порядка горизонтального растрового сканирования или другого порядка сканирования. Например, видеокодер может преобразовывать двумерный блок индексов палитр в одномерный массив посредством сканирования индексов палитр с использованием порядка горизонтального растрового сканирования. Аналогично, видеодекодер может восстанавливать блок индексов палитр с использованием порядка горизонтального растрового сканирования. Соответственно, это раскрытие сущности может именовать предыдущую выборку в качестве выборки, которая предшествует выборке, в данный момент кодируемой в блоке в порядке сканирования. Следует принимать во внимание, что также может быть применимым сканирование, отличное от горизонтального растрового сканирования, к примеру, порядок вертикального растрового сканирования. Вышеприведенный пример, а также другие примеры, изложенные в этом раскрытии сущности, имеют намерение предоставлять общее описание кодирования видео на основе палитр.

Фиг. 1 является блок-схемой, иллюстрирующей примерную систему 10 кодирования видео, которая может использовать технологии этого раскрытия сущности. При использовании в данном документе термин "видеокодер" относится обобщенно к видеокодерам и видеодекодерам. В этом раскрытии сущности, термины "кодирование видео" или "кодирование" могут относиться обобщенно к кодированию видео или декодированию видео. Видеокодер 20 и видеодекодер 30 системы 10 кодирования видео представляют примеры устройств, которые могут быть выполнены с возможностью осуществлять технологии для кодирования видео на основе палитр в соответствии с различными примерами, описанными в этом раскрытии сущности. Например, видеокодер 20 и видеодекодер 30 могут быть выполнены с возможностью избирательно кодировать различные блоки видеоданных, к примеру CU или PU при HEVC-кодировании, с использованием кодирования на основе палитр либо кодирования не на основе палитр. Режимы кодирования не на основе палитр могут означать различные режимы взаимного прогнозирующего временного кодирования или режимы внутреннего прогнозирующего пространственного кодирования, к примеру различные режимы кодирования, указываемые посредством HEVC-стандарта.

Как показано на фиг. 1, система 10 кодирования видео включает в себя исходное устройство 12 и целевое устройство 14. Исходное устройство 12 формирует кодированные видеоданные. Соответственно, исходное устройство 12 может упоминаться в качестве устройства кодирования видео или устройства кодирования видео. Целевое устройство 14 может декодировать кодированные видеоданные, сформированные посредством исходного устройства 12. Соответственно, целевое устройство 14 может упоминаться в качестве устройства декодирования видео или устройства декодирования видео. Исходное устройство 12 и целевое устройство 14 могут быть примерами устройств кодирования видео или устройств кодирования видео.

Исходное устройство 12 и целевое устройство 14 могут содержать широкий диапазон устройств, включающих в себя настольные компьютеры, мобильные вычислительные устройства, ноутбуки (например, переносные компьютеры), планшетные компьютеры, абонентские приставки, телефонные трубки, к примеру, так называемые смартфоны, телевизионные приемники, камеры, устройства отображения, цифровые мультимедийные проигрыватели, консоли для видеоигр, встроенные в автомобиль компьютеры и т.п.

Целевое устройство 14 может принимать кодированные видеоданные из исходного устройства 12 через канал 16. Канал 16 может содержать одну или более сред или устройств, допускающих перемещение кодированных видеоданных из исходного устройства 12 в целевое устройство 14. В одном примере канал 16 может содержать одну или более сред связи, которые предоставляют возможность исходному устройству 12 передавать кодированные видеоданные непосредственно в целевое устройство 14 в реальном времени. В этом примере, исходное устройство 12 может модулировать кодированные видеоданные согласно стандарту связи, такому как протокол беспроводной связи, и может передавать модулированные видеоданные в целевое устройство 14. Одна или более сред связи могут включать в себя беспроводные среды связи и/или проводные среды связи, к примеру, радиочастотный (RF) спектр или одну или более физических линий передачи. Одна или более сред связи могут составлять часть сети с коммутацией пакетов, такой как локальная вычислительная сеть, глобальная вычислительная сеть или глобальная сеть (например, Интернет). Одна или более сред связи могут включать в себя маршрутизаторы, коммутаторы, базовые станции или другое оборудование, которое упрощает передачу из исходного устройства 12 в целевое устройство 14.

В другом примере, канал 16 может включать в себя носитель хранения данных, который сохраняет кодированные видеоданные, сформированные посредством исходного устройства 12. В этом примере, целевое устройство 14 может осуществлять доступ к носителю хранения данных, например, через доступ к диску или доступ по карте. Носитель хранения данных может включать в себя множество локально доступных носителей хранения данных, таких как Blu-Ray-диски, DVD, CD-ROM, флэш-память или другие подходящие цифровые носители хранения данных для сохранения кодированных видеоданных.

В дополнительном примере, канал 16 может включать в себя файловый сервер или другое промежуточное устройство хранения данных, которое сохраняет кодированные видеоданные, сформированные посредством исходного устройства 12. В этом примере, целевое устройство 14 может осуществлять доступ к кодированным видеоданным, сохраненным на файловом сервере или другом промежуточном устройстве хранения данных через потоковую передачу или загрузку. Файловый сервер может представлять собой тип сервера, допускающего сохранение кодированных видеоданных и передачу кодированных видеоданных в целевое устройство 14. Примерные файловые серверы включают в себя веб-серверы (например, для веб-узла), серверы по протоколу передачи файлов (FTP), устройства по протоколу системы хранения данных с подключением по сети (NAS) и локальные накопители на дисках.

Целевое устройство 14 может осуществлять доступ к кодированным видеоданным через стандартное соединение для передачи данных, к примеру, Интернет-соединение. Примерные типы соединений для передачи данных могут включать в себя беспроводные каналы (например, Wi-Fi-соединения), проводные соединения (например, DSL, кабельный модем и т.д.) или комбинации вышеозначенного, которые являются подходящими для осуществления доступа к кодированным видеоданным, сохраненным на файловом сервере. Передача кодированных видеоданных из файлового сервера может представлять собой потоковую передачу, передачу на основе загрузки или комбинацию вышеозначенного.

Исходное устройство 12 и целевое устройство 14 могут быть выполнены с возможностью осуществлять кодирование на основе палитр в соответствии с этим раскрытием сущности. Тем не менее, технологии этого раскрытия сущности для кодирования на основе палитр не ограничены приложениями или настройками беспроводной связи. Технологии могут применяться к кодированию видео в поддержку множества мультимедийных приложений, таких как телевизионные широковещательные передачи по радиоинтерфейсу, кабельные телевизионные передачи, спутниковые телевизионные передачи, потоковые передачи видео, например через Интернет, кодирование видеоданных для хранения на носителе хранения данных, декодирование видеоданных, сохраненных на носителе хранения данных, или другие приложения. В некоторых примерах система 10 кодирования видео может быть выполнена с возможностью поддерживать одностороннюю или двустороннюю передачу видео, чтобы поддерживать такие приложения, как потоковая передача видео, воспроизведение видео, широковещательная передача видео и/или видеотелефония.

Система 10 кодирования видео, проиллюстрированная на фиг. 1, является просто примером, и тех-

нологии этого раскрытия сущности могут применяться к настройкам кодирования видео (например, кодирования видео или декодирования видео), которые необязательно включают в себя передачу данных между устройствами кодирования и декодирования. В других примерах данные извлекаются из локального запоминающего устройства, передаются в потоковом режиме по сети и 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 посредством устройств, которые не обмениваются данными друг с другом, а просто кодируют данные в запоминающее устройство и/или извлекают и декодируют данные из запоминающего устройства.

В примере по фиг. 1 исходное устройство 12 включает в себя видеоисточник 18, видеокодер 20 и интерфейс 22 вывода. В некоторых примерах интерфейс 22 вывода может включать в себя модулятор/демодулятор (модем) и/или передающее устройство. Видеоисточник 18 может включать в себя устройство видеозахвата, например видеокамеру, видеоархив, содержащий ранее захваченные видеоданные, интерфейс прямых видеотрансляций, чтобы принимать видеоданные от поставщика видеоконтента, и/или компьютерную графическую систему для формирования видеоданных либо комбинацию таких источников видеоданных.

Видеокодер 20 может кодировать видеоданные из видеоисточника 18. В некоторых примерах исходное устройство 12 непосредственно передает кодированные видеоданные в целевое устройство 14 через интерфейс 22 вывода. В других примерах кодированные видеоданные также могут быть сохраняться на носителе хранения данных или файловом сервере для последующего доступа посредством целевого устройства 14 для декодирования и/или воспроизведения.

В примере по фиг. 1, целевое устройство 14 включает в себя интерфейс 28 ввода, видеодекодер 30 и устройство 32 отображения. В некоторых примерах интерфейс 28 ввода включает в себя приемное устройство и/или модем. Интерфейс 28 ввода может принимать кодированные видеоданные по каналу 16. Устройство 32 отображения может быть интегрировано или может быть внешним для целевого устройства 14. В общем, устройство 32 отображения отображает декодированные видеоданные. Устройство 32 отображения может содержать множество устройств отображения, таких как жидкокристаллический дисплей (LCD), плазменный дисплей, дисплей на органических светодиодах (OLED) или другой тип устройства отображения.

Это раскрытие сущности, в общем, может означать "передачу в служебных сигналах" или "передачу" определенной информации посредством видеокодера 20 в другое устройство, такое как видеодекодер 30. Термин "передача служебных сигналов" или "передача", в общем, может означать передачу синтаксических элементов и/или других данных, используемых для того, чтобы декодировать сжатые видеоданные. Эта связь может осуществляться в реальном или практически в реальном времени. Альтернативно, эта связь может осуществляться в промежутке времени, к примеру, может осуществляться при сохранении синтаксических элементов на считываемом компьютером носителе хранения данных в кодированном потоке битов во время кодирования, которые затем могут извлекаться посредством устройства декодирования видео в любое время после сохранения на этом носителе. Таким образом, тогда как видеодекодер 30 может упоминаться как "принимающий" определенную информацию, прием информации не обязательно осуществляется в реальном или практически в реальном времени и может извлекаться из носителя в определенное время после хранения.

Видеокодер 20 и видеодекодер 30 могут быть реализованы как любая из множества надлежащих схем, к примеру, как один или более микропроцессоров, процессоров цифровых сигналов (DSP), специализированных интегральных схем (ASIC), программируемых пользователем вентильных матриц (FPGA), дискретная логика, аппаратные средства либо любые комбинации вышеозначенного. Если технологии реализуются частично в программном обеспечении, устройство может сохранять инструкции для программного обеспечения на подходящем долговременном считываемом компьютером носителе хранения данных и может выполнять инструкции в аппаратных средствах с использованием одного или более процессоров, чтобы осуществлять технологии этого раскрытия сущности. Любое из вышеозначенного (включающее в себя аппаратные средства, программное обеспечение, комбинацию аппаратных средств и программного обеспечения и т.д.) может рассматриваться в качестве одного или более процессоров. Каждый из видеокодера 20 и видеодекодера 30 может быть включен в один или более кодеров или декодеров, любой из которых может быть интегрирован как часть комбинированного кодера/декодера (кодека) в соответствующем устройстве.

В некоторых примерах видеокодер 20 и видеодекодер 30 работают согласно стандарту сжатия видео, такому как HEVC-стандарт, упомянутый выше, и описанному в HEVC-стандарте. В дополнение к базовому HEVC-стандарту, прилагаются постоянные усилия для того, чтобы формировать расширения для масштабируемого кодирования видео, кодирования многовидового видео и трехмерного кодирования для HEVC. Помимо этого, режимы кодирования на основе палитр, например, как описано в этом раскрытии сущности, могут предоставляться для расширения HEVC-стандарта. В некоторых примерах технологии, описанные в этом раскрытии сущности для кодирования на основе палитр, могут применяться к кодерам и декодерам, выполненным с возможностью работы согласно другим стандартам коди-

рования видео. Соответственно, применение режима кодирования на основе палитр для кодирования единиц кодирования (CU) или единиц прогнозирования (PU) в HEVC-кодеке описывается для целей примера.

В HEVC и других стандартах кодирования видео видеопоследовательность типично включает в себя последовательность изображений. Изображения также могут упоминаться в качестве "кадров". Изображение может включать в себя три массива выборок, обозначаемых как S_L, S_{Cb} и S_{Cr} S_L представляет собой двумерный массив (т.е. блок) выборок сигнала яркости. S_{Cb} представляет собой двумерный массив выборок Cb-цветности. S_{Cr} представляет собой двумерный массив выборок Cr-цветности. Выборки цветности также могут упоминаться в документе как выборки "сигнала цветности". В других случаях изображение может быть монохромным и может включать в себя только массив выборок сигнала яркости.

Для того чтобы формировать кодированное представление изображения, видеокодер 20 может формировать набор единиц дерева кодирования (CTU). Каждая из CTU может представлять собой блок дерева кодирования выборок сигнала яркости, два соответствующих блока дерева кодирования выборок сигнала цветности и синтаксические структуры, используемые для того, чтобы кодировать выборки блоков дерева кодирования. Блок дерева кодирования может представлять собой блок N×N выборок. CTU также может упоминаться в качестве "древовидного блока" или наибольшей единицы кодирования (LCU). CTU HEVC в широком смысле могут быть аналогичными макроблокам других стандартов, таких как H.264/AVC. Тем не менее, CTU не обязательно ограничивается конкретным размером и может включать в себя одну или более единиц кодирования (CU). Серия последовательных макроблоков (слайс) может включать в себя целое число CTU, упорядоченных последовательно в растровом сканировании. Кодированная серия последовательных макроблоков может содержать заголовок серии последовательных макроблоков и данные серии последовательных макроблоков. Заголовок серии последовательных макроблоков серии последовательных макроблоков может представлять собой синтаксическую структуру, которая включает в себя синтаксические элементы, которые предоставляют информацию относительно серии последовательных макроблоков. Данные серии последовательных макроблоков могут включать в себя кодированные CTU серии последовательных макроблоков.

Это раскрытие сущности может использовать термин "видеоединица" или "видеоблок", или "блок" для того, чтобы означать один или более блоков выборок и синтаксических структур, используемых для того, чтобы кодировать выборки одного или более блоков выборок. Примерные типы видеоединиц или блоков могут включать в себя CTU, CU, PU, единицы преобразования (TU), макроблоки, сегменты макроблока и т.д. В примере HEVC, для того чтобы формировать кодированную CTU, видеокодер 20 может рекурсивно выполнять сегментацию на дерево квадрантов для блоков дерева кодирования CTU для того, чтобы разделять блоки дерева кодирования на блоки кодирования, отсюда имя "единицы дерева кодирования". Блок кодирования представляет собой блок N×N выборок. CU может представлять собой блок кодирования выборок сигнала яркости и два соответствующих блока кодирования выборок сигнала цветности изображения, которое имеет массив выборок сигнала яркости, массив Cb-выборок и массив Cr-выборок, и синтаксические структуры, используемые для того, чтобы кодировать выборки блоков кодирования. Видеокодер 20 может сегментировать блок кодирования CU на один или более прогнозных блоков. Прогнозный блок может представлять собой прямоугольный (т.е. квадратный или неквадратный) блок выборок, к которым применяется идентичное прогнозирование. Единица прогнозирования (PU) CU может представлять собой прогнозный блок выборок сигнала яркости, два соответствующих прогнозных блока выборок сигнала цветности изображения и синтаксические структуры, используемые для того, чтобы прогнозировать выборки прогнозных блоков. Видеокодер 20 может формировать прогнозирующие блоки сигналов яркости, прогнозирующие Cb-блоки и прогнозирующие Cr-блоки для прогнозных блоков сигналов яркости, прогнозных Cb-блоков и прогнозных Cr-блоков каждой PU CU.

Видеокодер 20 может использовать внутреннее прогнозирование или взаимное прогнозирование для того, чтобы формировать прогнозирующие блоки для PU. Если видеокодер 20 использует внутреннее прогнозирование для того, чтобы формировать прогнозирующие блоки PU, видеокодер 20 может формировать прогнозирующие блоки PU на основе декодированных выборок изображения, ассоциированного с PU.

Если видеокодер 20 использует взаимное прогнозирование для того, чтобы формировать прогнозирующие блоки PU, видеокодер 20 может формировать прогнозирующие блоки PU на основе декодированных выборок одного или более изображений, за исключением изображения, ассоциированного с PU. Видеокодер 20 может использовать унипрогнозирование или бипрогнозирование для того, чтобы формировать прогнозирующие блоки PU. Когда видеокодер 20 использует унипрогнозирование для того, чтобы формировать прогнозирующие блоки для PU, PU может иметь один вектор движения (MV). Когда видеокодер 20 использует бипрогнозирование для того, чтобы формировать прогнозирующие блоки для PU, PU может иметь два MV.

После того как видеокодер 20 формирует прогнозирующие блоки (например, прогнозирующие блоки сигналов яркости, прогнозирующие Cb-блоки и прогнозирующие Cr-блоки) для одной или более PU CU, видеокодер 20 может формировать остаточные блоки для CU. Каждая выборка в остаточном блоке CU может указывать разность между выборкой в прогнозирующем блоке PU CU и соответствующей вы-

боркой в блоке кодирования CU. Например, видеокодер 20 может формировать остаточный блок сигналов яркости для CU. Каждая выборка в остаточном блоке сигналов яркости CU указывает разность между выборкой сигнала яркости в одном из прогнозирующих блоков сигналов яркости CU и соответствующей выборкой в исходном блоке кодирования сигналов яркости CU. Помимо этого, видеокодер 20 может формировать остаточный Cb-блок для CU. Каждая выборка в остаточном Cb-блоке CU может указывать разность между Cb-выборкой в одном из прогнозирующих Cb-блоков CU и соответствующей выборкой в исходном Cb-блоке кодирования CU. Видеокодер 20 также может формировать остаточный Cr-блок для CU. Каждая выборка в остаточном Cr-блоке CU может указывать разность между Cr-выборкой в одном из прогнозирующих Cr-блоков CU и соответствующей выборкой в исходном Cr-блоке кодирования CU.

Кроме того, видеокодер 20 может использовать сегментацию на дерево квадрантов для того, чтобы разлагать остаточные блоки (например, остаточные блоки сигналов яркости, остаточные Cb-блоки и остаточные Cr-блоки) CU на один или более блоков преобразования (например, на блоки преобразования сигналов яркости, Cb-блоки преобразования и Cr-блоки преобразования). Блок преобразования может представлять собой прямоугольный блок выборок, к которым применяется идентичное преобразование. Единица преобразования (TU) CU может представлять собой блок преобразования выборок сигнала яркости, два соответствующих блока преобразования выборок сигнала цветности и синтаксические структуры, используемые для того, чтобы преобразовывать выборки блоков преобразования. Таким образом, каждая TU CU может быть ассоциирована с блоком преобразования сигналов яркости, Cb-блоком преобразования и Cr-блоком преобразования. Блок преобразования сигналов яркости, ассоциированный с TU, может представлять собой субблок остаточного блока сигналов яркости CU. Cb-блок преобразования может представлять собой субблок остаточного Cb-блока CU. Cr-блок преобразования может представлять собой субблок остаточного Cr-блока CU.

Видеокодер 20 может применять одно или более преобразований к блоку преобразования для того, чтобы формировать блок коэффициентов для TU. Блок коэффициентов может представлять собой двумерный массив коэффициентов преобразования. Коэффициент преобразования может быть скалярной величиной. Например, видеокодер 20 может применять одно или более преобразований к блоку преобразования сигналов яркости TU для того, чтобы формировать блок коэффициентов сигнала яркости для TU. Видеокодер 20 может применять одно или более преобразований к Cb-блоку преобразования TU для того, чтобы формировать Cb-блок коэффициентов для TU. Видеокодер 20 может применять одно или более преобразований к Cr-блоку преобразования TU для того, чтобы формировать Cr-блок коэффициентов для TU.

После формирования блока коэффициентов (например, блока коэффициентов сигнала яркости, Cb-блока коэффициентов или Cr-блока коэффициентов), видеокодер 20 может квантовать блок коэффициентов. Квантование, в общем, означает процесс, в котором коэффициенты преобразования квантуются, чтобы, возможно, уменьшать объем данных, используемых для того, чтобы представлять коэффициенты 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 выполнять контекстно-адаптивное двоичное арифметическое кодирование (CABAC) для синтаксических элементов, указывающих квантованные коэффициенты преобразования.

Относительно CABAC в качестве примера, видеокодер 20 и видеодекодер 30 могут выбирать вероятностную модель (также называемую "контекстной моделью") для того, чтобы кодировать символы, ассоциированные с блоком видеоданных, на основе контекста. Например, контекстная модель (Ctx) может представлять собой индекс или смещение, которое применяется для того, чтобы выбирать один из множества различных контекстов, каждый из которых может соответствовать конкретной вероятностной модели. Соответственно, различная вероятностная модель типично задается для каждого контекста. После кодирования или декодирования элемента выборки вероятностная модель дополнительно обновляется на основе значения элемента выборки, чтобы отражать самые актуальные оценки вероятности для элемента выборки. Например, вероятностная модель может поддерживаться в качестве состояния в конечном автомате. Каждое конкретное состояние может соответствовать конкретному значению вероятности. Следующее состояние, которое соответствует обновлению вероятностной модели, может зависеть от значения текущего элемента выборки (например, элемента выборки, в данный момент кодируемого). Соответственно, на выбор вероятностной модели могут оказывать влияние значения ранее кодированных элементов выборки, поскольку значения указывают, по меньшей мере, частично, вероятность элемента выборки, имеющего данное значение. Процесс контекстного кодирования, описанный выше, в общем, может упоминаться как режим контекстно-адаптивного декодирования.

Следовательно, видеокодер 20 может кодировать целевой символ с использованием вероятностной модели. Аналогично, видеодекодер 30 может синтаксически анализировать целевой символ с использованием вероятностной модели. В некоторых случаях видеокодер 20 может кодировать синтаксические элементы с использованием комбинации контекстно-адаптивного и неконтекстно-адаптивного кодирования. Например, видеокодер 20 может кодировать элементы выборки контекста посредством выбора вероятностной модели или "контекстной модели", которая работает в контексте для того, чтобы кодиро-

вать элементы выборки. Напротив, видеокодер 20 может выполнять обходное кодирование элементов выборки посредством обхода или опускания процесса регулярного арифметического кодирования при кодировании элементов выборки. В таких примерах, видеокодер 20 может использовать фиксированную вероятностную модель для того, чтобы выполнять обходное кодирование элементов выборки. Иными словами, элементы выборки после обходного кодирования не включают в себя обновления контекста или вероятностей.

Видеокодер 20 может выводить поток битов, который включает в себя энтропийно кодированные синтаксические элементы. Поток битов также может включать в себя синтаксические элементы, которые не кодируются энтропийно. Поток битов может включать в себя последовательность битов, которая формирует представление кодированных изображений и ассоциированных данных. Поток битов может содержать последовательность единиц уровня абстрагирования от сети (NAL). Каждая из NAL-единиц включает в себя заголовок NAL-единицы и инкапсулирует первичную байтовую последовательность данных (RBSP). Заголовок NAL-единицы может включать в себя синтаксический элемент, который указывает код типа NAL-единицы. Код типа NAL-единицы, указываемый посредством заголовка NAL-единицы для NAL-единицы, указывает тип NAL-единицы. RBSP может представлять собой синтаксическую структуру, содержащую целое число байтов, которое инкапсулируется в NAL-единице. В некоторых случаях, RBSP включает в себя нулевые биты.

Различные типы NAL-единиц могут инкапсулировать различные типы RBSP. Например, первый тип NAL-единицы может инкапсулировать RBSP для набора параметров изображения (PPS), второй тип NAL-единицы может инкапсулировать RBSP для кодированной серии последовательных макроблоков, третий тип NAL-единицы может инкапсулировать RBSP для дополнительной улучшающей информации (SEI) и т.д. NAL-единицы, которые инкапсулируют RBSP для данных кодирования видео (в противоположность RBSP для наборов параметров и SEI-сообщений), могут упоминаться в качестве NAL-единиц слоя кодирования видео (VCL).

Видеодекодер 30 может принимать поток битов, сформированный посредством видеокодера 20. Помимо этого, видеодекодер 30 может синтаксически анализировать поток битов для того, чтобы декодировать синтаксические элементы из потока битов. Видеодекодер 30 может восстанавливать изображения видеоданных, по меньшей мере, частично на основе синтаксических элементов, декодированных из потока битов. Процесс для того, чтобы восстанавливать видеоданные, в общем, может быть обратным по отношению к процессу, выполняемому посредством видеокодера 20. Например, видеодекодер 30 может использовать MV PU для того, чтобы определять прогнозирующие блоки для PU текущей CU. Помимо этого, видеодекодер 30 может обратно квантовать блоки коэффициентов преобразования, ассоциированные с TU текущей CU. Видеодекодер 30 может выполнять обратные преобразования для блоков коэффициентов преобразования для того, чтобы восстанавливать блоки преобразования, ассоциированные с TU текущей CU. Видеодекодер 30 может восстанавливать блоки кодирования текущей CU посредством суммирования выборок прогнозирующих блоков для PU текущей CU с соответствующими выборками блоков преобразования TU текущей CU. Посредством восстановления блоков кодирования для каждой CU изображения видеодекодер 30 может восстанавливать изображение.

В некоторых примерах видеокодер 20 и видеодекодер 30 могут быть выполнены с возможностью осуществлять кодирование на основе палитр. Например, при кодировании на основе палитр, вместо выполнения технологий внутреннего прогнозирующего или взаимного прогнозирующего кодирования, описанных выше, видеокодер 20 и видеодекодер 30 могут кодировать так называемую палитру в качестве таблицы цветов или пиксельных значений, представляющих видеоданные конкретной области (например, данного блока). Таким образом, вместо кодирования фактических пиксельных значений или их остатков для текущего блока видеоданных, видеокодер может кодировать значения индекса для одного или более пиксельных значений текущего блока, причем значения индекса указывают записи в палитре, которые используются для того, чтобы представлять пиксельные значения текущего блока.

Например, видеокодер 20 может кодировать блок видеоданных посредством определения палитры для блока, нахождения записи в палитре, чтобы представлять значение каждого пиксела, и кодирования палитры и значений индекса для пикселов, связывающих пиксельное значение с палитрой. Видеодекодер 30 может получать, из кодированного потока битов, палитру для блока, а также значения индекса для пикселов блока. Видеодекодер 30 может увязывать значения индекса отдельных пикселов с записями палитры, чтобы восстанавливать пиксельные значения блока. В случаях, когда значение индекса, ассоциированное с отдельным пикселом, не увязывается ни с одним значением индекса соответствующей палитры для блока, видеодекодер 30 может идентифицировать такой пиксел в качестве пиксела с управляющим кодом, в целях кодирования на основе палитр.

В другом примере, видеокодер 20 может кодировать блок видеоданных согласно следующим операциям. Видеокодер 20 может определять значения остатка прогнозирования для отдельных пикселов блока, определять палитру для блока и находить запись (например, значение индекса) в палитре, имеющую значение, представляющее значение одного или более значений остатка прогнозирования отдельных пикселов. Дополнительно, видеокодер 20 может кодировать блок со значениями индекса, которые указывают запись в палитре, используемую для того, чтобы представлять соответствующее значение

остатка прогнозирования для каждого отдельного пиксела блока. Видеодекодер 30 может получать из кодированного потока битов, передаваемого в служебных сигналах посредством исходного устройства 12, палитру для блока, а также значения индекса для значений остатка прогнозирования, соответствующих отдельным пикселам блока. Как описано выше, значения индекса могут соответствовать записям в палитре, ассоциированной с текущим блоком. В свою очередь, видеодекодер 30 может связывать значения индекса остаточных прогнозных значений с записями палитры, чтобы восстанавливать остаточные прогнозные значения блока. Остаточные прогнозные значения могут суммироваться с прогнозными значениями (например, полученными с использованием внутреннего или взаимного прогнозирования), чтобы восстанавливать пиксельные значения блока. Как подробнее описано ниже, базовая идея касательно кодирования на основе палитр состоит в том, что для данного блока видеоданных, которые должны кодироваться, видеокодер 20 может извлекать палитру, которая включает в себя наиболее доминирующие пиксельные значения в текущем блоке. Например, палитра может означать число пиксельных значений, которые определяются или предполагаются как доминирующие и/или характерные для текущей CU. Видеокодер 20 может сначала передавать размер и элементы палитры в видеодекодер 30. Дополнительно, видеокодер 20 может кодировать пиксельные значения в данном блоке согласно определенному порядку сканирования. Для каждого пиксела, включенного в данный блок, видеокодер 20 может передавать в служебных сигналах значение индекса, которое увязывает пиксельное значение с соответствующей записью в палитре. Если пиксельное значение не включено в палитру (т.е. не существуют записи палитры, которые указывают конкретное пиксельное значение кодированного в палитровом режиме блока), то такой пиксел задается как "пиксел с управляющим кодом". В соответствии с кодированием на основе палитр видеокодер 20 может кодировать и передавать в служебных сигналах значение индекса, которое зарезервировано для пиксела с управляющим кодом. В некоторых примерах видеокодер 20 также может кодировать и передавать в служебных сигналах пиксельное значение или остаточное значение (либо их квантованные версии) для пиксела с управляющим кодом, включенного в данный блок.

После приема кодированного потока видеобитов, передаваемого в служебных сигналах посредством видеокодера 20, видеодекодер 30 может сначала определять палитру на основе информации, принимаемой из видеокодера 20. Видеодекодер 30 затем может увязывать принятые значения индекса, ассоциированные с пиксельными местоположениями в данном блоке, с записями палитры, чтобы восстанавливать пиксельные значения данного блока. В некоторых случаях видеодекодер 30 может определять то, что пиксел кодированного в палитровом режиме блока представляет собой пиксел с управляющим кодом, к примеру, посредством определения того, что пиксел кодируется в палитровом режиме со значением индекса, зарезервированным для пикселов с управляющим кодом. В случаях, когда видеодекодер 30 идентифицирует пиксел с управляющим кодом в кодированном в палитровом режиме блоке, видеодекодер 30 может принимать пиксельное значение или остаточное значение (либо их квантованные версии) для пиксела с управляющим кодом, включенного в данный блок. Видеодекодер 30 может восстанавливать кодированный в палитровом режиме блок посредством увязки отдельных пиксельных значений с соответствующими записями палитры и посредством использования пиксельного значения или остаточного значения (либо их квантованных версий), чтобы восстанавливать все пикселы с управляющим кодом, включенные в кодированный в палитровом режиме блок.

Как изложено выше, в примерном режиме палитрового кодирования палитра может включать в себя записи, пронумерованные посредством индекса. Каждая запись может представлять значения или интенсивности цветовых компонентов (например, в таких цветовых пространствах, как YCbCr, RGB, YUV, CMYK или другие форматы), которые могут использоваться в качестве предиктора для блока или в качестве конечных восстановленных выборок блоков. Как описано в заявочном документе по стандарту JCTVC-Q0094 (Wei Pu и др., "AHG10: Suggested Software for Palette Coding based on REext6.0", JCTVC-Q0094, Valencia, ES, 27 марта - 4 апреля 2014 года), палитра может включать в себя записи, которые копируются из палитры предикторов. Палитра предикторов может включать в себя записи палитры из блоков, ранее кодированных с использованием палитрового режима, или других восстановленных выборок. Для каждой записи в палитре предиктора двоичный флаг отправляется, чтобы указывать то, копируется или нет эта запись в текущую палитру (указываемую посредством флага=1). Это упоминается как двоичный вектор палитрового прогнозирования. Дополнительно, текущая палитра может содержать (например, состоять из) новые записи, явно передаваемые в служебных сигналах. Число новых записей также может передаваться в служебных сигналах.

В качестве другого примера, в палитровом режиме палитра может включать в себя записи, пронумерованные посредством индекса, представляющего значения цветовых компонентов, которые могут использоваться в качестве предикторов для выборок блоков или в качестве конечных восстановленных выборок блоков. Каждая запись в палитре может содержать, например, один компонент сигнала яркости (например, значение сигнала яркости), два компонента сигнала цветности (например, два значения сигнала цветности) или три цветовых компонента (например, RGB, YUV и т.д.). Ранее декодированные записи палитры могут сохраняться в списке. Этот список может использоваться, например, для того, чтобы прогнозировать записи палитры в CU текущего палитрового режима. Двоичный вектор прогнозирования может передаваться в служебных сигналах в потоке битов, чтобы указывать то, какие записи в списке

многократно используются в текущей палитре. В некоторых примерах кодирование по длинам серий может использоваться для того, чтобы сжимать двоичный предиктор палитры. Например, значение длины серии может кодироваться с использованием экспоненциального кода Голомба нулевого порядка.

В этом раскрытии сущности, предполагается, что каждая запись палитры указывает значения для всех цветовых компонентов выборки. Тем не менее, принципы этого раскрытия сущности являются применимыми к использованию отдельной палитры и/или отдельной записи палитры для каждого цветового компонента. Кроме того, предполагается, что выборки в блоке обрабатываются с использованием порядка горизонтального растрового сканирования. Тем не менее, также являются применимыми другие сканирования, к примеру, порядок вертикального растрового сканирования. Как упомянуто выше, палитра может содержать прогнозированные записи палитры, например прогнозированные из палитры, используемой для того, чтобы кодировать предыдущий блок(и), и новые записи, которые могут быть конкретными для текущего блока и явно передаются в служебных сигналах. Кодер и декодер могут знать число прогнозных и новых записей палитры, и их сумма может указывать полный размер палитры в блоке.

Как предложено в примере JCTVC-Q0094, процитированного выше, каждая выборка в блоке, кодированном с палитрой, может принадлежать одному из трех режимов, как указано ниже.

Режим кодирования управляющим кодом. В этом режиме выборочное значение не включено в палитру в качестве записи палитры, и значение квантованной выборки передается в служебных сигналах явно для всех цветовых компонентов. Это является аналогичным передаче служебных сигналов новых записей палитры, хотя для новых записей палитры, значения цветовых компонентов не квантуются.

Режим с копированием сверху (также называемый "режимом с копированием сверху" или "режимом копирования"). В этом режиме индекс записи палитры для текущей выборки копируется из выборки, расположенной непосредственно выше текущей выборки в блоке выборок. В других примерах для режима с копированием сверху, блок видеоданных может транспонироваться таким образом, что выборка выше блока фактически представляет собой выборку слева от блока.

Режим значений (также называемый "индексным режимом" или "режимом серий"). В этом режиме значение индекса записи палитры явно передается в служебных сигналах.

Как описано в данном документе, индекс записи палитры может упоминаться в качестве "индекса палитры" или просто "индекса". Эти термины могут использоваться взаимозаменяемо, чтобы описывать технологии этого раскрытия сущности. Помимо этого, как подробнее описано ниже, индекс палитры может иметь одно или более ассоциированных значений цвета или интенсивности. Например, индекс палитры может иметь одно ассоциированное значение цвета или интенсивности, ассоциированное с одним цветовым компонентом или компонентом интенсивности пиксела (например, компонентом красного цвета RGB-данных, Y-компонентом YUV-данных и т.п.). В другом примере, индекс палитры может иметь несколько ассоциированных значений цвета или интенсивности. В некоторых случаях, кодирование видео на основе палитр может применяться для того, чтобы кодировать монохромное видео. Соответственно, "значение цвета", в общем, может означать любой цветовой или нецветовой компонент, используемый для того, чтобы формировать пиксельное значение.

Значение серии может указывать серию значений индекса палитры, которые кодируются с использованием идентичного режима палитрового кодирования. Например, относительно режима значений, видеокодер (например, видеокодер 20 или видеодекодер 30) может кодировать значение индекса и значение серии, которое указывает число последовательных выборок в порядке сканирования, которые имеют идентичное значение индекса и которые кодируются с индексом палитры. Относительно режима с копированием сверху, видеокодер может кодировать индикатор того, что значение индекса для текущего выборочного значения является идентичным значению индекса соседней сверху выборки (например, выборки, которая позиционируется выше выборки, в данный момент кодируемой в блоке), и значение серии, которое указывает число последовательных выборок в порядке сканирования, которые также копируют значение индекса из соседней сверху выборки и которые кодируются с индексом палитры. Соответственно, в вышеприведенных примерах, серия значений индекса палитры означает серию значений палитры, имеющих идентичное значение, или серию значений индекса, которые копируются из соседних сверху выборок.

Следовательно, серия может указывать, для данного режима, число последующих выборок, которые принадлежат идентичному режиму. В некоторых случаях, передача в служебных сигналах значения индекса и значения серии может быть аналогичной кодированию по длинам серий. В примере для целей иллюстрации, строка последовательных значений индекса палитры индексного блока, соответствующего блоку видеоданных, может быть равной 0, 2, 2, 2, 2, 5. В некоторых примерах индексный блок может включать в себя одно или более пиксельных значений с управляющим кодом. Каждое значение индекса в индексном блоке может соответствовать выборке в блоке видеоданных. В этом примере, видеокодер может кодировать вторую выборку (например, первое значение индекса палитры "2") с использованием режима значений. После кодирования значения индекса в 2, видеокодер может кодировать серию в 3, которая указывает то, что три последующих выборки также имеют идентичное значение индекса палитры в 2. Аналогичным образом, кодирование серии четырех индексов палитр после кодирования индекса с использованием режима с копированием сверху может указывать то, что всего пять индексов палитр

копируются из соответствующих значений индекса палитры в строке выше позиции выборки, в данный момент кодируемой.

С использованием палитры видеокодер 20 и/или видеодекодер 30 могут быть выполнены с возможностью кодировать блок выборок (например, блок видеоданных) в индексный блок, причем индексный блок представляет собой блок, включающий в себя значения индекса, например, для каждой выборки, которые преобразуют выборки в одну или более записей палитры, и в некоторых примерах, включающий в себя одно или более пиксельных значений с управляющим кодом. Каждый пиксел блока видеоданных может кодироваться с использованием режима серий, режима копирования или режима кодирования управляющим кодом. В некоторых примерах пикселы в первой строке блока видеоданных могут кодироваться только с использованием режима серий или режима кодирования управляющим кодом.

Синтаксический элемент palette_run_type_flag указывает то, используется ли режим серий или режим копирования. Например, видеокодер 20 может быть выполнен с возможностью передавать в служебных сигналах синтаксический элемент palette run type flag посредством кодирования значения, соответствующего синтаксическому элементу palette_run_type_flag, в кодированный поток битов для выборки блока видеоданных. Видеодекодер 20 может быть выполнен с возможностью принимать кодированный поток битов, содержащий кодированное значение, соответствующее синтаксическому элементу palette_run_type_flag. Видеодекодер 20 может быть выполнен с возможностью декодировать кодированное значение, чтобы определять значение, соответствующее синтаксическому элементу palette_run_type_flag, и в силу этого определять то, режим серий или режим копирования используется для выборки блока видеоданных. Например, когда значение palette_run_type_flag является первым значением, то режим серий может использоваться для выборки блока видеоданных. В качестве другого примера, когда значение palette_run_type_flag является вторым значением, то режим копирования может использоваться для выборки блока видеоданных.

В некоторых примерах, когда используется режим серий или режим копирования, синтаксический элемент palette_index может передаваться в служебных сигналах вместе с синтаксическим элементом palette_run. Например, видеокодер 20 может быть выполнен с возможностью передавать в служебных сигналах синтаксические элементы palette_index и palette_run посредством кодирования значения (например, значения индекса), соответствующего palette_index, и значения (например, значения серии), соответствующего palette_run, в кодированный поток битов. Видеодекодер 30 может быть выполнен с возможностью принимать кодированный поток битов, содержащий кодированное значение, соответствующее синтаксическому элементу palette_index, и кодированное значение, соответствующее синтаксическому элементу palette_run. Видеодекодер 20 может быть выполнен с возможностью декодировать кодированное значение, соответствующее palette_index, и кодированное значение, соответствующее palette_run, чтобы, соответственно, определять значение (например, значение индекса), соответствующее palette_index, и значение (например, значение серии), соответствующее palette_run.

Когда используется режим серий, значение серии указывает число пикселов, которые должны иметь идентичный индекс палитры. Тем не менее, когда используется режим копирования, значение серии указывает число пикселов, для которых индекс палитры (например, значение индекса) копируется из другого пиксела, соответствующего каждому пикселу (например, непосредственно выше каждого соответствующего пиксела).

В некоторых примерах режим кодирования управляющим кодом кодируется в режиме серий, причем конкретный индекс палитры может использоваться для того, чтобы указывать этот режим. Индекс палитры, используемый для того, чтобы указывать режим кодирования управляющим кодом, равен размеру палитры текущего блока согласно некоторым примерам. В режиме кодирования управляющим кодом, значение серии может не кодироваться, поскольку режим кодирования управляющим кодом применяется к одному пикселу (например, пиксельному триплету (Y, U и V)), при этом значение(я) цветового компонента(ов) для одного пиксела явно передается в служебных сигналах в качестве palette_escape_val. В некоторых примерах режим копирования не может активироваться для первой строки в блоке, поскольку отсутствуют пикселы выше первой строки, принадлежащие идентичному блоку.

Флаг palette_escape_val_present_flag может передаваться в служебных сигналах в расчете на блок, чтобы указывать использование пикселов с управляющим кодом. Этот флаг равен 1, что указывает то, что имеется, по меньшей мере, один пиксел с управляющим кодом в кодированном в палитровом режиме блоке, и флаг равен 0 в противном случае. Например, видеокодер 20 может быть выполнен с возможностью передавать в служебных сигналах синтаксический элемент palette_escape_val_present_flag посредством кодирования значения, соответствующего синтаксическому элементу palette_escape_ val_present_flag, в кодированный поток битов. Видеодекодер 20 может быть выполнен с возможностью принимать кодированный поток битов, содержащий кодированное значение, соответствующее синтаксическому элементу palette_escape_val_present_flag. Видеодекодер 20 может быть выполнен с возможностью декодировать кодированное значение, чтобы определять значение, соответствующее синтаксическому элементу palette_escape_val_present_flag, и в силу этого определять то, существует или нет по меньшей мере один пиксел с управляющим кодом в кодированном в палитровом режиме блоке.

В некоторых примерах размер палитры ограничивается диапазоном от 0 до max_palette_size, при-

чем второе значение передается в служебных сигналах. Для блока, кодированного с помощью палитрового режима, палитра, в некоторых примерах, может прогнозироваться из записей палитры одного или более ранее кодированных в палитровом режиме блоков. Палитра может явно передаваться в служебных сигналах для текущего блока в качестве одной или более новых записей. В других примерах палитра ранее кодированного блока может полностью многократно использоваться (например, копироваться) для текущего блока, что называется "режимом совместного использования палитр". В некоторых примерах флаг palette_share_flag может передаваться в служебных сигналах, чтобы указывать то, что полная палитра предыдущего блока многократно используется без модификации как есть для текущего блока.

При кодировании блока видео с использованием палитрового режима, шаблон пиксельного сканирования (например, порядок сканирования) может включать в себя, например: вертикальное пересекающееся или горизонтальное пересекающееся (змеевидное) сканирование. Шаблон сканирования, используемый в блоке, может извлекаться согласно флагу palette_transpose_flag, передаваемому в служебных сигналах в расчете на единицу блока.

В ходе кодирования в палитровом режиме может применяться процесс регулирования индексов палитр. Начиная со второго пиксела в текущем блоке, палитровый режим предыдущего пиксела в порядке сканирования может проверяться (например, определяться). В некоторых примерах максимальный размер индекса палитры может сначала уменьшаться на 1. Если палитровый режим для предыдущего пиксела в порядке сканирования равен режиму серий (т.е. если предыдущий пиксел в порядке сканирования кодирован или должен кодироваться с использованием режима серий), индекс палитры (например, значение индекса) для текущего пиксела может уменьшаться на 1, если значение индекса превышает или равно значению индекса для предыдущего пиксела в порядке сканирования. Аналогично, если палитровый режим для предыдущего пиксела в порядке сканирования равен режиму копирования (т.е. если предыдущий пиксел в порядке сканирования кодирован или должен кодироваться с использованием режима копирования), то индекс палитры (например, значение индекса) для текущего пиксела может уменьшаться на 1, если индекс превышает индекс верхней палитры.

Видеокодер 20 может быть выполнен с возможностью энтропийно кодировать индексный блок, чтобы сжимать индексный блок. Аналогично, видеодекодер 30 может быть выполнен с возможностью энтропийно декодировать кодированный индексный блок, чтобы формировать индексный блок, из которого видеодекодер 30 может формировать блок выборок (например, блок видеоданных, кодированных посредством кодера 20). Например, энтропийное кодирование на основе длин серий может использоваться для того, чтобы сжимать и распаковывать индексный блок. В некоторых примерах видеокодер 20 и видеодекодер 30 могут быть выполнены с возможностью, соответственно, энтропийно кодировать и декодировать значения индекса в индексном блоке с использованием CABAC.

Чтобы применять CABAC-кодирование к информации (например, синтаксическому элементу, индексному блоку, такому как значения индекса индексного блока, или к другой информации), видеокодер (например, видеокодер 20 и видеодекодер 30) может выполнять преобразование в двоичную форму для информации. Преобразование в двоичную форму означает процесс преобразования информации в последовательность из одного или более битов. Каждая последовательность из одного или более битов может упоминаться как "элементы выборки". Преобразование в двоичную форму представляет собой процесс без потерь и может включать в себя одну или комбинацию следующих технологий кодирования: кодирование фиксированной длины, унарное кодирование, усеченное унарное кодирование, кодирование усеченным кодом Раиса, кодирование кодом Голомба, кодирование экспоненциальным кодом Голомба, кодирование кодом Голомба-Райса, любую форму кодирования кодом Голомба, любую форму кодирования кодом Райса и любую форму энтропийного кодирования. Например, преобразование в двоичную форму может включать в себя представление целочисленного значения 5 в качестве 00000101 с использованием технологии на основе 8-битовой фиксированной длины или в качестве 11110 с использованием технологии унарного кодирования.

После преобразования в двоичную форму видеокодер может идентифицировать контекст кодирования. Контекст кодирования может идентифицировать вероятности кодирования элементов выборки, имеющих конкретные значения. Например, контекст кодирования может указывать вероятность в 0,7 кодирования элемента выборки со значением 0 и вероятность в 0,3 кодирования элемента выборки со значением 1. После идентификации контекста кодирования видеокодер может арифметически кодировать этот элемент выборки на основе контекста, что известно как кодирование в контекстном режиме. Элементы выборки, кодированные с использованием CABAC-кодирования в контекстном режиме, могут упоминаться как "контекстные элементы выборки".

Дополнительно, вместо выполнения кодирования в контекстном режиме для всех элементов выборки, видеокодер (например, видеокодер 20 и видеодекодер 30) может кодировать некоторые элементы выборки с использованием обходного CABAC-кодирования (например, кодирования в обходном режиме). Кодирование в обходном режиме означает обходной режим CABAC-кодера, при этом обходное кодирование представляет собой процесс арифметического кодирования элемента выборки без использования адаптивного контекста (например, контекста кодирования). Иными словами, механизм обходного кодирования не выбирает контексты и может допускать вероятность в 0,5 для обоих символов (0 и 1).

Хотя кодирование в обходном режиме не может быть настолько эффективным по полосе пропускания, как кодирование в контекстном режиме, может быть вычислительно менее затратным выполнять кодирование в обходном режиме для элемента выборки, а не выполнять кодирование в контекстном режиме для элемента выборки. Дополнительно, выполнение кодирования в обходном режиме может обеспечивать более высокую степень параллелизации и пропускную способность. Элементы выборки, кодированные с использованием кодирования в обходном режиме, могут упоминаться как "обходные элементы выборки".

Видеокодер 20 и видеодекодер 30 могут быть сконфигурированы с CABAC-кодером (например, CABAC-кодером и CABAC-декодером, соответственно). CABAC-кодер может включать в себя механизм кодирования в контекстном режиме, чтобы выполнять CABAC-кодирование в контекстном режиме, и механизм кодирования в обходном режиме, чтобы выполнять кодирование в обходном режиме. Если элемент выборки кодируется в контекстном режиме, механизм кодирования в контекстном режиме используется для того, чтобы кодировать этот элемент выборки. Механизму кодирования в контекстном режиме может требоваться более двух циклов обработки для того, чтобы кодировать один элемент выборки. Тем не менее, при использовании надлежащего проектного решения по конвейеру, механизму кодирования в контекстном режиме может требоваться только n+M циклов для того, чтобы кодировать n элементов выборки, где M является объемом служебной информации для того, чтобы запускать конвейер. M обычно больше 0.

В начале процесса CABAC-кодирования (т.е. каждое переключение с обходного режима на контекстный режим и наоборот), вводится объем служебной информации по конвейеру. Если элемент выборки кодируется в обходном режиме, механизм кодирования в обходном режиме используется для того, чтобы кодировать этот элемент выборки. Механизму кодирования в обходном режиме предположительно требуется только один цикл для того, чтобы кодировать n-битовую информацию, где n может превышать единицу. Таким образом, общее число циклов для того, чтобы кодировать набор обходных элементов выборки и контекстных элементов выборки, может уменьшаться, если все обходные элементы выборки в наборе кодируются совместно (например, в последовательности без перемеженных элементов выборки после контекстного кодирования), и все контекстные элементы выборки в наборе кодируются совместно (например, в последовательности без перемеженных элементов выборки после обходного кодирования). В частности, совместное кодирование обходных элементов выборки до или после перехода к кодированию в контекстном режиме позволяет сокращать объем служебной информации, требуемый для того, чтобы перезапускать механизм кодирования в контекстном режиме. Например, видеокодер 20 и видеодекодер 30 могут быть выполнены с возможностью переключаться из обходного режима на контекстный режим (или из контекстного режима на обходной режим в других примерах) один раз, за последовательность элементов выборки после обходного и контекстного кодирования, при одновременном, соответственно, кодировании или декодировании блока видеоданных с использованием палитрового режима. В другом примере, видеокодер 20 и видеодекодер 30 могут быть выполнены с возможностью уменьшать число раз, когда процесс кодирования или декодирования переключается из обходного режима на контекстный режим (и из контекстного режима на обходной режим) при кодировании или декодировании блока видеоданных с использованием палитрового режима.

Технологии, описанные в этом раскрытии сущности, могут включать в себя технологии для различных комбинаций одного или более из передачи в служебных сигналах режимов кодирования видео на основе палитр, передачи палитр, извлечения палитр, передачи в служебных сигналах порядка сканирования, извлечения порядка сканирования и передачи карт кодирования видео на основе палитр и других синтаксических элементов. Например, технологии этого раскрытия сущности могут быть направлены на энтропийное кодирование информации палитры. В некоторых примерах технологии этого раскрытия сущности, в числе прочего, могут использоваться для того, чтобы увеличивать эффективность кодирования и уменьшать неэффективность кодирования, ассоциированную с кодированием видео на основе палитр. Соответственно, как подробнее описано ниже, технологии этого раскрытия сущности, в некоторых случаях, могут повышать эффективность и повышать скорость передачи битов при кодировании видеоданных с использованием палитрового режима.

Технологии, аспекты и/или примеры, описанные в данном документе, могут быть использованы в сочетании друг с другом в любой комбинации или отдельно друг от друга. Например, видеокодер 20 и видеодекодер 30 могут быть выполнены с возможностью осуществлять любую одну либо любую подходящую комбинацию одной или более технологий, аспектов и/или примеров, описанных в документе.

Проблема с примерной системой кодирования, описанная в документе Tzu-Der Chuang и др., "CE-1 related: Index Map scan for 64×64 palette coding block", JCTVC-T0058 version 3, выгруженном в JCT-VC Document Management System 10 февраля 2015 года (далее "JCTVC-T0058"), заключается в том, что размер палитрового блока может составлять самое большое 64×64 и с шаблоном сканирования самое большое 64×64, но самый большой размер блока преобразования составляет 32×32, если, например, применяется сканирование коэффициентов. Таким образом, в этом случае, конвейер в реализации увеличивается до размеров блоков 64×64, что не требуется без палитрового режима, и в силу этого представляет част-

ный случай для палитрового режима. JCTVC-T0058 описывает кодирование блока 64×64 в палитровом режиме в качестве четырех субблоков 32×32 посредством изменения пересекающегося сканирования 64×64 на четыре пересекающихся сканирования 32×32. Тем не менее, выполнение означенного должно требовать изменения кодирования в палитровом режиме, которое является конкретным только для палитрового блока 64×64, и в силу этого вводит, например, неоднородность в кодирование в палитровом режиме.

В различных примерах этого раскрытия сущности, технологии этого раскрытия сущности могут быть направлены на процессы прогнозирования или кодирования блока в палитровом режиме, чтобы повышать эффективность кодирования и/или уменьшать сложность кодека, например, посредством разрешения того, как (если вообще) блок 64×64 должен кодироваться с использованием палитрового режима.

В некоторых примерах этого раскрытия сущности, кодирование в палитровом режиме может деактивироваться для любого палитрового блока, имеющего размер 64×64 или больше. В других примерах кодирование в палитровом режиме может ограничиваться палитровыми блоками, имеющими размер меньше 64×64, что означает то, что кодирование в палитровом режиме может активироваться или иным образом использоваться для палитровых блоков, имеющих размер меньше 64×64. В других примерах наибольший размер палитрового блока может нормативно ограничиваться на основе наибольшего размера единицы преобразования, к примеру, нормативно ограничиваться наибольшим размером единицы преобразования. Кодирование в палитровом режиме может деактивироваться для размера палитрового блока, который превышает или в других отношениях больше наибольшего размера единицы преобразования. В таких примерах, следует понимать, что наибольший размер палитрового блока может быть основан на наибольшем размере единицы преобразования в том, что наибольший размер палитрового блока нормативно ограничивается наибольшим размером единицы преобразования. Например, видеокодер 20 может быть выполнен с возможностью нормативно ограничивать наибольший размер палитрового блока, который может кодироваться с использованием палитрового режима, наибольшим размером единицы преобразования. В этом примере, видеокодер 20 может быть выполнен с возможностью деактивировать палитровый режим или иным образом не использовать палитровый режим для любого палитрового блока, имеющего размер, больший наибольшего размера единицы преобразования, который видеокодер 20 выполнен с возможностью кодировать.

Например, если наибольший размер единицы преобразования, который видеокодер 20 выполнен с возможностью кодировать, составляет 32×32, то видеокодер 20 может быть выполнен с возможностью нормативно ограничивать наибольший размер палитрового блока 32×32. В таком примере, видеокодер 20 может быть выполнен с возможностью деактивировать палитровый режим или иным образом не использовать палитровый режим для любого палитрового блока, имеющего размер, больший 32×32. В таком примере также следует понимать, что видеокодер 20 может быть выполнен с возможностью активировать палитровый режим или иным образом использовать палитровый режим для любого палитрового блока, имеющего размер, меньший или равный 32×32. Примеры палитровых блоков, имеющих размер, больший 32×32, включают в себя, например, 64×64, 64×16, 16×64, 64×32 и 32×64.

В качестве другого примера, если наибольший размер единицы преобразования, который видеокодер 20 выполнен с возможностью кодировать, составляет 16×16, то видеокодер 20 может быть выполнен с возможностью нормативно ограничивать наибольший размер палитрового блока 16×16. В таком примере, видеокодер 20 может быть выполнен с возможностью деактивировать палитровый режим или иным образом не использовать палитровый режим для любого палитрового блока, имеющего размер, больший 16×16. В таком примере также следует понимать, что видеокодер 20 может быть выполнен с возможностью активировать палитровый режим или иным образом использовать палитровый режим для любого палитрового блока, имеющего размер, меньший или равный 16×16.

В других примерах наибольший размер единицы преобразования, который видеокодер 20 выполнен с возможностью кодировать, может нормативно ограничиваться размером блока M×N, где M и N являются положительными целыми числами и могут быть равны или не равны друг другу. В некоторых примерах M и/или N могут быть основаны на наибольшем размере единицы преобразования. Например, если наибольший размер единицы преобразования составляет 32×32, то M и N равны 32. Тем не менее, в примере, в котором наибольший размер единицы преобразования составляет 32×16, то M должен быть равен 32, и N должен быть равен 16. В таком примере, примеры палитровых блоков, имеющих размер, больший 32×16, включают в себя, например, 64×64, 64×16, 16×64, 64×32, 32×64, 32×32 и 16×32.

В некоторых примерах видеокодер 20 может быть выполнен с возможностью передавать в служебных сигналах наибольший размер единицы преобразования для конкретного набора данных. В таких примерах, видеокодер 20 может быть выполнен с возможностью деактивировать палитровый режим или иным образом не использовать палитровый режим для любого палитрового блока, ассоциированного с конкретным набором данных, который имеет размер блока, больший передаваемого в служебных сигналах наибольшего размера единицы преобразования. Соответственно, при использовании в данном документе, наибольшая единица преобразования может означать наибольшую единицу преобразования, ко-

торую видеокодер 20 выполнен с возможностью кодировать, или может означать передаваемую в служебных сигналах наибольшую единицу преобразования для конкретного набора данных (например, одного или более блоков видеоданных). Например, в то время как наибольший размер единицы преобразования может составлять 32×32, видеокодер 20 может передавать в служебных сигналах, для конкретного набора данных, то, что наибольший размер единицы преобразования составляет 16×16. Следовательно, для этого конкретного набора данных в этом примере, наибольший размер единицы преобразования составляет 16×16.

Соответственно, следует понимать, что видеокодер 20 может быть выполнен с возможностью динамически деактивировать палитровый режим или выполнен с возможностью иным образом не использовать палитровый режим на основе наибольшего размера единицы преобразования. Аналогично, следует понимать, что видеокодер 20 может быть выполнен с возможностью динамически деактивировать палитровый режим или выполнен с возможностью иным образом не использовать палитровый режим для любого палитрового блока, имеющего размер, больший наибольшей единицы преобразования. В силу этого также следует понимать, что видеокодер 20 может быть выполнен с возможностью динамически деактивировать палитровый режим или выполнен с возможностью иным образом не использовать палитровый режим для любого палитрового блока, имеющего размер, который не равен или меньше наибольшей единицы преобразования. В силу этого дополнительно следует понимать, что видеокодер 20 может быть выполнен с возможностью кодировать блок видеоданных с использованием палитрового режима только тогда, когда блок видеоданных имеет размер, который не превышает наибольшую единицу преобразования, которую видеокодер 20 может быть выполнен с возможностью кодировать. Аналогично, видеокодер 20 может быть выполнен с возможностью активировать кодирование в палитровом режиме для блока видеоданных только тогда, когда блок видеоданных имеет размер, который не превышает наибольшую единицу преобразования.

Аналогично, в силу этого следует понимать, что видеодекодер 30 может быть выполнен с возможностью динамически деактивировать палитровый режим или выполнен с возможностью иным образом не использовать палитровый режим для любого палитрового блока, имеющего размер, который не равен или меньше наибольшей единицы преобразования. В силу этого дополнительно следует понимать, что видеодекодер 30 может быть выполнен с возможностью декодировать блок видеоданных с использованием палитрового режима только тогда, когда блок видеоданных имеет размер, который не превышает наибольшую единицу преобразования, которую видеокодер 20 может быть выполнен с возможностью кодировать, и/или которую видеодекодер 30 может быть выполнен с возможностью декодировать. Аналогично, видеодекодер 30 может быть выполнен с возможностью активировать кодирование в палитровом режиме для блока видеоданных только тогда, когда блок видеоданных имеет размер, который не превышает наибольшую единицу преобразования. В других примерах видеодекодер 30 может быть выполнен с возможностью определять то, активируется или деактивируется палитровый режим, на основе значения, соответствующего флагу палитрового режима, такого как значение для синтаксического элемента palette_mode_flag.

В качестве другого примера видеодекодер 30 может быть выполнен с возможностью принимать блок видеоданных. Видеодекодер 30 может быть выполнен с возможностью определять размер видеоданных блока относительно наибольшего размера единицы преобразования. Видеодекодер 30 может быть выполнен с возможностью определять то, что принимаемый блок видео не кодируется в палитровом режиме, когда принимаемый блок видеоданных превышает размер наибольшего размера единицы преобразования.

Как изложено в данном документе, наибольший размер палитрового блока может нормативно ограничиваться. Например, наибольший размер палитрового блока может быть основан на наибольшем размере единицы преобразования, к примеру, нормативно ограничиваться наибольшим размером единицы преобразования. В некоторых примерах видеокодер 20 может быть сконфигурирован с ограничением по совместимым потокам битов, чтобы реализовывать любое ограничение по размеру палитрового блока, описанное в данном документе, что приводит к управлению тем, когда палитровый режим деактивируется, активируется или иным образом используется. Например, ограничение по совместимым потокам битов может заключаться в том, что совместимый поток битов не должен иметь блока, превышающего определенный размер, кодированного с помощью палитрового режима. В качестве другого примера ограничение по совместимым потокам битов может заключаться в том, что совместимый поток битов должен иметь блок, кодированный с помощью палитрового режима, только в случае, если блок равен или меньше определенного размера. В обоих примерах опорный определенный размер может составлять 32×32 или любой другой размер M×N, где M и N являются положительными целыми числами и могут быть равны или не равны друг другу. Тем не менее, в других примерах, опорный определенный размер в обоих вышеприведенных примерах может быть основан на наибольшем размере единицы преобразования. В таких примерах ограничение по совместимым потокам битов, например, может заключаться в том, что совместимый поток битов не должен иметь блока, превышающего наибольшую единицу преобразования. В качестве другого примера ограничение по совместимым потокам битов может заключаться в том, что

совместимый поток битов должен соответствовать одному или более нормативных ограничений, описанных в данном документе.

Относительно любого ограничения по совместимым потокам битов, описанного в данном документе, следует понимать, что, видеокодер 20 может быть сконфигурирован с любым таким ограничением(ями) в любой комбинации, чтобы управлять тем, когда палитровый режим деактивируется, активируется или иным образом используется для блока видеоданных.

В других примерах видеокодер 20 может быть выполнен с возможностью реализовывать любое ограничение по размеру палитрового блока, описанное в данном документе, посредством выполнения с возможностью разделять любой блок видеоданных, который должен быть кодирован в палитровом режиме, на субблоки M×N таким образом, что весь блок видеоданных представлен посредством субблоков M×N, где M и N являются положительными целыми числами и могут быть равны или не равны друг другу. Разделение всего блока видеоданных означает, что каждый пиксел (например, выборка) блока видеоданных является частью субблока M×N. Размер субблока может зависеть от одного или более критериев. Например, размер субблока M×N может зависеть от размера блока, используемого при кодировании коэффициентов преобразования (например, размера блока преобразования в TU), чтобы совмещать кодирование в палитровом режиме с кодированием коэффициентов преобразования. В таком примере, если видеокодер 20 выполнен с возможностью кодировать коэффициенты преобразования с использованием блоков размера 4×4, то видеокодер 20 может быть выполнен с возможностью разделять любой блок видеоданных, который должен быть кодирован в палитровом режиме, на субблоки 4×4, где M и N равны 4. Например, вместо кодирования блока 64×64 с использованием палитрового режима, видеокодер 20 может быть выполнен с возможностью разделять блок 64×64 на множество субблоков 4×4, что приводит к двумстам пятидесяти шести субблокам 4×4 в этом примере, при этом каждый субблок отдельно кодируется с использованием палитрового режима.

В другом примере вместо зависимости от одного или более критериев размер субблока M×N может составлять размер по умолчанию меньше 64×64. Например, размер по умолчанию субблока M×N может составлять 4×4, 8×8, 16×16, 32×32 или любой другой размер меньше 64×64. В этом примере видеокодер 20 может быть выполнен с возможностью реализовывать любое ограничение по размеру палитрового блока, описанное в данном документе, посредством выполнения с возможностью разделять любой блок видеоданных, который должен быть кодирован в палитровом режиме, на размер по умолчанию, к примеру, 4×4, 8×8, 16×16, 32×32 или любой другой размер меньше 64×64 соответственно.

В некоторых примерах субблоки M×N могут сканироваться согласно любому порядку сканирования. Например, видеокодер 20 может быть выполнен с возможностью сканировать субблоки M×N с использованием зигзагообразного порядка сканирования, горизонтального порядка сканирования, вертикального порядка сканирования, "змеевидного" порядка сканирования (т.е. пересекающегося порядка сканирования) или любого другого порядка сканирования.

В других примерах видеокодер 20 может быть выполнен с возможностью реализовывать любое ограничение по размеру палитрового блока, описанное в данном документе, посредством выполнения с возможностью передавать в служебных сигналах палитровый режим (например, посредством передачи в служебных сигналах значения для синтаксического элемента palette_mode_flag) для блоков, имеющих размер 64×64, но также и посредством выполнения с возможностью передавать в служебных сигналах другую связанную информацию палитры (например, многократно используемые записи палитры, новые записи палитры, размер таблицы палитры и т.д.) для размеров субблока M×N, меньших 64×64, где M и N являются положительными целыми числами и могут быть равны или не равны друг другу. Например, размер субблока M×N может составлять 32×32. В некоторых примерах размер субблока M×N может составлять 32×32, поскольку 32×32 соответствует размеру наибольшей единицы преобразования. В таком примере видеокодер 20 может быть выполнен с возможностью реализовывать любое ограничение по размеру палитрового блока, описанное в данном документе, посредством выполнения с возможностью передавать в служебных сигналах палитровый режим для блоков, имеющих размер 64×64, но также и посредством выполнения с возможностью передавать в служебных сигналах другую связанную информацию палитры с размерами субблоков 32×32 (или с любым другим размером субблока M×N). Он представляет собой один пример, описанный в данном документе, в котором видеокодер 20 может быть выполнен с возможностью гармонизировать размер блока палитрового режима с размером блока единиц преобразования. В некоторых примерах M и/или N могут быть основаны на наибольшем размере единицы преобразования. Например, если наибольший размер единицы преобразования составляет 32×32, то M и N равны 32. Порядок сканирования для размера палитрового блока 64×64 может быть идентичным порядку сканирования для каждого из блоков M×N.

В одном примере предусматривающем размер палитрового блока 64×64, видеокодер может быть выполнен с возможностью передавать в служебных сигналах палитровый режим для этого палитрового блока размера 64×64. Видеокодер затем может быть выполнен с возможностью передавать в служебных сигналах другую связанную информацию палитры для каждого субблока M×N. Например, видеокодер 20

может быть выполнен с возможностью передавать в служебных сигналах max_palette_size для каждого субблока M×N.

В других примерах видеокодер 20 может быть выполнен с возможностью реализовывать любое ограничение по размеру палитрового блока, описанное в данном документе, посредством выполнения с возможностью ограничивать самую большую длину серии значений индекса и/или значений с управляющим кодом таким образом, что она меньше порогового значения T. В таких примерах, вместо разбиения палитрового блока 64×64 на субблоки, видеокодер 20 может быть выполнен с возможностью ограничивать максимальную длину серии таким образом, что она меньше порогового значения T. Посредством ограничения значения максимальной длины серии, видеокодер 20 может быть выполнен с возможностью реализовывать ограничение по размеру палитрового блока без разделения палитрового блока на субблоки.

В некоторых примерах T может быть равен наибольшему размеру единицы преобразования. Например, если наибольший размер единицы преобразования составляет 32×32, то T может быть равен 32×32. Ниже описывается пример, предусматривающий горизонтальный пересекающийся порядок сканирования для палитрового блока 64×64 и значения T в 32×32 Вместо того, чтобы обрабатывать палитровый блок в квадрантах 32×32 (например, значение индекса с последующей длиной серии, меньшей T, к примеру, 32×32 минус 1) в этом примере, видеокодер 20 может обрабатывать палитровый блок 64×64 как есть, но ограничивать максимальную длину серии значением, меньшим значения T, к примеру, 32×32 минус 1.

В некоторых примерах видеокодер 20 может быть сконфигурирован с ограничением по совместимым потокам битов, чтобы реализовывать любое ограничение по размеру палитрового блока, описанное в данном документе. Например, ограничение по совместимым потокам битов может заключаться в том, что совместимый поток битов не должен включать в себя значение длины серии, равное или большее порогового значения T.

Фиг. 2 является блок-схемой, иллюстрирующей примерный видеокодер 20, который может реализовывать технологии этого раскрытия сущности. Фиг. 2 предоставляется для целей пояснения и не должен считаться ограничением технологий, как проиллюстрировано и описано в общих чертах в этом раскрытии сущности. Для целей пояснения, это раскрытие сущности описывает видеокодер 20 в контексте HEVC-кодирования. Тем не менее, технологии этого раскрытия сущности могут быть применимыми к другим стандартам или способам кодирования.

Видеокодер 20 представляет пример устройства, которое может быть выполнено с возможностью осуществлять технологии для кодирования на основе палитр в соответствии с различными примерами, описанными в этом раскрытии сущности.

В примере по фиг. 2, видеокодер 20 включает в себя модуль 100 блочного кодирования, запоминающее устройство 101 видеоданных, модуль 102 формирования остатков, модуль 104 обработки преобразования, модуль 106 квантования, модуль 108 обратного квантования, модуль 110 обработки обратного преобразования, модуль 112 восстановления, модуль 114 фильтрации, буфер 116 декодированных изображений и модуль 118 энтропийного кодирования. Модуль 100 блочного кодирования включает в себя модуль 120 обработки взаимного прогнозирования и модуль 126 обработки внутреннего прогнозирования. Модуль 120 обработки взаимного прогнозирования включает в себя модуль оценки движения и модуль компенсации движения (не показаны). Видеокодер 20 также включает в себя модуль 122 кодирования на основе палитр, выполненный с возможностью осуществлять различные аспекты технологий кодирования на основе палитр, описанных в этом раскрытии сущности. В других примерах видеокодер 20 может включать в себя большее, меньшее число или другие функциональные компоненты.

Запоминающее устройство 101 видеоданных может сохранять видеоданные, которые должны кодироваться посредством компонентов видеокодера 20. Видеоданные, сохраненные в запоминающем устройстве 101 видеоданных, могут получаться, например, из видеоисточника 18. Буфер 116 декодированных изображений может представлять собой запоминающее устройство опорных изображений, которое сохраняет опорные видеоданные для использования при кодировании видеоданных посредством видеокодера 20, например, в режимах внутреннего или взаимного кодирования. Запоминающее устройство 101 видеоданных и буфер 116 декодированных изображений могут формироваться посредством любого из множества запоминающих устройств, к примеру, как динамическое оперативное запоминающее устройство (DRAM), включающее в себя синхронное DRAM (SDRAM), магниторезистивное RAM (MRAM), резистивное RAM (RRAM) или другие типы запоминающих устройств. Запоминающее устройство 101 видеоданных и буфер 116 декодированных изображений могут предоставляться посредством идентичного запоминающего устройства или отдельных запоминающих устройств. В различных примерах, запоминающее устройство 101 видеоданных может быть внутрикристальным с другими компонентами видеокодера 20 или внекристальным относительно этих компонентов.

Видеокодер 20 может принимать видеоданные. Видеокодер 20 может кодировать каждую CTU в серии последовательных макроблоков изображения видеоданных. Каждая из CTU может быть ассоциирована с блоками дерева кодирования (CTB) сигнала яркости одинакового размера и соответствующими

CTB изображения. В качестве части кодирования CTU, модуль 100 блочного кодирования может выполнять сегментацию на дерево квадрантов для того, чтобы разделять CTB CTU на постепенно меньшие блоки. Меньший блок может представлять собой блоки кодирования CU. Например, модуль 100 блочного кодирования может сегментировать CTB, ассоциированный с CTU, на четыре субблока одинакового размера, сегментировать один или более субблоков на четыре субсубблока одинакового размера и т.д.

Видеокодер 20 может кодировать CU CTU для того, чтобы формировать кодированные представления CU (т.е. кодированные CU). В качестве части кодирования CU модуль 100 блочного кодирования может сегментировать блоки кодирования, ассоциированные с CU, из числа одной или более PU CU. Таким образом, каждая PU может быть ассоциирована с прогнозным блоком сигналов яркости и соответствующими прогнозными блоками сигнала цветности. Видеокодер 20 и видеодекодер 30 могут поддерживать PU, имеющие различные размеры. Как указано выше, размер CU может означать размер блока кодирования сигналов яркости CU, и размер PU может означать размер прогнозного блока сигналов яркости PU. При условии, что размер конкретной CU составляет 2N×2N, видеокодер 20 и видеодекодер 30 могут поддерживать PU-размеры 2N×2N или N×N для внутреннего прогнозирования и симметричные PU-размеры 2N×2N, 2N×N, N×2N, N×N или аналогичные для взаимного прогнозирования. Видеокодер 20 и видеодекодер 30 также могут поддерживать асимметричное сегментирование для PU-размеров 2N×nU, 2N×nD, nL×2N и nR×2N для взаимного прогнозирования.

Модуль 120 обработки взаимного прогнозирования может формировать прогнозирующие данные для PU посредством выполнения взаимного прогнозирования для каждой PU CU. Прогнозирующие данные для PU могут включать в себя прогнозирующие блоки, которые соответствуют PU, и информацию движения для PU. Модуль 121 взаимного прогнозирования может выполнять различные операции для PU CU в зависимости от того, находится PU в серии последовательных I-макроблоков, серии последовательных P-макроблоков или серии последовательных B-макроблоков. В серии последовательных I-макроблоков все PU внутренне прогнозируются. Следовательно, если PU находится в серии последовательных I-макроблоков, модуль 121 взаимного прогнозирования не выполняет взаимное прогнозирование для PU. Таким образом, для блоков, кодированных в I-режиме, прогнозированный блок формируется с использованием пространственного прогнозирования из ранее кодированных соседних блоков в идентичном кадре.

Если PU находится в серии последовательных P-макроблоков, модуль оценки движения модуля 120 обработки взаимного прогнозирования может выполнять поиск в опорных изображениях в списке опорных изображений (например, в RefPicList0) на предмет опорной области для PU. Опорная область для PU может представлять собой область в опорном изображении, которая содержит блоки выборок, которые наиболее близко соответствуют блокам выборок PU. Модуль оценки движения может формировать опорный индекс, который указывает позицию в RefPicList0 опорного изображения, содержащего опорную область для PU. Помимо этого, модуль оценки движения может формировать MV, который указывает пространственное смещение между блоком кодирования PU и опорным местоположением, ассоциированным с опорной областью. Например, MV может представлять собой двумерный вектор, который предоставляет смещение от координат в текущем декодированном изображении до координат в опорном изображении. Модуль оценки движения может выводить опорный индекс и MV в качестве информации движения PU. Модуль компенсации движения модуля 120 обработки взаимного прогнозирования может формировать прогнозирующие блоки PU на основе фактических или интерполированных выборок в опорном местоположении, указываемом посредством вектора движения PU.

Если PU находится в серии последовательных B-макроблоков, модуль оценки движения может выполнять унипрогнозирование или бипрогнозирование для PU. Чтобы выполнять унипрогнозирование для PU, модуль оценки движения может выполнять поиск в опорных изображениях RefPicList0 или второго списка опорных изображений (RefPicList1) на предмет опорной области для PU. Модуль оценки движения может выводить, в качестве информации движения PU, опорный индекс, который указывает позицию в RefPicList0 или RefPicList1 опорного изображения, которое содержит опорную область, MV, который указывает пространственное смещение между прогнозным блоком PU и опорным местоположением, ассоциированным с опорной областью, и один или более индикаторов направления прогнозирования, которые указывают то, находится опорное изображение в RefPicList0 или в RefPicList1. Модуль компенсации движения модуля 120 обработки взаимного прогнозирования может формировать прогнозирующие блоки PU, по меньшей мере, частично на основе фактических или интерполированных выборок в опорной области, указываемой посредством вектора движения PU.

Чтобы выполнять двунаправленное взаимное прогнозирование для PU, модуль оценки движения может выполнять поиск в опорных изображениях RefPicList0 на предмет опорной области для PU, а также может выполнять поиск в опорных изображениях RefPicList1 на предмет другой опорной области для PU. Модуль оценки движения может формировать индексы опорных изображений, которые указывают позиции в RefPicList0 и RefPicList1 опорных изображений, которые содержат опорные области. Помимо этого, модуль оценки движения может формировать MV, которые указывают пространственные смещения между опорным местоположением, ассоциированным с опорными областями, и блоком выборок PU.

Информация движения PU может включать в себя опорные индексы и MV PU. Модуль компенсации движения может формировать прогнозирующие блоки PU, по меньшей мере, частично на основе фактических или интерполированных выборок в опорных областях, указываемой посредством векторов движения PU.

В соответствии с различными примерами этого раскрытия сущности, видеокодер 20 может быть выполнен с возможностью осуществлять кодирование на основе палитр. Относительно HEVC-инфраструктуры в качестве примера, технологии кодирования на основе палитр могут быть выполнены с возможностью использоваться на уровне CU. В других примерах технологии кодирования видео на основе палитр могут быть выполнены с возможностью использоваться на уровне PU. В других примерах технологии кодирования на основе палитр могут быть выполнены с возможностью использоваться на уровне субъединиц прогнозирования (суб-PU) (например, субблока единицы прогнозирования). Соответственно, все раскрытые процессы, описанные в данном документе (в ходе этого раскрытия сущности) в контексте уровня CU, дополнительно или альтернативно, могут применяться к уровню PU или уровню суб-PU. Тем не менее, эти примеры на основе HEVC не должны считаться ограничением или недочетом технологий кодирования видео на основе палитр, описанных в данном документе, поскольку технологии в данном документе могут применяться, чтобы работать независимо либо в качестве части других существующих или будущих разрабатываемых систем/стандартов. В этих случаях единица для палитрового кодирования может представлять собой квадратные блоки, прямоугольные блоки или даже области непрямоугольной формы.

Модуль 122 кодирования на основе палитр, например, может выполнять декодирование на основе палитр, когда режим кодирования на основе палитр выбирается, например, для CU или PU. Например, модуль 122 кодирования на основе палитр может быть выполнен с возможностью формировать палитру, имеющую записи, указывающие пиксельные значения, выбирать пиксельные значения в палитре, чтобы представлять пиксельные значения, по меньшей мере, некоторых позиций блока видеоданных, и передавать в служебных сигналах информацию, ассоциирующую, по меньшей мере, некоторые позиции блока видеоданных с записями в палитре, соответствующими надлежащим образом выбранным пиксельным значениям. Хотя различные функции описываются как выполняемые посредством модуля 122 кодирования на основе палитр, некоторые или все такие функции могут выполняться посредством других модулей обработки или комбинации различных модулей обработки.

Согласно аспектам этого раскрытия сущности, модуль 122 кодирования на основе палитр может быть выполнен с возможностью осуществлять любую комбинацию технологий для палитрового кодирования, описанных в данном документе.

Например, модуль 122 кодирования на основе палитр может быть выполнен с возможностью деактивировать кодирование в палитровом режиме для любого палитрового блока, имеющего размер 64×64 или больше. В других примерах модуль 122 кодирования на основе палитр может быть выполнен с возможностью ограничивать кодирование в палитровом режиме палитровыми блоками, имеющими размер меньше 64×64, что означает то, что кодирование в палитровом режиме может активироваться или иным образом использоваться для палитровых блоков, имеющих размер меньше 64×64. В других примерах модуль 122 кодирования на основе палитр может быть выполнен с возможностью нормативно ограничивать наибольший размер палитрового блока на основе наибольшего размера единицы преобразования. В качестве другого примера, модуль 122 кодирования на основе палитр может быть выполнен с возможностью деактивировать кодирование в палитровом режиме для палитрового блока, имеющего размер, который превышает или в других отношениях больше наибольшего размера единицы преобразования. Модуль 122 кодирования на основе палитр аналогично может быть выполнен с возможностью выполнять любые другие технологии для палитрового кодирования, описанные в данном документе.

Модуль 126 обработки внутреннего прогнозирования может формировать прогнозирующие данные для PU посредством выполнения внутреннего прогнозирования для PU. Прогнозирующие данные для PU могут включать в себя прогнозирующие блоки для PU и различные синтаксические элементы. Модуль 126 обработки внутреннего прогнозирования может выполнять внутреннее прогнозирование для PU в сериях последовательных I-макроблоков, сериях последовательных P-макроблоков и сериях последовательных B-макроблоков.

Чтобы выполнять внутреннее прогнозирование для PU, модуль 126 обработки внутреннего прогнозирования может использовать несколько режимов внутреннего прогнозирования для того, чтобы формировать несколько наборов прогнозирующих данных для PU. Модуль 126 обработки внутреннего прогнозирования может использовать выборки из блоков выборок соседних PU, чтобы формировать прогнозирующий блок для PU. Соседние PU могут располагаться выше, выше и справа, выше и слева или слева от PU, при условии порядка кодирования слева направо, сверху вниз для PU, CU и CTU. Модуль 126 обработки внутреннего прогнозирования может использовать различные числа режимов внутреннего прогнозирования, например, 33 режима направленного внутреннего прогнозирования. В некоторых примерах число режимов внутреннего прогнозирования может зависеть от размера области, ассоциированной с PU.

Модуль 100 блочного кодирования может выбирать прогнозирующие данные для PU CU из числа

прогнозирующих данных, сформированных посредством модуля 120 обработки взаимного прогнозирования для PU, или прогнозирующих данных, сформированных посредством модуля 126 обработки внутреннего прогнозирования для PU. В некоторых примерах модуль 100 блочного кодирования выбирает прогнозирующие данные для PU CU на основе показателей искажения в зависимости от скорости передачи наборов прогнозирующих данных. Прогнозирующие блоки выбранных прогнозирующих данных могут упоминаться в данном документе как выбранные прогнозирующие блоки.

Модуль 102 формирования остатков может формировать, на основе блока кодирования сигналов яркости, Cb-блока кодирования и Cr-блока кодирования CU и выбранных прогнозирующих блоков сигналов яркости, прогнозирующих Cb-блоков и прогнозирующих Cr-блоков PU CU, остаточные блоки сигналов яркости, остаточные Cb-блоки и остаточные Cr-блоки CU. Например, модуль 102 формирования остатков может формировать остаточные блоки CU таким образом, что каждая выборка в остаточных блоках имеет значение, равное разности между выборкой в блоке кодирования CU и соответствующей выборкой в соответствующем выбранном прогнозирующем блоке PU CU.

Модуль 104 обработки преобразования может выполнять сегментацию на дерево квадрантов для того, чтобы сегментировать остаточные блоки, ассоциированные с CU, на блоки преобразования, ассоциированные с TU CU. Таким образом, в некоторых примерах, TU может быть ассоциирована с блоком преобразования сигналов яркости и двумя блоками преобразования сигналов цветности. Размеры и позиции блоков преобразования сигналов яркости и сигналов цветности TU CU могут быть основаны либо могут не быть основаны на размерах и позициях прогнозных блоков PU CU. Структура в виде дерева квадрантов, известная как "остаточное дерево квадрантов" (RQT), может включать в себя узлы, ассоциированные с каждой из областей. TU CU могут соответствовать концевым узлам RQT.

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

Модуль 106 квантования может квантовать коэффициенты преобразования в блоке коэффициентов. Процесс квантования может уменьшать битовую глубину, ассоциированную с некоторыми или всеми коэффициентами преобразования. Например, n-битовый коэффициент преобразования может округляться в меньшую сторону до m-битового коэффициента преобразования во время квантования, где n превышает m. Модуль 106 квантования может квантовать блок коэффициентов, ассоциированный с TU CU, на основе значения параметра квантования (QP), ассоциированного с CU. Видеокодер 20 может регулировать степень квантования, применяемого к блокам коэффициентов, ассоциированным с CU, посредством регулирования QP-значения, ассоциированного с CU. Квантование может вводить потери информации, в силу чего квантованные коэффициенты преобразования могут иметь меньшую точность по сравнению с исходными коэффициентами преобразования.

Модуль 108 обратного квантования и модуль 110 обработки обратного преобразования могут применять обратное квантование и обратные преобразования к блоку коэффициентов, соответственно, для того чтобы восстанавливать остаточный блок из блока коэффициентов. Модуль 112 восстановления может суммировать восстановленный остаточный блок с соответствующими выборками из одного или более прогнозирующих блоков, сформированных посредством модуля 100 блочного кодирования, для того чтобы формировать восстановленный блок преобразования, ассоциированный с TU. Посредством восстановления блоков преобразования для каждой TU CU таким способом, видеокодер 20 может восстанавливать блоки кодирования CU.

Модуль 114 фильтрации может выполнять одну или более операций удаления блочности, чтобы уменьшать артефакты блочности в блоках кодирования, ассоциированных с CU. Модуль 114 фильтрации может выполнять другие операции фильтрации, включающие в себя фильтрацию на основе дискретизированного адаптивного смещения (SAO) и/или адаптивную контурную фильтрацию (ALF). Буфер 116 декодированных изображений может сохранять восстановленные блоки кодирования после того, как модуль 114 фильтрации выполняет одну или более операций удаления блочности для восстановленных блоков кодирования. Модуль 120 обработки взаимного прогнозирования может использовать опорное изображение, которое содержит восстановленные блоки кодирования, для того чтобы выполнять взаимное прогнозирование для PU других изображений. Помимо этого, модуль 126 обработки внутреннего прогнозирования может использовать восстановленные блоки кодирования в буфере 116 декодированных изображений для того, чтобы выполнять внутреннее прогнозирование для других PU в изображении, идентичном изображению CU.

Модуль 118 энтропийного кодирования может принимать данные из других функциональных компонентов видеокодера 20. Например, модуль 118 энтропийного кодирования может принимать блоки коэффициентов из модуля 106 квантования и может принимать синтаксические элементы из модуля 100

блочного кодирования. Модуль 118 энтропийного кодирования может выполнять одну или более операций энтропийного кодирования для данных, чтобы формировать энтропийно кодированные данные. Например, модуль 118 энтропийного кодирования может выполнять операцию контекстно-адаптивного декодирования, к примеру, CABAC-операцию, операцию контекстно-адаптивного кодирования переменной длины (CAVLC), операцию кодирования переменно-переменной (V2V) длины, операцию синтаксического контекстно-адаптивного двоичного арифметического кодирования (SBAC), операцию энтропийного кодирования на основе сегментации на интервалы вероятности (PIPE), операцию кодирования экспоненциальным кодом Голомба или другой тип операции энтропийного кодирования для данных. Видеокодер 20 может выводить поток битов, который включает в себя энтропийно кодированные данные, сформированные посредством модуля 118 энтропийного кодирования. Например, поток битов может включать в себя данные, которые представляют RQT для CU.

В некоторых примерах остаточное кодирование не выполняется при палитровом кодировании. Соответственно, видеокодер 20 не может выполнять преобразование или квантование при кодировании с использованием режима палитрового кодирования. Помимо этого, видеокодер 20 может энтропийно кодировать данные, сформированные с использованием режима палитрового кодирования, отдельно от остаточных данных.

Согласно одной или более технологий этого раскрытия сущности видеокодер 20 и, в частности, модуль 122 кодирования на основе палитр может выполнять кодирование видео на основе палитр прогнозных видеоблоков. Как описано выше, палитра, сформированная посредством видеокодера 20, может быть явно кодирована и отправлена в видеодекодер 30, прогнозирована из предыдущих записей палитры, прогнозирована из предыдущих пиксельных значений либо как комбинация вышеозначенного.

Фиг. 3 является блок-схемой, иллюстрирующей примерный видеодекодер 30, который выполнен с возможностью выполнять технологии этого раскрытия сущности. Фиг. 3 предоставляется для целей пояснения и не является ограничением технологий, как проиллюстрировано и описано в общих чертах в этом раскрытии сущности. Для целей пояснения это раскрытие сущности описывает видеодекодер 30 в контексте HEVC-кодирования. Тем не менее, технологии этого раскрытия сущности могут быть применимыми к другим стандартам или способам кодирования.

Подробности палитрового кодирования, описанные выше относительно кодера 20, не повторяются здесь относительно декодера 30, но следует понимать, что декодер 30 может выполнять взаимнообратный процесс декодирования относительно любого процесса кодирования, описанного в данном документе относительно кодера 20.

Например, следует понимать, что видеодекодер 30 может быть выполнен с возможностью динамически деактивировать палитровый режим или выполнен с возможностью иным образом не использовать палитровый режим для любого палитрового блока, имеющего размер, который не равен или меньше наибольшей единицы преобразования. Дополнительно следует понимать, что видеодекодер 30 может быть выполнен с возможностью декодировать блок видеоданных с использованием палитрового режима только тогда, когда блок видеоданных имеет размер, который не превышает наибольшую единицу преобразования, которую видеокодер 20 может быть выполнен с возможностью кодировать, и/или которую видеодекодер 30 может быть выполнен с возможностью декодировать. Аналогично, видеодекодер 30 может быть выполнен с возможностью активировать кодирование в палитровом режиме для блока видеоданных только тогда, когда блок видеоданных имеет размер, который не превышает наибольшую единицу преобразования. В других примерах видеодекодер 30 может быть выполнен с возможностью определять то, активируется или деактивируется палитровый режим, на основе значения, соответствующего флагу палитрового режима, такого как значение для синтаксического элемента palette_mode_flag.

В качестве другого примера видеодекодер 30 может быть выполнен с возможностью принимать блок видеоданных. Видеодекодер 30 может быть выполнен с возможностью определять размер видеоданных блока относительно наибольшего размера единицы преобразования. Видеодекодер 30 может быть выполнен с возможностью определять то, что принимаемый блок видео не кодируется в палитровом режиме, когда принимаемый блок видеоданных превышает размер наибольшего размера единицы преобразования.

Видеодекодер 30 представляет пример устройства, которое может быть выполнено с возможностью осуществлять технологии для кодирования на основе палитр и энтропийного кодирования (к примеру, CABAC) в соответствии с различными примерами, описанными в этом раскрытии сущности.

В примере по фиг. 3, видеодекодер 30 включает в себя модуль 150 энтропийного декодирования, запоминающее устройство 151 видеоданных, модуль 152 блочного декодирования, модуль 154 обратного квантования, модуль 156 обработки обратного преобразования, модуль 158 восстановления, модуль 160 фильтрации и буфер 162 декодированных изображений. Модуль 152 блочного декодирования включает в себя модуль 164 компенсации движения и модуль 166 обработки внутреннего прогнозирования. Видеодекодер 30 также включает в себя модуль 165 декодирования на основе палитр, выполненный с возможностью осуществлять различные аспекты технологий кодирования на основе палитр, описанных в этом раскрытии сущности. В других примерах видеодекодер 30 может включать в себя большее, меньшее число или другие функциональные компоненты.

Запоминающее устройство 151 видеоданных может сохранять видеоданные, такие как кодированный поток видеобитов, которые должны декодироваться посредством компонентов видеодекодера 30. Видеоданные, сохраненные в запоминающем устройстве 151 видеоданных, могут получаться, например, из считываемого компьютером носителя 16, например из локального видеоисточника, такого как камера, через передачу по проводной или беспроводной сети видеоданных, либо посредством осуществления доступа к физическим носителям хранения данных. Запоминающее устройство 151 видеоданных может формировать буфер кодированных изображений (CPB), который сохраняет кодированные видеоданные из кодированного потока видеобитов. Буфер 162 декодированных изображений может представлять собой запоминающее устройство опорных изображений, которое сохраняет опорные видеоданные для использования при декодировании видеоданных посредством видеодекодера 30, например, в режимах внутреннего или взаимного кодирования. Запоминающее устройство 151 видеоданных и буфер 162 декодированных изображений могут формироваться посредством любого из множества запоминающих устройств, к примеру, как динамическое оперативное запоминающее устройство (DRAM), включающее в себя синхронное DRAM (SDRAM), магниторезистивное RAM (MRAM), резистивное RAM (RRAM) или другие типы запоминающих устройств. Запоминающее устройство 151 видеоданных и буфер 162 декодированных изображений могут предоставляться посредством идентичного запоминающего устройства или отдельных запоминающих устройств. В различных примерах, запоминающее устройство 151 видеоданных может быть внутрикристальным с другими компонентами видеодекодера 30 или внекристальным относительно этих компонентов.

Буфер кодированных изображений (CPB) может принимать и сохранять кодированные видеоданные (например, NAL-единицы) потока битов. Модуль 150 энтропийного декодирования может принимать кодированные видеоданные (например, NAL-единицы) из CPB и синтаксически анализировать NAL-единицы, чтобы декодировать синтаксические элементы. Модуль 150 энтропийного декодирования может энтропийно декодировать энтропийно кодированные синтаксические элементы в NAL-единицах. Модуль 152 блочного декодирования, модуль 154 обратного квантования, модуль 156 обработки обратного преобразования, модуль 158 восстановления и модуль 160 фильтрации могут формировать декодированные видеоданные на основе синтаксических элементов, извлеченных из потока битов.

Видеодекодер 30 может быть выполнен с возможностью осуществлять процесс, в общем, обратный относительно процесса видеокодера 20, описанного в данном документе. Аналогично, видеокодер 20 может быть выполнен с возможностью осуществлять процесс, в общем, обратный относительно процесса видеодекодера 20, описанного в данном документе. Например, такое раскрытие сущности, что видеодекодер 30 может быть выполнен с возможностью декодировать кодированный синтаксический элемент в потоке битов, аналогично обязательно раскрывает то, что видеокодер 20 может быть выполнен с возможностью кодировать синтаксический элемент в поток битов.

В качестве другого примера, модуль 150 энтропийного декодирования может быть выполнен с возможностью осуществлять процесс, в общем, обратный относительно процесса модуля 118 энтропийного кодирования, описанного в данном документе. Согласно аспектам этого раскрытия сущности, модуль 150 энтропийного декодирования может быть выполнен с возможностью энтропийно декодировать любые кодовые слова, сформированные посредством модуля 118 энтропийного кодирования.

NAL-единицы потока битов могут включать в себя NAL-единицы кодированных серий последовательных макроблоков. В качестве части декодирования потока битов, модуль 150 энтропийного декодирования может извлекать и энтропийно декодировать синтаксические элементы из NAL-единиц кодированных серий последовательных макроблоков. Каждая из кодированных серий последовательных макроблоков может включать в себя заголовок серии последовательных макроблоков и данные серии последовательных макроблоков. Заголовок серии последовательных макроблоков может содержать синтаксические элементы, связанные с серией последовательных макроблоков. Синтаксические элементы в заголовке серии последовательных макроблоков могут включать в себя синтаксический элемент, который идентифицирует PPS, ассоциированный с изображением, которое содержит серию последовательных макроблоков.

Помимо декодирования синтаксических элементов из потока битов, видеодекодер 30 может выполнять операцию восстановления для несегментированной CU. Чтобы выполнять операцию восстановления для несегментированной CU, видеодекодер 30 может выполнять операцию восстановления для каждой TU CU. Посредством выполнения операции восстановления для каждой TU CU, видеодекодер 30 может восстанавливать остаточные блоки, ассоциированные с CU.

В качестве части выполнения операции восстановления для TU CU, модуль 154 обратного квантования может обратно квантовать, т.е. деквантовать, блоки коэффициентов, ассоциированные с TU. Модуль 154 обратного квантования может использовать QP-значение, ассоциированное с CU TU, для того чтобы определять степень квантования и, аналогично, степень обратного квантования для модуля 154 обратного квантования для применения. Иными словами, коэффициент сжатия, т.е. соотношение числа битов, используемых для того, чтобы представлять исходную последовательность и сжатую последовательность, может управляться посредством регулирования значения QP, используемого при квантовании коэффициентов преобразования. Коэффициент сжатия также может зависеть от используемого способа

энтропийного кодирования.

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

Если PU кодируется с использованием внутреннего прогнозирования, модуль 166 обработки внутреннего прогнозирования может выполнять внутреннее прогнозирование для того, чтобы формировать прогнозирующие блоки для PU. Модуль 166 обработки внутреннего прогнозирования может использовать режим внутреннего прогнозирования для того, чтобы формировать прогнозирующие блоки сигналов яркости, прогнозирующие Cb-блоки и прогнозирующие Cr-блоки для PU на основе прогнозных блоков пространственно соседних PU. Модуль 166 обработки внутреннего прогнозирования может определять режим внутреннего прогнозирования для PU на основе одного или более синтаксических элементов, декодированных из потока битов.

Модуль 152 блочного декодирования может составлять первый список опорных изображений (RefPicList0) и второй список опорных изображений (RefPicList1) на основе синтаксических элементов, извлеченных из потока битов. Кроме того, если PU кодируется с использованием взаимного прогнозирования, модуль 150 энтропийного декодирования может извлекать информацию движения для PU. Модуль 164 компенсации движения может определять, на основе информации движения PU, одну или более опорных областей для PU. Модуль 164 компенсации движения может формировать, на основе блоков выборок в одном или более опорных блоков для PU, прогнозирующие блоки сигналов яркости, прогнозирующие Cb-блоки и прогнозирующие Cr-блоки для PU.

Модуль 158 восстановления может использовать блоки преобразования сигналов яркости, Cb-блоки преобразования и Cr-блоки преобразования, ассоциированные с TU CU, и прогнозирующие блоки сигналов яркости, прогнозирующие Cb-блоки и прогнозирующие Cr-блоки PU CU, т.е. либо данные внутреннего прогнозирования, либо данные взаимного прогнозирования, при соответствующих условиях, для того чтобы восстанавливать блоки кодирования сигналов яркости, Cb-блоки кодирования и Cr-блоки кодирования CU. Например, модуль 158 восстановления может суммировать выборки блоков преобразования сигналов яркости, Cb-блоков преобразования и Cr-блоков преобразования с соответствующими выборками прогнозирующих блоков сигналов яркости, прогнозирующих Cb-блоков и прогнозирующих Cr-блоков для того, чтобы восстанавливать блоки кодирования сигналов яркости, Cb-блоки кодирования и Cr-блоки кодирования CU.

Модуль 160 фильтрации может выполнять операцию удаления блочности, чтобы уменьшать артефакты блочности, ассоциированные с блоками кодирования сигналов яркости, Cb-блоками кодирования и Cr-блоками кодирования CU. Видеодекодер 30 может сохранять блоки кодирования сигналов яркости, Cb-блоки кодирования и Cr-блоки кодирования CU в буфере 162 декодированных изображений. Буфер 162 декодированных изображений может предоставлять опорные изображения для последующей компенсации движения, внутреннего прогнозирования и представления на устройстве отображения, к примеру, на устройстве 32 отображения по фиг. 1. Например, видеодекодер 30 может выполнять, на основе блоков сигналов яркости, Cb-блоков и Cr-блоков в буфере 162 декодированных изображений, операции внутреннего прогнозирования или взаимного прогнозирования для PU других CU.

В соответствии с различными примерами этого раскрытия сущности, видеодекодер 30 может быть выполнен с возможностью осуществлять кодирование на основе палитр. Модуль 165 декодирования на основе палитр, например, может выполнять декодирование на основе палитр, когда режим декодирования на основе палитр выбирается, например, для CU или PU. Например, модуль 165 декодирования на основе палитр может быть выполнен с возможностью формировать палитру, имеющую записи, указывающие пиксельные значения, принимать информацию, ассоциирующую, по меньшей мере, некоторые пиксельные местоположения в блоке видеоданных с записями в палитре, выбирать пиксельные значения в палитре на основе информации и восстанавливать пиксельные значения блока на основе значений выбранного пиксела в палитре. Хотя различные функции описываются как выполняемые посредством модуля 165 декодирования на основе палитр, некоторые или все такие функции могут выполняться посредством других модулей обработки или комбинации различных модулей обработки.

Модуль 165 декодирования на основе палитр может принимать информацию режима палитрового кодирования и выполнять вышеуказанные операции, когда информация режима палитрового кодирования указывает то, что режим палитрового кодирования применяется к блоку. Когда информация режима палитрового кодирования указывает то, что режим палитрового кодирования не применяется к блоку, либо когда информация других режимов указывает использование другого режима, модуль 165 декодирования на основе палитр декодирует блок видеоданных с использованием режима кодирования не на основе палитр, например, такого как режим взаимного прогнозирующего или внутреннего прогнозирующего HEVC-кодирования. Блок видеоданных, например, может представлять собой CU или PU, сформированную согласно процессу HEVC-кодирования. Режим кодирования на основе палитр может

содержать один из множества различных режимов кодирования на основе палитр, либо может быть предусмотрен один режим кодирования на основе палитр.

Согласно аспектам этого раскрытия сущности, модуль 165 декодирования на основе палитр может быть выполнен с возможностью осуществлять любую комбинацию технологий для палитрового кодирования, описанных в данном документе. Подробности палитрового кодирования, описанные выше относительно кодера 20, не повторяются здесь относительно декодера 30, но следует понимать, что декодер 30 может выполнять взаимно-обратный процесс декодирования на основе палитр относительно любого процесса кодирования на основе палитр, описанного в данном документе относительно кодера 20.

Фиг. 4 является концептуальной схемой, иллюстрирующей пример определения палитры для кодирования видеоданных, в соответствии с технологиями этого раскрытия сущности. Пример по фиг. 4 включает в себя изображение 178, имеющее первую единицу 180 PAL (палитрового) кодирования (CU), которая ассоциирована с первыми палитрами 184, и вторую PAL CU 188, которая ассоциирована со вторыми палитрами 192. Как подробнее описано ниже и в соответствии с технологиями этого раскрытия сущности, вторые палитры 192 основаны на первых палитрах 184. Изображение 178 также включает в себя блок 196, кодированный с использованием режима внутреннего прогнозирующего кодирования, и блок 200, который кодируется с использованием режима взаимного прогнозирующего кодирования.

Технологии по фиг. 4 описываются в контексте видеокодера 20 (фиг. 1 и 2) и видеодекодера 30 (фиг. 1 и 3) и относительно стандарта HEVC-кодирования видео для целей пояснения. Тем не менее, следует понимать, что технологии этого раскрытия сущности не ограничены в этом отношении и могут применяться посредством других процессоров и/или устройств кодирования видео в других процессах и/или стандартах кодирования видео.

В общем, палитра означает число пиксельных значений, которые являются доминирующими и/или характерными для CU, в данный момент кодируемой, CU 188 в примере по фиг. 4. Первые палитры 184 (которые также могут упоминаться как индексы 184) и вторые палитры 192 (которые также могут упоминаться как индексы 192) показаны как включающие в себя несколько палитр (которые также могут упоминаться как несколько индексов). В некоторых примерах согласно аспектам этого раскрытия сущности, видеокодер (к примеру, видеокодер 20 или видеодекодер 30) может кодировать палитры (например, индексы) отдельно для каждого цветового компонента CU. Например, видеокодер 20 может кодировать палитру для компонента сигнала (Y) яркости CU, другую палитру для компонента сигнала (U) цветности CU и еще одну другую палитру для компонента (V) сигнала цветности CU. В этом примере записи Y-палитры могут представлять Y-значения пикселов CU, записи U-палитры могут представлять U-значения пикселов CU, и записи V-палитры могут представлять V-значения пикселов CU.

В других примерах видеокодер 20 может кодировать одну палитру для всех цветовых компонентов CU. В этом примере видеокодер 20 может кодировать палитру, имеющую i-ю запись, которая является тройным значением, включающим в себя Y_i, U_i и V_i. В этом случае, палитра включает в себя значения для каждого из компонентов пикселов. Соответственно, представление палитр 184 и 192 в качестве набора палитр, имеющего несколько отдельных палитр, является просто одним примером и не служит в качестве ограничения.

В примере по фиг. 4 первые палитры 184 включают в себя три записи 202-206, имеющие значение 1 индекса записи, значение 2 индекса записи и значение 3 индекса записи соответственно. Первые палитры 184 связывают значения индекса (например, значения, показанные в левом столбце первых палитр 184) с пиксельными значениями. Например, как показано на фиг. 4, одна из первых палитр 184 связывает значения 1, 2 и 3 индекса с пиксельными значениями A, B и C соответственно. Как описано в данном документе, вместо кодирования фактических пиксельных значений первой CU 180, видеокодер (к примеру, видеокодер 20 или видеодекодер 30) может использовать кодирование на основе палитр для того, чтобы кодировать пикселы блока с использованием индексов 1-3 (которые также могут выражаться как значения 1-3 индекса). Иными словами, для каждой пиксельной позиции первой CU 180, видеокодер 20 может кодировать значение индекса для пиксела, при этом значение индекса ассоциировано с пиксельным значением в одной или более первых палитр 184. Видеодекодер 30 может получать значения индекса из потока битов и восстанавливать пиксельные значения с использованием значений индекса и одной или более первых палитр 184. Таким образом, первые палитры 184 передаются посредством видеокодера 20 в потоке битов кодированных видеоданных для использования посредством видеодекодера 30 в декодировании на основе палитр.

В некоторых примерах видеокодер 20 и видеодекодер 30 могут определять вторые палитры 192 на основе первых палитр 184. Например, видеокодер 20 и/или видеодекодер 30 могут находить один или более блоков, из которых определяются прогнозирующие палитры, в этом примере, первые палитры 184. В некоторых примерах таких как пример, проиллюстрированный на фиг. 4, видеокодер 20 и/или видеодекодер 30 могут находить ранее кодированную CU, к примеру, левую соседнюю CU (первую CU 180) при определении прогнозирующей палитры для второй CU 188.

В примере по фиг. 4 вторые палитры 192 включают в себя три записи 208-212, имеющие значение 1 индекса записи, значение 2 индекса записи и значение 3 индекса записи соответственно. Вторые палитры 192 связывают значения индекса (например, значения, показанные в левом столбце первых палитр 192) с

пиксельными значениями. Например, как показано на фиг. 4, одна из вторых палитр 192 связывает значения 1, 2 и 3 индекса с пиксельными значениями A, B и D соответственно. В этом примере видеокодер 20 может кодировать один или более синтаксических элементов, указывающих то, какие записи первых палитр 184 включены во вторые палитры 192. В примере по фиг. 4, один или более синтаксических элементов проиллюстрированы в качестве вектора 216. Вектор 216 имеет определенное число ассоциированных элементов выборки (или битов), при этом каждый элемент выборки указывает то, используется или нет предиктор палитры, ассоциированный с этим элементом выборки, для того чтобы прогнозировать запись текущей палитры. Например, вектор 216 указывает то, что первые две записи первых палитр 184 (202 и 204) включены во вторые палитры 192 (значение "1" в векторе 216), тогда как третья запись первых палитр 184 не включена во вторые палитры 192 (значение "0" в векторе 216) . В примере по фиг. 4, вектор является булевым вектором.

В некоторых примерах видеокодер 20 и видеодекодер 30 могут определять список предикторов палитр (который также может упоминаться как таблица предикторов палитр) при выполнении палитрового прогнозирования. Список предикторов палитр может включать в себя записи из палитр одного или более соседних блоков, которые используются для того, чтобы прогнозировать одну или более записей палитры для кодирования текущего блока. Видеокодер 20 и видеодекодер 30 могут составлять список таким же образом. Видеокодер 20 и видеодекодер 30 могут кодировать данные (к примеру, вектор 216), чтобы указывать то, какие записи списка предикторов палитр должны быть включены в палитру для кодирования текущего блока.

Фиг. 5 является концептуальной схемой, иллюстрирующей примеры определения индексов для палитры для видеоблока, в соответствии с технологиями этого раскрытия сущности. Например, фиг. 5 включает в себя индексный блок 240 (который также может упоминаться как карта 240 или карта 240 индексов), включающий в себя значения индекса (например, значения 1, 2 и 3 индекса), которые связывают соответствующие позиции пикселов, ассоциированных со значениями индекса, с записью палитр 244.

Хотя индексный блок 240 проиллюстрирован в примере по фиг. 5 как включающий в себя значение индекса для каждой пиксельной позиции, следует понимать, что в других примерах не все пиксельные позиции могут быть ассоциированы со значением индекса, связывающим пиксельное значение с записью палитр 244. Иными словами, как отмечено выше, в некоторых примерах, видеокодер 20 может кодировать (и видеодекодер 30 может получать из кодированного потока битов) индикатор относительно фактического пиксельного значения (или его квантованной версии) для позиции в индексном блоке 240, если пиксельное значение не включено в палитры 244.

В некоторых примерах видеокодер 20 и видеодекодер 30 могут быть выполнены с возможностью кодировать дополнительную карту, указывающую то, какие пиксельные позиции ассоциированы с какими значениями индекса. Например, допустим, что запись (i, j) в индексном блоке 240 соответствует позиции (i, j) CU. Видеокодер 20 может кодировать один или более синтаксических элементов для каждой записи индексного блока (т.е. каждой пиксельной позиции), указывающих то, имеет или нет запись ассоциированное значение индекса. Например, видеокодер 20 может кодировать флаг, имеющий значение в единицу, чтобы указывать то, что пиксельное значение в местоположении (i, j) в CU является одним из значений в палитрах 244.

Видеокодер 20 в таком примере может также кодировать палитру (показанную в примере по фиг. 5 в качестве 244). В случаях, в которых палитры 244 включают в себя одну запись и ассоциированное пиксельное значение, видеокодер 20 может пропускать передачу в служебных сигналах значения индекса. Видеокодер 20 может кодировать флаг таким образом, что он имеет значение в нуль, чтобы указывать то, что пиксельное значение в местоположении (i, j) в CU не является одним из значений в палитрах 244. В этом примере видеокодер 20 также может кодировать индикатор относительно пиксельного значения для использования посредством видеодекодера 30 при восстановлении пиксельного значения. В некоторых случаях, пиксельное значение может кодироваться с потерями.

Значение пиксела в одной позиции CU может предоставлять индикатор относительно значений одного или более других пикселов в других позициях CU. Например, может быть относительно высокая вероятность того, что позиции соседнего пиксела CU имеют идентичное пиксельное значение или могут преобразовываться в идентичное значение индекса (в случае кодирования с потерями, в котором более одного пиксельного значения могут преобразовываться в одно значение индекса).

Соответственно, видеокодер 20 может кодировать один или более синтаксических элементов, указывающих число последовательных пикселов или значений индекса в данном порядке сканирования, которые имеют идентичное значение пиксельного значения или индекса. Как отмечено выше, строка подобнозначных пиксельных значений или значений индекса может упоминаться в данном документе как серия. В примере для целей иллюстрации, если два последовательных пиксела или индекса в данном порядке сканирования имеют различные значения, серия равна нулю. Если два последовательных пиксела или индекса в данном порядке сканирования имеют идентичное значение, но третий пиксел или индекс в порядке сканирования имеет другое значение, серия равна единице. Для трех последовательных индексов или пикселов с идентичным значением серия равна двум и т.д. Видеодекодер 30 может получать синтаксические элементы, указывающие серию из кодированного потока битов, и использовать

данные для того, чтобы определять число последовательных местоположений, которые имеют идентичное пиксельное значение или значение индекса.

В некоторых примерах в соответствии с технологиями этого раскрытия сущности, модуль 118 энтропийного кодирования и модуль 150 энтропийного декодирования могут быть выполнены с возможностью энтропийно кодировать индексный блок 240. Например, модуль 118 кодирования и модуль 150 энтропийного декодирования могут быть выполнены с возможностью энтропийно кодировать длины серий (например, значения или коды по длинам серий) и/или двоичный вектор палитрового прогнозирования, связанный с индексным блоком в палитровом режиме.

Фиг. 6 является концептуальной схемой, иллюстрирующей пример определения максимальной длины серии с копированием сверху при условии примера порядка растрового сканирования в соответствии с технологиями этого раскрытия сущности. В примере по фиг. 6, если ни один из пикселов, обведенных посредством пунктирных линий 280, не кодируется в качестве выборки с управляющим кодом, максимальная возможная длина серии равна 35 (т.е. числу незаштрихованных пиксельных позиций). Если один или более пикселов внутри пунктирных линий 280 кодируются в качестве выборки с управляющим кодом, при условии, что пиксел, помеченный в качестве пиксела с управляющим кодом (пиксельная позиция с "X"), является первым пикселом с управляющим кодом внутри пунктирных линий 280 в порядке сканирования, то максимальная возможная кодированная длина серии с копированием сверху равна пяти.

В некоторых примерах видеодекодер 30 может определять режим серий (например, палитровый режим, в котором пикселы кодируются) только для пикселов внутри пунктирных линий 280. Следовательно, в наихудшем случае, видеодекодер 30 выполняет определение для BlockWidth-1 пикселов. В некоторых примерах видеодекодер 30 может быть выполнен с возможностью реализовывать определенные ограничения относительно максимума числа пикселов, для которых проверяется режим серий. Например, видеодекодер 30 может проверять только пикселы внутри пунктирных линий 280, если пикселы находятся в строке, идентичной строке текущего пиксела. Видеодекодер 30 может логически выводить, что все другие пикселы внутри пунктирных линий 280 не кодируются в качестве выборок с управляющим кодом. Пример на фиг. 6 допускает порядок растрового сканирования. Тем не менее, технологии могут применяться к другим порядкам сканирования, таким как вертикальный, горизонтальный пересекающийся и вертикальный пересекающийся.

Фиг. 7 является блок-схемой последовательности операций способа, иллюстрирующей примерный процесс для кодирования видеоданных в соответствии с технологиями этого раскрытия сущности. Процесс по фиг. 7, в общем, описывается как выполняемый посредством видеокодера (например, видеокодера 20) в целях иллюстрации, хотя множество других процессоров также может выполнять процесс, показанный на фиг. 7. В некоторых примерах модуль 100 блочного кодирования, модуль 122 кодирования на основе палитр и/или модуль 118 энтропийного кодирования могут выполнять один или более процессов, показанных на фиг. 7.

В примере по фиг. 7 видеокодер (например, видеокодер 20) может быть выполнен с возможностью принимать блок видеоданных, имеющий размер (700). Видеокодер может быть выполнен с возможностью определять размер блока видеоданных (702). Видеокодер может быть выполнен с возможностью деактивировать кодирование в палитровом режиме для блока видеоданных на основе определенного размера блока видеоданных (704).

В некоторых примерах видеокодер может быть выполнен с возможностью ограничивать кодирование в палитровом режиме любым блоком видеоданных, имеющим первый размер, меньший второго размера. В некоторых примерах первый размер может составлять 32×32. В некоторых примерах второй размер может составлять 64×64. В таких примерах видеокодер может быть выполнен с возможностью ограничивать палитровый режим любым блоком видеоданных, имеющим первый размер меньше 64×64. В некоторых примерах первый размер может составлять 32×32, и второй размер может составлять 64×64.

В некоторых примерах видеокодер может быть выполнен с возможностью ограничивать кодирование в палитровом режиме любым блоком видеоданных, имеющим первый размер, меньший или равный размеру наибольшей единицы преобразования, указываемой для видеоданных. Размер наибольшей единицы преобразования может составлять 32×32. В таком примере видеокодер может быть выполнен с возможностью ограничивать кодирование в палитровом режиме любым блоком видеоданных, имеющим первый размер, меньший или равный 32×32.

В некоторых примерах видеокодер может быть выполнен с возможностью разделять блок видеоданных на множество субблоков 4×4. В таких примерах видеокодер может быть выполнен с возможностью кодировать, с использованием палитрового режима, множество субблоков 4×4.

В некоторых примерах видеокодер может быть выполнен с возможностью ограничивать любое значение длины серии при кодировании в палитровом режиме значением максимальной длины серии таким образом, что кодирование в палитровом режиме деактивируется для блока видеоданных на основе определенного размера блока видеоданных, только если любое значение длины серии при кодировании в палитровом режиме не ограничивается значением максимальной длины серии. В одном примере значение

максимальной длины серии составляет 32×32 минус 1. В другом примере, максимальная длина серии основана на размере наибольшей единицы преобразования. В этом примере если размер наибольшей единицы преобразования составляет 32×32, то максимальная длина серии может составлять меньше 32×32, к примеру, 32×32 минус 1. В другом примере, максимальная длина серии основана на числе коэффициентов в наибольшей единице преобразования.

Следует понимать, что все технологии, описанные в данном документе, могут использоваться отдельно или в комбинации. Например, видеокодер 20 и/или один или более его компонентов и видеодекодер 30 и/или один или более его компонентов могут выполнять технологии, описанные в этом раскрытии сущности, в любой комбинации.

Следует признавать, что в зависимости от примера определенные этапы или события любой из технологий, описанных в данном документе, могут выполняться в другой последовательности, могут добавляться, объединяться или вообще исключаться (например, не все описанные этапы или события требуются для практической реализации технологий). Кроме того, в определенных примерах, этапы или события могут выполняться одновременно, например, посредством многопоточной обработки, обработки прерывания или посредством нескольких процессоров, а не последовательно. Помимо этого, хотя конкретные аспекты этого раскрытия сущности описываются как выполняемые посредством одного модуля или блока для понятности, следует понимать, что технологии этого раскрытия сущности могут выполняться посредством комбинации блоков или модулей, ассоциированных с видеокодером.

Конкретные аспекты этого раскрытия сущности описаны относительно разрабатываемого HEVC-стандарта для целей иллюстрации. Тем не менее, технологии, описанные в этом раскрытии сущности, могут быть полезными для других процессов кодирования видео, включающих в себя другие стандартные или собственные процессы кодирования видео, еще не разработанные.

Технологии, описанные выше, могут выполняться посредством видеокодера 20 (фиг. 1 и 2) и/или видеодекодера 30 (фиг. 1 и 3), оба из которых, в общем, могут упоминаться в качестве видеокодера. Аналогично, кодирование видео может означать кодирование видео или декодирование видео, при соответствующих условиях.

В соответствии с этим раскрытием сущности термин "или" может прерываться как "и/или", если контекст не предписывает иное. Дополнительно, хотя такие фразы, как "один или более" или "по меньшей мере один" и т.п. могут использоваться для некоторых признаков, раскрытых в данном документе, но не для других; признаки, для которых такой язык не используется, могут интерпретироваться как имеющие такой смысл подразумеваемым, если контекст не предписывает иное.

Хотя выше описываются конкретные комбинации различных аспектов технологий, эти комбинации предоставляются просто для того, чтобы иллюстрировать примеры технологий, описанных в этом раскрытии сущности. Соответственно, технологии этого раскрытия сущности не должны быть ограничены этими примерными комбинациями и могут охватывать любую возможную комбинацию различных аспектов технологий, описанных в этом раскрытии сущности.

В одном или более примеров, описанные функции могут быть реализованы в аппаратных средствах, программном обеспечении, микропрограммном обеспечении или любой комбинации вышеозначенного. При реализации в программном обеспечении функции могут сохраняться или передаваться, в качестве одной или более инструкций или кода, по считываемому компьютером носителю и выполняться посредством аппаратного модуля обработки. Считываемые компьютером носители могут включать в себя считываемые компьютером носители хранения данных, которые соответствуют материальному носителю, такие как носители хранения данных, или среды связи, включающие в себя любой носитель, который упрощает перенос компьютерной программы из одного места в другое, например, согласно протоколу связи. Таким образом, считываемые компьютером носители, в общем, могут соответствовать (1) материальному считываемому компьютером носителю хранения данных, который является долговременным, или (2) среде связи, такой как сигнал или несущая. Носители хранения данных могут представлять собой любые доступные носители, к которым может осуществляться доступ посредством одного или более компьютеров или одного или более процессоров, с тем чтобы извлекать инструкции, код и/или структуры данных для реализации технологий, описанных в этом раскрытии сущности. Компьютерный программный продукт может включать в себя считываемый компьютером носитель.

В качестве примера, а не ограничения, эти считываемые компьютером носители хранения данных могут содержать RAM, ROM, EEPROM, CD-ROM или другое устройство хранения данных на оптических дисках, устройство хранения данных на магнитных дисках или другие магнитные устройства хранения, флэш-память либо любой другой носитель, который может быть использован для того, чтобы сохранять требуемый программный код в форме инструкций или структур данных, и к которому можно осуществлять доступ посредством компьютера. Кроме того, любое соединение корректно называть считываемым компьютером носителем. Например, если инструкции передаются из веб-узла, сервера или другого удаленного источника с помощью коаксиального кабеля, оптоволоконного кабеля, "витой пары", цифровой абонентской линии (DSL) или беспроводных технологий, таких как инфракрасные, радиопередающие и микроволновые среды, то коаксиальный кабель, оптоволоконный кабель, "витая пара", DSL

или беспроводные технологии, такие как инфракрасные, радиопередающие и микроволновые среды, включаются в определение носителя. Тем не менее, следует понимать, что считываемые компьютером носители хранения данных и носители хранения данных не включают в себя соединения, несущие, сигналы или другие энергозависимые носители, а вместо этого направлены на долговременные материальные носители хранения данных. Диск (disk) и диск (disc) при использовании в данном документе включают в себя компакт-диск (CD), лазерный диск, оптический диск, универсальный цифровой диск (DVD), гибкий диск и диск Blu-Ray, при этом диски (disk) обычно воспроизводят данные магнитно, тогда как диски (disc) обычно воспроизводят данные оптически с помощью лазеров. Комбинации вышеперечисленного также следует включать в число считываемых компьютером носителей.

Инструкции могут выполняться посредством одного или более процессоров, например, одного или более процессоров цифровых сигналов (DSP), микропроцессоров общего назначения, специализированных интегральных схем (ASIC), программируемых пользователем вентильных матриц (FPGA) либо других эквивалентных интегральных или дискретных логических схем. Соответственно, термин "процессор" при использовании в данном документе может означать любую вышеуказанную структуру или другую структуру, подходящую для реализации технологий, описанных в данном документе. Помимо этого, в некоторых аспектах функциональность, описанная в данном документе, может быть предоставлена в рамках специализированных аппаратных и/или программных модулей, выполненных с возможностью кодирования или декодирования либо встроенных в комбинированный кодек. Кроме того, технологии могут быть полностью реализованы в одной или более схем или логических элементов.

Технологии этого раскрытия сущности могут быть реализованы в широком спектре устройств или приборов, в том числе в беспроводном переносном телефоне, в интегральной схеме (IC) или в наборе IC (к примеру, в наборе микросхем). Различные компоненты, модули или блоки описываются в этом раскрытии сущности для того, чтобы подчеркивать функциональные аспекты устройств, выполненных с возможностью осуществлять раскрытые технологии, но необязательно требуют реализации посредством различных аппаратных модулей. Наоборот, как описано выше, различные блоки могут быть комбинированы в аппаратный модуль кодека или предоставлены посредством набора взаимодействующих аппаратных модулей, включающих в себя один или более процессоров, как описано выше, в сочетании с надлежащим программным обеспечением и/или микропрограммным обеспечением.

В данном документе описаны различные примеры. Предполагается любая комбинация описанных систем, операций, функций или систем. Эти и другие примеры находятся в пределах объема прилагаемой формулы изобретения.

ФОРМУЛА ИЗОБРЕТЕНИЯ

1. Способ кодирования видеоданных согласно стандарту видеокодирования HEVC или любому другому стандарту видеокодирования в семействе стандартов H.26x или согласно любым расширениям этих стандартов, при этом способ содержит этапы, на которых

принимают блок видеоданных, имеющий размер;

определяют размер блока видеоданных

когда определенный размер блока видеоданных меньше или равен размеру наибольшей единицы преобразования, осуществляют кодирование блока видеоданных в палитровом режиме при одновременном ограничении максимального значения длины серии для этого блока значением длины серии, определяемым как размер упомянутой наибольшей единицы преобразования минус 1, при этом размер наибольшей единицы преобразования составляет 32×32; или

когда определенный размер блока видеоданных превышает размер наибольшей единицы преобразования, деактивируют кодирование в палитровом режиме для этого блока видеоданных или разделяют этот блок на субблоки и кодируют эти субблоки в палитровом режиме.

2. Устройство для кодирования видеоданных согласно стандарту видеокодирования HEVC или любому другому стандарту видеокодирования в семействе стандартов H.26x или согласно любым расширениям этих стандартов, при этом устройство содержит

запоминающее устройство, выполненное с возможностью сохранять видеоданные; и

видеокодер, поддерживающий связь с запоминающим устройством, причем видеокодер выполнен с возможностью

принимать блок видеоданных, имеющий размер, из запоминающего устройства;

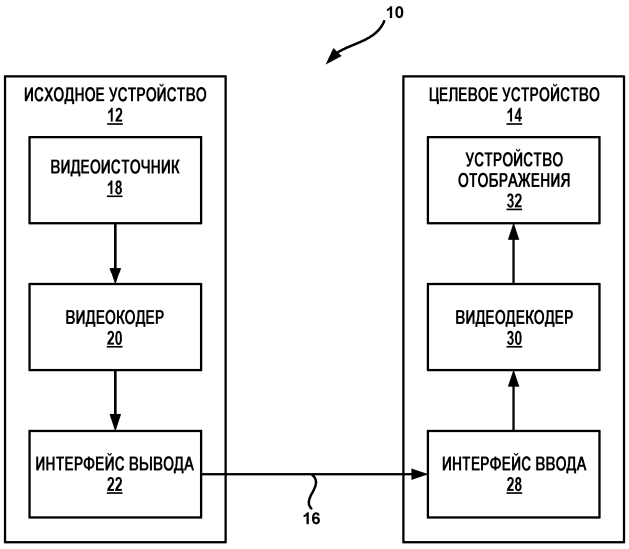
определять размер блока видеоданных

когда определенный размер блока видеоданных меньше или равен размеру наибольшей единицы преобразования, осуществлять кодирование блока видеоданных в палитровом режиме при одновременном ограничении максимального значения длины серии для этого блока значением длины серии, определяемым как размер упомянутой наибольшей единицы преобразования минус 1, при этом размер наибольшей единицы преобразования составляет 32×32; или

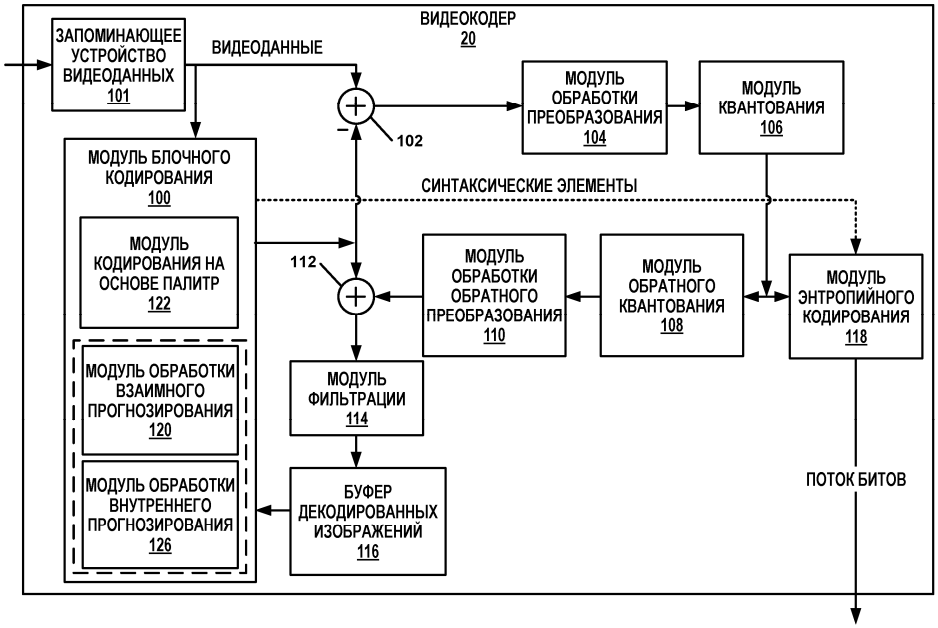
когда определенный размер блока видеоданных превышает размер наибольшей единицы преобразования, деактивировать кодирование в палитровом режиме для блока видеоданных или разделять этот

блок на субблоки и кодировать эти субблоки в палитровом режиме.

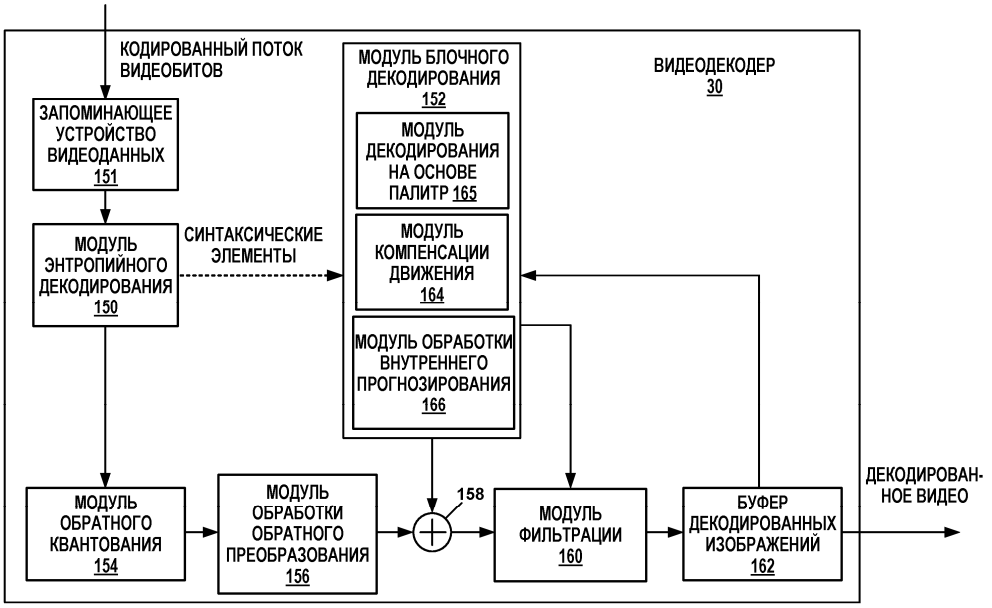
3. Долговременный считываемый компьютером носитель хранения данных, имеющий сохраненные инструкции, которые при выполнении инструктируют одному или более процессорам выполнить способ кодирования видеоданных по п.1.



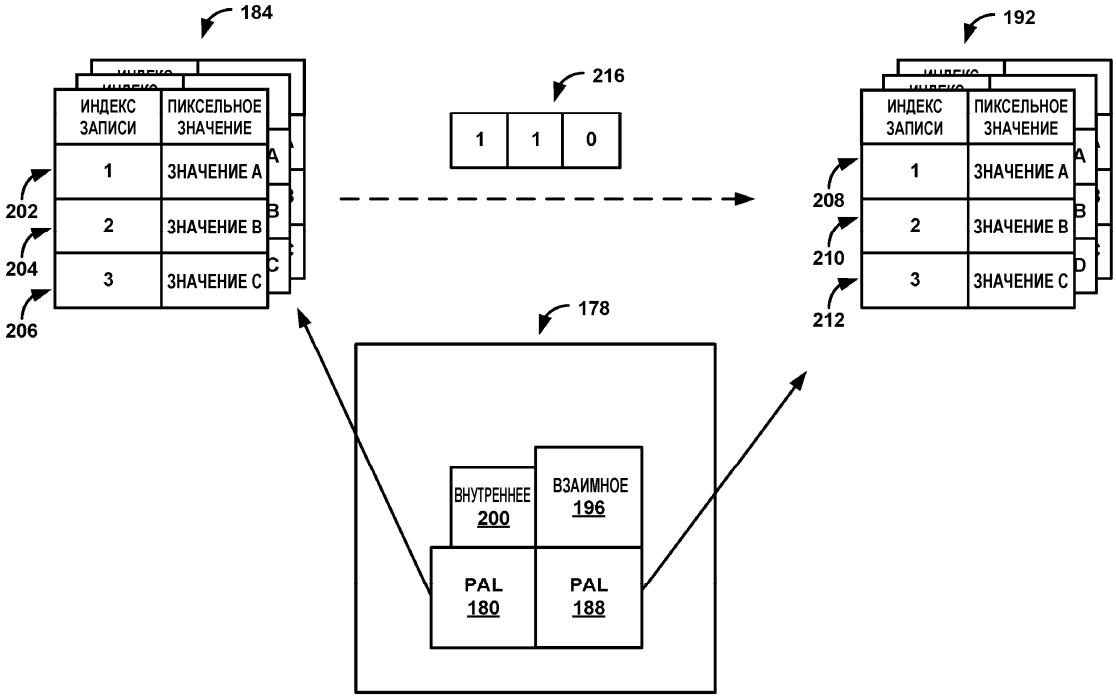
Фиг. 1



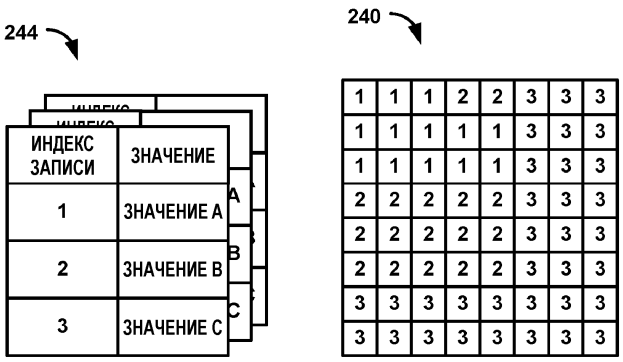
Фиг. 2



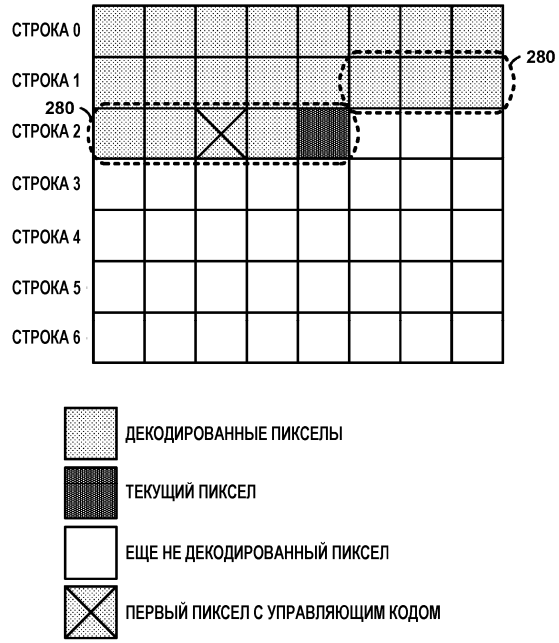
Фиг. 3



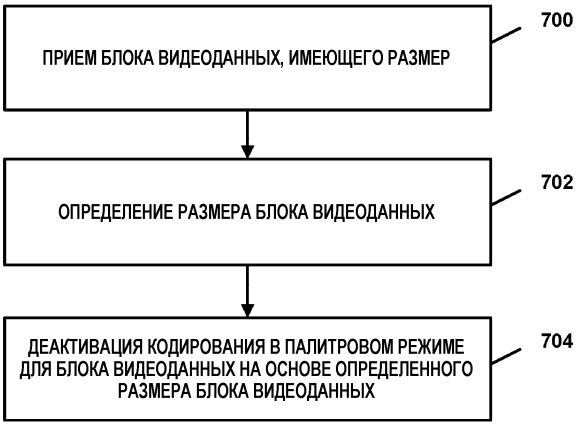
Фиг. 4



Фиг. 5


СТРОКА 0
СТРОКА 1
СТРОКА 2
СТРОКА 3
СТРОКА 4
СТРОКА 5
СТРОКА 6
280
280
ДЕКОДИРОВАННЫЕ ПИКСЕЛЫ
ТЕКУЩИЙ ПИКСЕЛ
ЕЩЕ НЕ ДЕКОДИРОВАННЫЙ ПИКСЕЛ
ПЕРВЫЙ ПИКСЕЛ С УПРАВЛЯЮЩИМ КОДОМ

Фиг. 6


ПРИЕМ БЛОКА ВИДЕОДАННЫХ, ИМЕЮЩЕГО РАЗМЕР
700
ОПРЕДЕЛЕНИЕ РАЗМЕРА БЛОКА ВИДЕОДАННЫХ
702
ДЕАКТИВАЦИЯ КОДИРОВАНИЯ В ПАЛИТРОВОМ РЕЖИМЕ ДЛЯ БЛОКА ВИДЕОДАННЫХ НА ОСНОВЕ ОПРЕДЕЛЕННОГО РАЗМЕРА БЛОКА ВИДЕОДАННЫХ
704

Фиг. 7