

CODIFICACIÓN DE ÍNDICES DE RECORTE PARA FILTRO EN BUCLE ADAPTATIVO EN LA CODIFICACIÓN DE VIDEO

[0001] Esta solicitud reivindica prioridad a la solicitud estadounidense N.º 16/897.049, presentada el 9 de junio de 2020, que reivindica el beneficio de la solicitud provisional estadounidense N.º 62/859.948, presentada el 11 de junio 2019, cuyos contenidos completos se incorporan por referencia.

CAMPO TÉCNICO

[0002] Esta divulgación se refiere a la codificación y decodificación de video.

ANTECEDENTES

[0003] Se pueden incorporar capacidades de video digitales en un amplio rango de dispositivos, incluso televisores digitales, sistemas de transmisión directa digital, sistemas de transmisión inalámbrica, asistentes digitales personales (PDA), computadoras portátiles o de escritorio, computadora tipo *tablet*, lectores de libros electrónicos, cámaras digitales, dispositivos de registro digital, reproductores de medios digitales, dispositivos de juegos de video, consolas de juegos de video, teléfonos celulares o de radio satélite, los denominados "teléfonos inteligentes", dispositivos de teleconferencia de video, dispositivos de *streaming* de video y similares. Los dispositivos de video digital implementan técnicas de codificación de video, tales como aquellas descritas en los estándares definidos por MPEG-2, MPEG-4, ITU-T H.263, ITU-T H.264/MPEG-4, parte 10, codificación de video avanzada (AVC), ITU-T H.265/codificación de video de alta eficiencia (HEVC) y las extensiones de estos estándares. Los dispositivos de video pueden transmitir, recibir, codificar, decodificar y/o almacenar información de video digital de manera más eficaz implementando las técnicas de codificación de video.

[0004] Las técnicas de codificación de video incluyen la predicción espacial (intraimagen) y/o la predicción temporal (interimagen) para reducir o eliminar la redundancia inherente en secuencias de video. Para la codificación de video basada en bloques, un plano de video (por ejemplo, una imagen de video o una porción de una imagen de video) se pueden dividir en bloques de video, que también se pueden denominar unidades de árboles de codificación (CTU), unidades de codificación (CU) y/o nodos de codificación. Los bloques de video en una porción intracodificada (I) de una imagen se codifican mediante el uso de predicción espacial con respecto a las muestras de referencia en bloques vecinos de la misma imagen. Los bloques de video

en una porción intercodificada (P o B) de la imagen pueden usar predicción espacial con respecto a las muestras de referencia en bloques vecinos de la misma imagen o predicción temporal con respecto a las muestras de referencia en otras imágenes de referencia. Las imágenes pueden denominarse cuadros, y las imágenes de referencia pueden denominarse cuadros de referencia.

SÍNTESIS

[0005] En general, esta divulgación describe técnicas para señalar índices de recorte para filtros en bucle adaptativo (ALF) en la codificación de video. Las técnicas de esta divulgación se pueden aplicar a los códecs de video existentes, tales como los códecs que cumplen con el estándar ITU-T H.265, codificación de video de alta eficiencia (HEVC), o se pueden usar como una herramienta de codificación en un estándar que se está desarrollando actualmente, tal como codificación de video versátil (VVC) y otros estándares de codificación de video futuros.

[0006] En un ejemplo, un método para codificar datos de video incluye codificar un índice de recorte de filtro en bucle adaptativo (ALF) como uno de: un número entero sin signo de longitud fija, un valor binario truncado, un valor unario truncado o un valor codificado mediante Golomb exponencial de orden 0 sin signo; y aplicar, en función del índice de recorte de ALF, un ALF a un bloque de una imagen de los datos de video.

[0007] En otro ejemplo, un dispositivo para codificar datos de video incluye una memoria configurada para almacenar datos de video y uno o más circuitos de procesamiento configurados para codificar un índice de recorte de ALF como uno de: un número entero sin signo de longitud fija, un valor binario truncado, un valor unario truncado o un valor codificado mediante Golomb exponencial de orden 0 sin signo; y aplicar, en función del índice de recorte de ALF, un ALF a un bloque de una imagen de los datos de video.

[0008] En otro ejemplo, un dispositivo para codificar datos de video incluye medios para codificar un índice de recorte de ALF como uno de: un número entero sin signo de longitud fija, un valor binario truncado, un valor unario truncado o un valor codificado mediante Golomb exponencial de orden 0 sin signo; y medios para aplicar, en función del índice de recorte de ALF, un ALF a un bloque de una imagen de los datos de video.

[0009] En otro ejemplo, un medio de almacenamiento legible por computadora está codificado con instrucciones que, cuando se ejecutan, hacen que un procesador programable codifique un índice de recorte de ALF como uno de: un número entero sin signo de longitud fija, un valor binario truncado, un valor unario truncado o un valor codificado mediante Golomb exponencial de orden 0 sin signo; y aplique, en función del índice de recorte de ALF, un ALF a un bloque de una imagen de los datos de video.

[0010] Los detalles de uno o más ejemplos se exponen en las figuras adjuntas y en la descripción siguiente. Otras características, objetivos y ventajas serán evidentes a partir de la descripción, figuras y reivindicaciones.

BREVE DESCRIPCIÓN DE LOS DIBUJOS

[0011] La Figura 1 es un diagrama de bloques que ilustra un sistema de codificación y decodificación de video de ejemplo que puede realizar las técnicas de la presente divulgación.

[0012] La Figura 2A es un diagrama conceptual que ilustra un soporte de filtro en bucle adaptativo (ALF) en forma de diamante de 5x5 de ejemplo.

[0013] La Figura 2B es un diagrama conceptual que ilustra un soporte de ALF en forma de diamante de 7x7 de ejemplo.

[0014] La Figura 3 es un diagrama conceptual que ilustra un soporte de filtro en forma de diamante de 5x5 de ejemplo.

[0015] Las Figuras 4A-4C son diagramas conceptuales que ilustran los soportes de filtro de 5x5 de ejemplo con diferentes transformaciones geométricas.

[0016] Las Figuras 5A y 5B son diagramas conceptuales que ilustran una estructura de árbol binario de árbol cuaternario (QTBT) de ejemplo y una unidad de árbol de codificación (CTU) correspondiente.

[0017] La Figura 6 es un diagrama de bloques que ilustra un codificador de video de ejemplo que puede realizar las técnicas de la presente divulgación.

[0018] La Figura 7 es un diagrama de bloques que ilustra un decodificador de video de ejemplo que puede realizar las técnicas de la presente divulgación.

[0019] La Figura 8 es un diagrama de flujo que ilustra un método de ejemplo para codificar un bloque actual de una imagen actual de datos de video.

[0020] La Figura 9 es un diagrama de flujo que ilustra un método de ejemplo para decodificar un bloque actual de una imagen actual de datos de video.

[0021] La Figura 10 es un diagrama de flujo que ilustra una operación de ejemplo para codificar datos de video de acuerdo con una o más técnicas de la presente divulgación.

DESCRIPCIÓN DETALLADA

[0022] Los codificadores y decodificadores de video pueden aplicar un filtro en bucle adaptativo (ALF) a muestras de una imagen en una señal de video decodificada. La aplicación de un ALF puede mejorar la calidad de una señal de video decodificada. Durante la aplicación de un ALF, un enciptador de video (por ejemplo, un codificador de video o un decodificador de video) puede determinar un valor filtrado para una muestra actual. Para determinar el valor filtrado para la muestra actual, el enciptador

de video puede multiplicar una muestra recortada de un soporte de filtro de ALF para la muestra actual por un coeficiente de filtro correspondiente. Un soporte es un conjunto de muestras utilizado para derivar un valor para una muestra que se filtra. El encriptador de video puede entonces determinar el valor filtrado para la muestra actual al sumar el valor de la muestra actual a una suma de los productos de multiplicación resultantes.

[0023] Como se señaló anteriormente, el encriptador de video multiplica las muestras recortadas por los coeficientes de filtro correspondientes. El recorte se controla mediante un conjunto de valores de recorte. Los valores de recorte especifican un límite superior y un límite inferior en el valor de la muestra. El encriptador de video puede usar diferentes valores de recorte en diferentes circunstancias. En consecuencia, un codificador de video puede señalar un índice (es decir, un índice de recorte de ALF) del conjunto aplicable de valores de recorte. Por ejemplo, el codificador de video puede señalar el índice de recorte de ALF en un conjunto de parámetros de adaptación (APS).

[0024] En VVC Test Model 5.0 (VTM-5.0) (Chen *et al.*, "Algorithm Description for Versatile Video Coding and Test Model 5 (VTM 5)", Joint Video Experts Team (JVET) de ITU-T SG 16 WP 3 e ISO/IEC JTC 1/SC 29/WG 11, 14.^a reunión: Ginebra, CH, 19-27 marzo de 2019, documento JVET-N1001), el índice de recorte de ALF se indica mediante el uso de un código de Golomb exponencial. Señalar el índice de recorte de ALF como un código de Golomb exponencial puede ralentizar el proceso de decodificación porque determinar el significado de un código de Golomb exponencial puede implicar la realización de múltiples operaciones de comparación, que tienden a ser relativamente lentas.

[0025] Esta divulgación puede solucionar el problema. Como se describe en la presente, un encriptador de video (por ejemplo, un codificador de video o un decodificador de video) puede codificar un índice de recorte de ALF como un número entero sin signo de longitud fija. El encriptador de video puede aplicar, en función del índice de recorte de ALF, un ALF a un bloque de una imagen de los datos de video. Debido a que el índice de recorte de ALF se señala como un número entero sin signo de longitud fija, un decodificador de video puede realizar un proceso de decodificación más rápido.

[0026] La Figura 1 es un diagrama de bloques que ilustra un sistema de codificación y decodificación de video 100 de ejemplo que puede realizar las técnicas de la presente divulgación. Las técnicas de la presente divulgación se dirigen generalmente a encriptar (codificar y/o decodificar) datos de video. En general, los datos de video incluyen cualquier dato para procesar un video. Por lo tanto, los datos de video pueden incluir

video sin procesar, no codificado, video codificado, video decodificado (por ejemplo, reconstruido) y metadatos de video, tales como datos de señalización.

[0027] Como se muestra en la Figura 1, el sistema 100 incluye un dispositivo fuente 102 que proporciona datos de video codificados que se desean decodificar y mostrar posteriormente mediante un dispositivo de destino 116, en este ejemplo. En particular, el dispositivo fuente 102 proporciona los datos de video al dispositivo de destino 116 mediante un medio legible por computadora 110. El dispositivo fuente 102 y el dispositivo de destino 116 pueden comprender cualquiera de un amplio rango de dispositivos, incluso computadoras de escritorio, computadoras portátiles (es decir, laptop), computadoras tipo *tablet*, decodificadores, aparatos telefónicos, tales como teléfonos inteligentes, televisores, cámaras, dispositivos de visualización, reproductores de medios digitales, consolas de juegos de video, dispositivo de *streaming* de video o similares. En algunos casos, el dispositivo fuente 102 y el dispositivo de destino 116 pueden estar equipados para la comunicación inalámbrica, y, por lo tanto, pueden denominarse dispositivos de comunicación inalámbrica.

[0028] En el ejemplo de la Figura 1, el dispositivo fuente 102 incluye la fuente de video 104, la memoria 106, el codificador de video 200 y la interfaz de salida 108. El dispositivo de destino 116 incluye la interfaz de entrada 122, el decodificador de video 300, la memoria 120 y el dispositivo de visualización 118. De acuerdo con la presente divulgación, el codificador de video 200 del dispositivo fuente 102 y el decodificador de video 300 del dispositivo de destino 116 pueden configurarse para aplicar las técnicas de señalización de índices de recorte para filtros en bucle adaptativo en la codificación de video. Por lo tanto, el dispositivo fuente 102 representa un ejemplo de un dispositivo de codificación de video, mientras que el dispositivo de destino 116 representa un ejemplo de un dispositivo de decodificación de video. En otros ejemplos, un dispositivo fuente y un dispositivo de destino pueden incluir otros componentes o disposiciones. Por ejemplo, el dispositivo fuente 102 puede recibir datos de video de una fuente de video externa, tal como una cámara externa. Asimismo, el dispositivo de destino 116 puede interactuar con un dispositivo de visualización externo, en lugar de incluir un dispositivo de visualización integrado.

[0029] El sistema 100 como se muestra en la Figura 1 es solo un ejemplo. En general, cualquier dispositivo de codificación y/o decodificación de video digital puede realizar técnicas de señalización de índices de recorte para filtros en bucle adaptativo en la codificación de video. El dispositivo fuente 102 y el dispositivo de destino 116 son solo ejemplos de los dispositivos de codificación, en donde el dispositivo fuente 102 genera

datos de video codificados para la transmisión al dispositivo de destino 116. Esta divulgación se refiere a un dispositivo de "encriptación" como un dispositivo que realiza la encriptación (codificación y/o decodificación) de datos. Por lo tanto, el codificador de video 200 y el decodificador de video 300 representan ejemplos de dispositivos de codificación, en particular, un codificador de video y un decodificador de video, respectivamente. En algunos ejemplos, los dispositivos 102, 116 pueden operar de manera sustancialmente simétrica, de modo que cada uno de los dispositivos 102, 116 incluya componentes de codificación y decodificación de video. Por lo tanto, el sistema 100 puede permitir la transmisión de video unidireccional o bidireccional entre el dispositivo fuente 102 y el dispositivo de destino 116, por ejemplo, para *streaming* de video, repetición de video, transmisión de video o videotelefonía.

[0030] En general, la fuente de video 104 representa una fuente de datos de video (es decir, datos de video sin procesar, no codificados) y proporciona una serie secuencial de imágenes (también denominadas "cuadros") de los datos de video al codificador de video 200, que codifica los datos para las imágenes. La fuente de video 104 del dispositivo fuente 102 puede incluir un dispositivo de captura de video, tal como una cámara de video, un archivo de video que contiene video sin procesar capturado anteriormente y/o una interfaz de transmisión de video para recibir videos de un proveedor de contenidos de video. Como alternativa adicional, la fuente de video 104 puede generar datos basados en gráficos informáticos como el video fuente, o una combinación de video en vivo, video archivado y video generado por computadora. En cada caso, el codificador de video 200 codifica los datos de video capturado, precapturado o generado por computadora. El codificador de video 200 puede reorganizar las imágenes del orden recibido (a veces denominado "orden de visualización") en un orden de codificación para la codificación. El codificador de video 200 puede generar un flujo de bits que incluye datos de video codificados. El dispositivo fuente 102 puede emitir entonces los datos de video codificados a través de la interfaz de salida 108 en un medio legible por computadora 110 para su recepción y/o recuperación mediante, por ejemplo, la interfaz de entrada 122 del dispositivo de destino 116.

[0031] La memoria 106 del dispositivo fuente 102 y la memoria 120 del dispositivo de destino 116 representan memorias de propósito general. En algunos ejemplos, las memorias 106, 120 pueden almacenar datos de video sin procesar, por ejemplo, video sin procesar de la fuente de video 104 y datos de video sin procesar, decodificados del decodificador de video 300. Adicional o alternativamente, las memorias 106, 120 pueden

almacenar instrucciones de *software* ejecutables por, por ejemplo, el codificador de video 200 y el decodificador de video 300, respectivamente. Aunque la memoria 106 y la memoria 120 se muestran por separado del codificador de video 200 y el decodificador de video 300 en este ejemplo, debe entenderse que el codificador de video 200 y el decodificador de video 300 también pueden incluir memorias internas para propósitos funcionalmente similares o equivalentes. Además, las memorias 106, 120 pueden almacenar datos de video codificados, por ejemplo, la salida del codificador de video 200 y la entrada al decodificador de video 300. En algunos ejemplos, las porciones de las memorias 106, 120 pueden asignarse como una o más memorias intermedias de video, por ejemplo, para almacenar datos de video sin procesar, decodificados y/o codificados.

[0032] El medio legible por computadora 110 puede representar cualquier tipo de medio o dispositivo capaz de transportar los datos de video codificados desde el dispositivo fuente 102 al dispositivo de destino 116. En un ejemplo, el medio legible por computadora 110 representa un medio de comunicación para permitir que el dispositivo fuente 102 transmita datos de video codificados directamente al dispositivo de destino 116 en tiempo real, por ejemplo, a través de una red de radiofrecuencia o una red basada en computadora. La interfaz de salida 108 puede modular una señal de transmisión que incluye los datos de video codificados, y la interfaz de entrada 122 puede demodular la señal de transmisión recibida, de acuerdo con un estándar de comunicación, tal como un protocolo de comunicación inalámbrica. El medio de comunicación puede comprender cualquier medio de comunicación inalámbrica o por cable, tales como un espectro de frecuencia de radio (RF) o una o más líneas de transmisión física. El medio de comunicación puede formar parte de una red basada en paquetes, tal como una red de área local, una red de área amplia o una red global tal como Internet. El medio de comunicación puede incluir *routers*, conmutadores, estaciones base, o cualquier otro equipo que pueda ser útil para facilitar la comunicación del dispositivo fuente 102 al dispositivo de destino 116.

[0033] En algunos ejemplos, el medio legible por computadora 110 puede incluir un dispositivo de almacenamiento 112. El dispositivo fuente 102 puede enviar datos codificados desde la interfaz de salida 108 al dispositivo de almacenamiento 112. De manera similar, el dispositivo de destino 116 puede acceder a datos codificados desde el dispositivo de almacenamiento 112 a través de la interfaz de entrada 122. El dispositivo de almacenamiento 112 puede incluir cualquiera de una variedad de medios de almacenamiento de datos distribuidos o a los que se accede localmente, tales como

un disco duro, discos Blu-ray, DVD, CD-ROM, memoria flash, memoria volátil o no volátil, o cualquier otro medio de almacenamiento digital adecuado para el almacenamiento de datos de video codificados.

[0034] En algunos ejemplos, el medio legible por computadora 110 puede incluir un servidor de archivos 114 u otro dispositivo de almacenamiento intermedio que puede almacenar los datos de video codificados generados por el dispositivo fuente 102. El dispositivo fuente 102 puede enviar datos de video codificados al servidor de archivos 114 u otro dispositivo de almacenamiento intermedio que puede almacenar el video codificado generado por el dispositivo fuente 102. El dispositivo de destino 116 puede acceder a los datos de video almacenados desde el servidor de archivos 114 mediante *streaming* o descarga. El servidor de archivos 114 puede ser cualquier tipo de servidor capaz de almacenar datos de video y de transmitir los datos de video codificados al dispositivo de destino 116. El servidor de archivos 114 puede representar un servidor web (por ejemplo, para un sitio web), un servidor de protocolo de transferencia de archivos (FTP), un dispositivo de red de entrega de contenido o un dispositivo de almacenamiento conectado a la red (NAS). El dispositivo de destino 116 puede acceder a los datos de video codificados del servidor de archivos 114 a través de cualquier conexión de datos estándar, incluso una conexión a Internet. Esto puede incluir un canal inalámbrico (por ejemplo, una conexión Wi-Fi), una conexión por cable (por ejemplo, línea de abonado digital (DSL, cable módem, etc.), o una combinación de ambos que es adecuada para acceder a los datos de video codificados y almacenados en el servidor de archivos 114. El servidor de archivos 114 y la interfaz de entrada 122 pueden configurarse para operar de acuerdo con un protocolo de transmisión de *streaming*, un protocolo de transmisión de descarga o una combinación de estos.

[0035] La interfaz de salida 108 y la interfaz de entrada 122 pueden representar transmisores/receptores inalámbricos, módems, componentes de red cableados (por ejemplo, tarjetas Ethernet), componentes de comunicación inalámbrica que operan de acuerdo con cualquiera de una variedad de estándares IEEE 802.11 u otros componentes físicos. En los ejemplos en los que la interfaz de salida 108 y la interfaz de entrada 122 comprenden componentes inalámbricos, la interfaz de salida 108 y la interfaz de entrada 122 pueden configurarse para transferir datos, tales como datos de video codificados, de acuerdo con un estándar de comunicación celular, tal como 4G, 4G-LTE (evolución a largo plazo), LTE avanzada, 5G o similares. En algunos ejemplos en los que la interfaz de salida 108 comprende un transmisor inalámbrico, la interfaz de salida 108 y la interfaz de entrada 122 pueden configurarse para transferir datos, tales

como datos de video codificados, de acuerdo con otros estándares inalámbricos, tales como una especificación IEEE 802.11, una especificación IEEE 802.15 (por ejemplo, ZigBee™), un estándar Bluetooth™ o similares. En algunos ejemplos, el dispositivo fuente 102 y/o el dispositivo de destino 116 pueden incluir respectivos dispositivos de sistema en un chip (SoC). Por ejemplo, el dispositivo fuente 102 puede incluir un dispositivo de SoC para realizar la funcionalidad atribuida al codificador de video 200 y/o la interfaz de salida 108, y el dispositivo de destino 116 puede incluir un dispositivo de SoC para realizar la funcionalidad atribuida al decodificador de video 300 y/o la interfaz de entrada 122.

[0036] Las técnicas de la presente divulgación se pueden aplicar a la codificación de video para diversas aplicaciones multimedia, tales como transmisiones por televisión inalámbrica, transmisiones por televisión por cable, transmisiones por televisión satelital, transmisiones de video por *streaming* en Internet, tales como *streaming* adaptativo dinámico en HTTP (DASH), video digital que se codifica en un medio de almacenamiento de datos, decodificación de video digital almacenado en un medio de almacenamiento de datos, u otras aplicaciones.

[0037] La interfaz de entrada 122 del dispositivo de destino 116 recibe un flujo de bits de video codificado desde un medio legible por computadora 110 (por ejemplo, un medio de comunicación, el dispositivo de almacenamiento 112, el servidor de archivos 114 o similares). El flujo de bits de video codificado puede incluir información de señalización definida por el codificador de video 200, que también es utilizada por el decodificador de video 300, tal como elementos de sintaxis que tienen valores que describen características y/o procesamiento de bloques de video u otras unidades codificadas (por ejemplo, porciones, imágenes, grupos de imágenes, secuencias o similares). El dispositivo de visualización 118 muestra imágenes decodificadas de los datos de video decodificados a un usuario. El dispositivo de visualización 118 puede representar cualquiera de los diversos dispositivos de visualización, tales como un tubo de rayos catódicos (CRT), una pantalla de cristal líquido (LCD), una pantalla de plasma, una pantalla de diodo orgánico de emisión de luz (OLED), u otro tipo de dispositivo de visualización.

[0038] Aunque no se muestra en la Figura 1, en algunos ejemplos, el codificador de video 200 y el decodificador de video 300 pueden cada uno estar integrados con un codificador de audio y/o un decodificador de audio, y pueden incluir unidades MUX-DEMUX apropiadas, u otro *hardware* y/o *software*, para manejar flujos múltiples, incluso la codificación de audio y video en un flujo común de datos. Si correspondiera, las

unidades MUX-DEMUX pueden adecuarse al protocolo de multiplexor ITU H.223, o a otros protocolos, tales como el protocolo de datagrama de usuario (UDP).

[0039] El codificador de video 200 y el decodificador de video 300 pueden cada uno implementarse como cualquiera de una variedad de circuito codificador y/o decodificador adecuado, tal como uno o más microprocesadores, procesadores de señal digital (DSP), circuitos integrados de aplicación específica (ASIC), matrices de puertas programables en campo (FPGA), circuitos discretos lógicos, *software*, *hardware*, *firmware* o cualquier combinación de estos. Cuando las técnicas se implementan parcialmente en *software*, un dispositivo puede almacenar las instrucciones para el *software* en un medio legible por computadora no transitorio adecuado y ejecutar las instrucciones en *hardware* mediante el uso de uno o más procesadores para realizar las técnicas de la presente divulgación. Cada uno del codificador de video 200 y el decodificador de video 300 pueden incluirse en uno o más codificadores o decodificadores, cualquiera de los cuales pueden estar integrados como parte de un codificador/decodificador combinado (CODEC) en un dispositivo respectivo. Un dispositivo que incluye un codificador de video 200 y/o un decodificador de video 300 puede comprender un circuito integrado, un microprocesador y/o un dispositivo de comunicación inalámbrica, tal como un teléfono celular.

[0040] El codificador de video 200 y el decodificador de video 300 pueden operar de acuerdo con un estándar de codificación de video, tal como ITU-T H.265, también conocido como codificación de video de alta eficiencia (HEVC) o extensiones de este, tales como las extensiones de codificación de video multivisión y/o escalable. Alternativamente, el codificador de video 200 y el decodificador de video 300 pueden operar de acuerdo con otros estándares patentados o de la industria, tales como ITU-T H.266, también denominado codificación de video versátil (VVC). Un borrador reciente del estándar VVC se describe en Bross, *et al.* "Versatile Video Coding (Draft 5)", Joint Video Experts Team (JVET) de ITU-T SG 16 WP 3 e ISO/IEC JTC 1/SC 29/WG 11, 14.^a reunión: Ginebra, CH, 19-27 de marzo de 2019, JVET-N1001-v3 (en adelante en la presente "Borrador 5 de VVC"). Las técnicas de esta divulgación, sin embargo, no se limitan a ningún estándar de codificación particular.

[0041] En general, el codificador de video 200 y el decodificador de video 300 pueden realizar una codificación de imágenes basada en bloques. El término "bloque" generalmente se refiere a una estructura que incluye datos para ser procesados (por ejemplo, codificados, decodificados o usados de otro modo en el proceso de codificación y/o decodificación). Por ejemplo, un bloque puede incluir una matriz bidimensional de

muestras de datos de luminancia y/o crominancia. En general, el codificador de video 200 y el decodificador de video 300 pueden codificar datos de video representados en un formato YUV (por ejemplo, Y, Cb, Cr). Es decir, en lugar de codificar datos de rojo, verde y azul (RGB) para muestras de una imagen, el codificador de video 200 y el decodificador de video 300 pueden codificar componentes de luminancia y crominancia, en donde los componentes de crominancia pueden incluir componentes de crominancia de tono rojo y azul. En algunos ejemplos, el codificador de video 200 convierte los datos con formato RGB recibidos en una representación YUV antes de la codificación, y el decodificador de video 300 convierte la representación YUV en el formato RGB. Alternativamente, las unidades de procesamiento previo y posterior (no mostradas) pueden realizar estas conversiones.

[0042] Esta divulgación puede referirse generalmente a la codificación (por ejemplo, codificación y decodificación) de imágenes para incluir el proceso de codificación o decodificación de datos de la imagen. De manera similar, esta divulgación puede referirse a la codificación de bloques de una imagen para incluir el proceso de codificación o decodificación de datos para los bloques, por ejemplo, predicción y/o codificación residual. Un flujo de bits de video codificado generalmente incluye una serie de valores para elementos de sintaxis representativos de decisiones de codificación (por ejemplo, modos de codificación) y partición de imágenes en bloques. Por lo tanto, las referencias a la codificación de una imagen o un bloque deben entenderse generalmente como valores de codificación para los elementos de sintaxis que forman la imagen o el bloque.

[0043] HEVC define diversos bloques, incluso las unidades de codificación (CU), las unidades de predicción (PU) y las unidades de transformada (TU). De acuerdo con HEVC, un encriptador de video (tal como el codificador de video 200) divide una unidad de árbol de codificación (CTU) en CU de acuerdo con una estructura de árbol cuaternario. Es decir, el encriptador de video divide las CTU y CU en cuatro cuadrados iguales que no se superponen, y cada nodo del árbol cuaternario tiene cero o cuatro nodos secundarios. Los nodos sin nodos secundarios pueden denominarse "nodos hoja", y las CU de dichos nodos hoja pueden incluir una o más PU y/o una o más TU. El encriptador de video puede dividir además las PU y las TU. Por ejemplo, en HEVC, un árbol cuaternario residual (RQT) representa la partición de TU. En HEVC, las PU representan datos de interpredicción, mientras que las TU representan datos residuales. Las CU que son intrapredichas incluyen información de intrapredicción, tal como una indicación intramodo.

[0044] Como otro ejemplo, el codificador de video 200 y el decodificador de video 300 pueden configurarse para operar de acuerdo con VVC. De acuerdo con VVC, un encriptador de video (tal como el codificador de video 200) divide una imagen en una pluralidad de CTU. Un bloque de árbol de codificación (CTB) es $N \times N$ bloques de muestras para algún valor de N tal que la división de un componente en CTB es una partición. En VVC, una CTU puede definirse como un CTB de muestras de luminancia, dos CTB de muestras de crominancia correspondientes de una imagen que tiene tres matrices de muestra, o un CTB de muestras de una imagen monocroma o una imagen que se codifica mediante el uso de tres planos de color separados y estructuras de sintaxis utilizadas para codificar las muestras.

[0045] El codificador de video 200 puede dividir una CTU de acuerdo con una estructura de árbol, tal como una estructura de árbol binario de árbol cuaternario (QTBT) o una estructura de árbol de tipos múltiples (MTT). La estructura QTBT elimina los conceptos de múltiples tipos de partición, tales como la separación entre CU, PU y TU de HEVC. Una estructura QTBT incluye dos niveles: un primer nivel dividido de acuerdo con la partición de árbol cuaternario y un segundo nivel dividido de acuerdo con la partición de árbol binario. Un nodo raíz de la estructura QTBT corresponde a una CTU. Los nodos hoja de los árboles binarios corresponden a unidades de codificación (CU).

[0046] En una estructura de partición MTT, los bloques se pueden particionar mediante el uso de una partición de árbol cuaternario (QT), una partición de árbol binario (BT) y uno o más tipos de particiones de árbol triple (TT). Una partición de árbol triple es una partición en la que un bloque se divide en tres subbloques. En algunos ejemplos, una partición de árbol triple divide un bloque en tres subbloques sin dividir el bloque original por el centro. Los tipos de partición en MTT (por ejemplo, QT, BT y TT) pueden ser simétricos o asimétricos.

[0047] En algunos ejemplos, el codificador de video 200 y el decodificador de video 300 pueden usar una sola estructura QTBT o MTT para representar cada uno de los componentes de luminancia y crominancia, mientras que en otros ejemplos, el codificador de video 200 y el decodificador de video 300 pueden usar dos o más estructuras QTBT o MTT, tal como una estructura QTBT/MTT para el componente de luminancia y otra estructura QTBT/MTT para ambos componentes de crominancia (o dos estructuras QTBT/MTT para los respectivos componentes de crominancia).

[0048] El codificador de video 200 y el decodificador de video 300 pueden configurarse para usar partición de árbol cuaternario de acuerdo con HEVC, partición QTBT, partición MTT u otras estructuras de partición. Para fines de explicación, la descripción de las

técnicas de esta divulgación se presenta con respecto a la partición QTBT. Sin embargo, debe entenderse que las técnicas de esta divulgación también se pueden aplicar a encriptadores de video configurados para usar partición de árbol cuaternario, así como también otros tipos de partición.

[0049] Esta divulgación puede usar "NxN" y "N por N" indistintamente para referirse a las dimensiones de muestra de un bloque (tal como una CU u otro bloque de video) en términos de dimensiones verticales y horizontales, por ejemplo, muestras de 16x16 o muestras de 16 por 16. En general, una CU de 16x16 tendrá 16 muestras en dirección vertical ($y = 16$) y 16 muestras en dirección horizontal ($x = 16$). Asimismo, una CU NxN generalmente tiene N muestras en una dirección vertical y N muestras en una dirección horizontal, donde N representa un valor entero no negativo. Las muestras en una CU pueden disponerse en filas y columnas. Además, las CU no necesitan tener necesariamente la misma cantidad de muestras en la dirección horizontal que en la dirección vertical. Por ejemplo, las CU pueden comprender muestras NxM, en donde M no es necesariamente igual a N.

[0050] El codificador de video 200 codifica datos de video para CU que representan información de predicción y/o residual y otra información. La información de predicción indica cómo se va a predecir la CU para formar un bloque de predicción para la CU. La información residual generalmente representa diferencias muestra por muestra entre muestras de la CU antes de la codificación y el bloque de predicción.

[0051] Para predecir una CU, el codificador de video 200 puede formar generalmente un bloque de predicción para la CU mediante interpredicción o intrapredicción. La interpredicción generalmente se refiere a predecir la CU a partir de datos de una imagen previamente codificada, mientras que la intrapredicción generalmente se refiere a predecir la CU a partir de datos previamente codificados de la misma imagen. Para realizar la interpredicción, el codificador de video 200 puede generar el bloque de predicción mediante el uso de uno o más vectores de movimiento. El codificador de video 200 generalmente puede realizar una búsqueda de movimiento para identificar un bloque de referencia que se asemeje mucho a la CU, por ejemplo, en términos de diferencias entre la CU y el bloque de referencia. El codificador de video 200 puede calcular una métrica de diferencia mediante el uso de una suma de diferencia absoluta (SAD), suma de diferencias cuadradas (SSD), diferencia absoluta media (MAD), diferencias cuadráticas medias (MSD) u otros cálculos de diferencias similares para determinar si una referencia de bloque se asemeja mucho a la CU actual. En algunos ejemplos, el codificador de video 200 puede predecir la CU actual mediante el uso de

predicción unidireccional o predicción bidireccional.

[0052] Algunos ejemplos de VVC también proporcionan un modo de compensación de movimiento afín, que puede considerarse un modo de interpredicción. En el modo de compensación de movimiento afín, el codificador de video 200 puede determinar dos o más vectores de movimiento que representan un movimiento no traslacional, tal como acercar o alejar, rotación, movimiento en perspectiva u otros tipos de movimiento irregular.

[0053] Para realizar la intrapredicción, el codificador de video 200 puede seleccionar un modo de intrapredicción para generar el bloque de predicción. Algunos ejemplos de VVC proporcionan sesenta y siete modos de intrapredicción, incluso diversos modos direccionales, así como el modo planar y el modo DC. En general, el codificador de video 200 selecciona un modo de intrapredicción que describe muestras vecinas a un bloque actual (por ejemplo, un bloque de una CU) desde el cual predecir muestras del bloque actual. Por lo general, dichas muestras pueden estar arriba, arriba y a la izquierda, o a la izquierda del bloque actual en la misma imagen que el bloque actual, asumiendo que el codificador de video 200 codifica CTU y CU en orden de exploración de trama (de izquierda a derecha, de arriba a abajo).

[0054] El codificador de video 200 codifica datos que representan el modo de predicción para un bloque actual. Por ejemplo, para los modos de interpredicción, el codificador de video 200 puede codificar datos que representen cuál de los diversos modos de interpredicción disponibles se usa, así como información de movimiento para el modo correspondiente. Para la interpredicción unidireccional o bidireccional, por ejemplo, el codificador de video 200 puede codificar vectores de movimiento mediante el uso de la predicción avanzada de vectores de movimiento (AMVP) o el modo de fusión. El codificador de video 200 puede usar modos similares para codificar vectores de movimiento para el modo de compensación de movimiento afín.

[0055] Después de la predicción, tal como intrapredicción o interpredicción de un bloque, el codificador de video 200 puede calcular datos residuales para el bloque. Los datos residuales, tales como un bloque residual, representan las diferencias muestra por muestra entre el bloque y un bloque de predicción para el bloque, formado mediante el uso del modo de predicción correspondiente. El codificador de video 200 puede aplicar una o más transformadas al bloque residual, para producir datos transformados en un dominio de transformada en lugar del dominio de muestra. Por ejemplo, el codificador de video 200 puede aplicar una transformada de coseno discreta (DCT), una transformada de número entero, una transformada de ondículas o una transformada

conceptualmente similar a los datos de video residuales. Además, el codificador de video 200 puede aplicar una transformada secundaria después de la primera transformada, tal como una transformada secundaria no separable dependiente del modo (MDNSST), una transformada dependiente de la señal, una transformada de Karhunen-Loeve (KLT) o similares. El codificador de video 200 produce coeficientes de transformada tras la aplicación de la una o más transformadas.

[0056] Como se indicó anteriormente, tras cualquier transformada para producir coeficientes de transformada, el codificador de video 200 puede realizar la cuantificación de los coeficientes de transformada. Cuantificación se refiere generalmente a un proceso donde los coeficientes de transformada se cuantifican para reducir posiblemente la cantidad de datos utilizados para representar los coeficientes de transformada, lo que proporciona una compresión adicional. Al realizar el proceso de cuantificación, el codificador de video 200 puede reducir la profundidad de bits asociada con algunos o todos los coeficientes de transformada. Por ejemplo, el codificador de video 200 puede redondear un valor de n bits a un valor de m bits durante la cuantificación, donde n es mayor que m . En algunos ejemplos, para realizar la cuantificación, el codificador de video 200 puede realizar un desplazamiento a la derecha bit a bit del valor que se va a cuantificar.

[0057] Después de la cuantificación, el codificador de video 200 puede explorar los coeficientes de transformada, lo que produce un vector unidimensional a partir de la matriz bidimensional que incluye los coeficientes de transformada cuantificados. La exploración puede diseñarse para colocar coeficientes de transformada de energía más alta (y por lo tanto de frecuencia más baja) en el frente del vector y para colocar coeficientes de transformada de energía más baja (y por lo tanto de frecuencia más alta) en la parte posterior del vector. En algunos ejemplos, el codificador de video 200 puede utilizar un orden de exploración predefinido para explorar los coeficientes de transformada cuantificados para producir un vector serializado, y luego codificar por entropía los coeficientes de transformada cuantificados del vector. En otros ejemplos, el codificador de video 200 puede realizar una exploración adaptativa. Después de explorar los coeficientes de transformada cuantificados para formar el vector unidimensional, el codificador de video 200 puede codificar por entropía el vector unidimensional, por ejemplo, de acuerdo con la codificación aritmética binaria adaptativa al contexto (CABAC). El codificador de video 200 también puede codificar por entropía valores para elementos de sintaxis que describen metadatos asociados con los datos de video codificados para su uso por el decodificador de video 300 en la decodificación

de los datos de video.

[0058] Para realizar CABAC, el codificador de video 200 puede asignar un contexto dentro de un modelo de contexto a un símbolo a transmitir. El contexto puede referirse, por ejemplo, a si los valores vecinos del símbolo tienen valor cero o no. La determinación de la probabilidad puede ser en función de un contexto asignado al símbolo.

[0059] El codificador de video 200 puede generar además datos de sintaxis, tales como datos de sintaxis basados en bloques, datos de sintaxis basados en imágenes y datos de sintaxis basados en secuencias, al decodificador de video 300, por ejemplo, en un encabezado de imagen, un encabezado de bloque, un encabezado de porción, u otros datos de sintaxis, tales como un conjunto de parámetros de secuencia (SPS), un conjunto de parámetros de imagen (PPS) o un conjunto de parámetros de video (VPS). El decodificador de video 300 puede decodificar igualmente tales datos de sintaxis para determinar cómo decodificar los datos de video correspondientes.

[0060] De esta manera, el codificador de video 200 puede generar un flujo de bits que incluye datos de video codificados, por ejemplo, elementos de sintaxis que describen la partición de una imagen en bloques (por ejemplo, CU) e información de predicción y/o residual para los bloques. Finalmente, el decodificador de video 300 puede recibir el flujo de bits y decodificar los datos de video codificados.

[0061] En general, el decodificador de video 300 realiza un proceso recíproco al realizado por el codificador de video 200 para decodificar los datos de video codificados del flujo de bits. Por ejemplo, el decodificador de video 300 puede decodificar valores para elementos de sintaxis del flujo de bits mediante el uso de CABAC de una manera sustancialmente similar, aunque recíproca al proceso de codificación CABAC del codificador de video 200. Los elementos de sintaxis pueden definir la información de partición para la partición de una imagen en CTU, y la partición de cada CTU de acuerdo con una estructura de partición correspondiente, tal como una estructura QTBT, para definir las CU de la CTU. Los elementos de sintaxis pueden definir además la información de predicción y residual para los bloques (por ejemplo, CU) de datos de video.

[0062] La información residual puede representarse, por ejemplo, mediante coeficientes de transformada cuantificados. El decodificador de video 300 puede cuantificar de forma inversa y transformar de forma inversa los coeficientes de transformada cuantificados de un bloque para reproducir un bloque residual para el bloque. El decodificador de video 300 utiliza un modo de predicción señalado (intrapredicción o interpredicción) e información de predicción relacionada (por ejemplo, información de movimiento para la

interpredicción) para formar un bloque de predicción para el bloque. El decodificador de video 300 puede combinar entonces el bloque de predicción y el bloque residual (muestra por muestra) para reproducir el bloque original. El decodificador de video 300 puede realizar un procesamiento adicional, tal como realizar un proceso de desbloqueo para reducir los artefactos visuales a lo largo de los límites del bloque.

[0063] En el campo de la codificación de video, es común aplicar filtros para mejorar la calidad de una señal de video decodificada. El filtro se puede aplicar como posfiltro, donde el cuadro filtrado no se usa para la predicción de cuadros futuros, o como un filtro en bucle, donde el cuadro filtrado se usa para predecir un cuadro futuro. Se puede diseñar un filtro, por ejemplo, al minimizar el error entre la señal original y la señal filtrada decodificada.

[0064] En VVC Test Model 5.0 (VTM-5.0) (Chen et al., "Algorithm Description for Versatile Video Coding and Test Model 5 (VTM 5)", Joint Video Experts Team (JVET) de ITU-T SG 16 WP 3 e ISO/IEC JTC 1/SC 29/WG 11, 14.^a reunión: Ginebra, CH, 19-27 de marzo de 2019, documento JVET-N1001), los coeficientes de filtro decodificados $f(k, l)$ y los valores de recorte $c(k, l)$ se aplican a la imagen reconstruida $R(i, j)$ de la siguiente manera:

$$\tilde{R}(i, j) = R(i, j) + \sum_{k, l = (-K, -K), k, l \neq (0, 0)}^{K, K} f(k, l) * \text{clip3}(-c(k, l), c(k, l), R(i + k, j + l) - R(i, j)) \quad (1)$$

En VTM-5.0, se aplica un filtro de 7x7 a los componentes de luminancia y un filtro de 5x5 a los componentes de crominancia. La Figura 2A es un diagrama conceptual que ilustra un soporte de ALF en forma de diamante de 5x5 de ejemplo. La Figura 2B es un diagrama conceptual que ilustra un soporte de ALF en forma de diamante de 7x7 de ejemplo. En la ecuación (1), K puede ser igual a $\frac{L}{2}$, donde L indica la longitud del filtro. Además, en la ecuación (1) y en otras partes de esta divulgación, la función clip3 se puede definir como:

$$\text{Clip3}(x, y, z) = \begin{cases} x & ; & z < x \\ y & ; & z > y \\ z & ; & \text{de otro modo} \end{cases}$$

[0065] En la ecuación (1) y en otras partes de esta divulgación, un valor de recorte $c(k, l)$ se puede calcular de la siguiente manera. Para el componente de luminancia, un valor de recorte $c(k, l)$ se puede calcular como:

$$c(k, l) = \text{Round}(2^{(\text{BitDepthY} * (4 - \text{clipIdx}(k, l)) / 4)}) \quad (1')$$

En la ecuación (1'), BitDepthY es la profundidad de bits para el componente de luminancia y clipIdx(k,l) es un índice de recorte para la posición (k,l). clipIdx(k,l) puede ser 0, 1, 2 o 3.

[0066] Para el componente de crominancia, un valor de recorte $c(k, l)$ se puede calcular como:

$$c(k, l) = \text{Round}(2^{(\text{BitDepthC} - 8)} * 2^{(8 * (3 - \text{clipIdx}[k, l]) / 3)}) \quad (1'')$$

En la ecuación (1''), BitDepthC es la profundidad de bits para el componente de crominancia y clipIdx(k,l) es un valor de recorte para la posición (k,l). clipIdx(k,l) puede ser 0, 1, 2 o 3.

[0067] Para el componente de luminancia, los bloques de 4×4 en la imagen completa se clasifican en función de una dirección laplaciana unidimensional (1D) (hasta 5 direcciones) y una actividad laplaciana 2D (hasta 5 valores de actividad). El cálculo de la dirección Dir_b y la actividad no cuantificada Act_b . Act_b se cuantifica además en el rango de 0 a 4 inclusive.

[0068] En primer lugar, un encriptador de video (por ejemplo, el codificador de video 200 o el decodificador de video 300) calcula los valores de dos gradientes diagonales, además de los gradientes horizontales y verticales usados en el ALF existente, mediante el uso de una laplaciana 1D. Como se puede ver en las ecuaciones (2) a (5), a continuación, la suma de los gradientes de todos los píxeles dentro de una ventana de 8x8 que cubre un píxel diana se emplea como el gradiente representado del píxel diana, donde $R(k, l)$ indica los píxeles reconstruidos en la ubicación (k, l) y los índices i y j se refieren a las coordenadas del píxel superior izquierdo en el bloque de 4×4. Cada píxel está asociado con cuatro valores de gradiente, con un gradiente vertical indicado mediante g_v , un gradiente horizontal indicado mediante g_h , un gradiente diagonal de 135 grados indicado mediante g_{d1} y un gradiente diagonal de 45 grados indicado mediante g_{d2} .

$$g_v = \sum_{k=i-2}^{i+5} \sum_{l=j-2}^{j+5} V_{k,l}, \quad (2)$$

$V_{k,l} = |2R(k,l) - R(k,l-1) - R(k,l+1)|$, cuando tanto k como l son números pares o tanto k como l no son números pares. De otro modo, 0.

$$g_h = \sum_{k=i-2}^{i+5} \sum_{l=j-2}^{j+5} H_{k,l},$$

$H_{k,l} = |2R(k,l) - R(k-1,l) - R(k+1,l)|$, cuando tanto k como l son números pares o tanto k como l no son números pares. De otro modo, 0.

$$g_{d1} = \sum_{k=i-2}^{i+5} \sum_{l=j-3}^{j+5} D1_{k,l},$$

$D1_{k,l} = |2R(k,l) - R(k-1,l-1) - R(k+1,l+1)|$, cuando tanto k como l son números pares o tanto k como l no son números pares. De otro modo, 0.

$$g_{d2} = \sum_{k=i-2}^{i+5} \sum_{j=j-2}^{j+5} D2_{k,l},$$

$D2_{k,l} = |2R(k,l) - R(k-1,l+1) - R(k+1,l-1)|$, cuando tanto k como l son números pares o tanto k como l no son números pares. De otro modo, 0.

[0069] Para asignar la direccionalidad Dir_b , una relación de un máximo y un mínimo de los gradientes horizontal y vertical (indicado mediante $R_{h,v}$ en la ecuación (6), a continuación) y la relación de un máximo y un mínimo de dos gradientes diagonales (indicado mediante $R_{d1,d2}$ en la ecuación (7), a continuación) se comparan entre sí con dos umbrales t_1 y t_2 .

$$R_{h,v} = g_{h,v}^{max} / g_{h,v}^{min}$$

en donde $g_{h,v}^{max} = \max(g_h, g_v)$, $g_{h,v}^{min} = \min(g_h, g_v)$,

$$R_{d0,d1} = g_{d0,d1}^{max} / g_{d0,d1}^{min}$$

en donde $g_{d0,d1}^{max} = \max(g_{d0}, g_{d1})$, $g_{d0,d1}^{min} = \min(g_{d0}, g_{d1})$

En las ecuaciones (6) y (7), y en otras partes de esta divulgación, $g_{h,v}^{max}$ indica el máximo de los gradientes horizontal y vertical; $g_{h,v}^{min}$ indica el mínimo de los gradientes horizontal

y vertical; $g_{d0,d1}^{max}$ indica el máximo de los dos gradientes diagonales; y $g_{h,v}^{min}$ indica el mínimo de los dos gradientes diagonales.

[0070] Al comparar las relaciones detectadas de los gradientes horizontal/vertical y diagonal, se definen cinco modos de dirección, es decir, Dir_b dentro del rango de [0, 4] inclusive, en la ecuación (8), a continuación. Los valores de Dir_b y sus significados físicos se describen en la Tabla 1.

$$Dir_b = \begin{cases} 0 & R_{h,v} \leq t_1 \ \&\& R_{d0,d1} \leq t_1 \\ 1 & R_{h,v} > t_1 \ \&\& R_{h,v} > R_{d0,d1} \ \&\& R_{h,v} > t_2 \\ 2 & R_{h,v} > t_1 \ \&\& R_{h,v} > R_{d0,d1} \ \&\& R_{h,v} \leq t_2 \\ 3 & R_{d0,d1} > t_1 \ \&\& R_{h,v} \leq R_{d0,d1} \ \&\& R_{d0,d1} > t_2 \\ 4 & R_{d0,d1} > t_1 \ \&\& R_{h,v} \leq R_{d0,d1} \ \&\& R_{d0,d1} \leq t_2 \end{cases} \quad (8)$$

Tabla 1. Valores de dirección y su significado físico

Valores de dirección	Significado físico
0	Textura
1	Horizontal/vertical fuerte
2	Horizontal/vertical
3	Diagonal fuerte
4	Diagonal

[0071] El encriptador de video (por ejemplo, el codificador de video 200 o el decodificador de video 300) puede calcular un valor de actividad Act como:

$$Act = \sum_{k=i-2}^{i+5} \sum_{l=j-2}^{j+5} (V_{k,l} + H_{k,l}). \quad (9)$$

[0072] El encriptador de video puede cuantificar además Act en un rango de 0 a 4, inclusive. El valor cuantificado de Act se indica como $\hat{A}$.

[0073] El proceso de cuantificación de Act puede definirse de la siguiente manera:

avg_var = Clip_post(NUM_ENTRY-1, (Act * ScaleFactor) >> shift);

$\hat{A} = \text{ActivityToIndex}[\text{avg_var}]$

en donde NUM_ENTRY se establece en 16, ScaleFactor se establece en 64, shift es (4

+ profundidad de bits codificada interna) y $\text{ActivityToIndex}[\text{NUM_ENTRY}] = \{0, 1, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 4\}$. La función $\text{Clip_post}(a, b)$ devuelve el valor más pequeño entre a y b .

[0074] En total, el encriptador de video puede clasificar cada bloque de luminancia de 4×4 en una de las 25 clases (5 direcciones x 5 niveles de actividad) y se asigna un índice a cada bloque de 4×4 de acuerdo con el valor de Dir_b y Act_b del bloque. El índice de grupo puede indicarse con C y establecerse igual a $5\text{Dir}_b + \hat{A}$, en donde $\hat{A}$ es el valor cuantificado de Act_b .

[0075] En algunos ejemplos, el encriptador de video puede aplicar transformaciones geométricas a coeficientes de filtro. Por ejemplo, en algunos de estos ejemplos, el encriptador de video puede, para cada categoría, señalar un conjunto de coeficientes de filtro y valores de recorte. Para distinguir mejor las diferentes direcciones de los bloques marcados con el mismo índice de categoría, se introducen cuatro transformaciones geométricas, que incluyen sin transformación, diagonal, giro vertical y rotación. Un ejemplo de soporte de filtro de 5×5 con las tres transformaciones geométricas se representa en las Figuras 4A-4C. En otras palabras, las Figuras 4A-4C son diagramas conceptuales que ilustran los soportes de filtro de 5×5 de ejemplo con diferentes transformaciones geométricas. Al comparar la Figura 3 y las Figuras 4A-4C, las formas de fórmula de las tres transformaciones geométricas adicionales se pueden derivar como:

$$\begin{aligned} \text{Diagonal: } f_D(k, l) &= f(l, k), \quad c_D(k, l) = c(l, k), \\ \text{Giro vertical: } f_V(k, l) &= f(k, K - l - 1), \quad c_V(k, l) = c(k, K - l - 1) \\ \text{Rotación: } f_R(k, l) &= f(K - l - 1, k), \quad c_R(k, l) = c(K - l - 1, k). \end{aligned} \quad (10)$$

En la ecuación (10), K es el tamaño del filtro y $0 \leq k, l \leq K - 1$ son las coordenadas de coeficientes, de modo que la ubicación $(0,0)$ está en la esquina superior izquierda y la ubicación $(K - 1, K - 1)$ está en la esquina inferior derecha. Debe observarse que cuando se usa el soporte de filtro de diamante, tal como en el ALF del Borrador 5 de VVC, los coeficientes de filtro con coordenadas fuera del soporte de filtro siempre se establecen en 0. Una forma de indicar el índice de transformación de geometría es derivar el índice de transformación de geometría implícitamente para evitar sobrecarga adicional. En ALF geométrico (GALF), las transformaciones se aplican (por ejemplo, mediante un encriptador de video tal como el codificador de video 200 o el decodificador de video 300) a los coeficientes de filtro $f(k, l)$ según los valores de gradiente calculados

para ese bloque. La relación entre la transformación y los cuatro gradientes calculados mediante el uso de las ecuaciones (2)-(5) se describe en la Tabla 2, a continuación. En resumen, las transformaciones se basan en cuál de los dos gradientes (horizontal y vertical, o gradientes de 45 y 135 grados) es mayor. En función de la comparación, el encriptador de video puede extraer información de dirección más precisa. Por lo tanto, podrían obtenerse diferentes resultados de filtrado debido a la transformación mientras no aumenta la sobrecarga de los coeficientes de filtro.

Tabla 2. Mapeo de gradientes y transformaciones.

Valores de gradiente	Transformación
$g_{d2} < g_{d1} \text{ y } g_h < g_v$	Sin transformación
$g_{d2} < g_{d1} \text{ y } g_v < g_h$	Diagonal
$g_{d1} < g_{d2} \text{ y } g_h < g_v$	Giro vertical
$g_{d1} < g_{d2} \text{ y } g_v < g_h$	Rotación

[0076] La información de filtro se puede señalar en un flujo de bits. Un conjunto de filtros de luminancia contiene información de filtro (incluso coeficientes de filtro y valores de recorte) para las 25 clases. Se pueden usar filtros fijos para predecir los filtros para cada clase. Se podría señalar una bandera para cada clase para indicar si esta clase usa un filtro fijo como su predictor de filtro. En caso afirmativo (es decir, si la bandera de una clase indica que la clase usa un filtro fijo como su predictor de filtro), se señala la información de filtro fijo.

[0077] Para reducir la cantidad de bits necesarios para representar los coeficientes de filtro, se pueden fusionar diferentes clases. La información sobre qué clases se fusionan puede proporcionarse al enviar, para cada una de las 25 clases, un índice i_c . Las clases que tienen el mismo índice i_c comparten los mismos coeficientes de filtro que están codificados. El mapeo entre clases y filtros se señala para cada conjunto de filtros de luminancia. El índice i_c está codificado con el método de binarización binaria truncada. Un filtro señalado se puede predecir a partir de un filtro señalado previamente.

[0078] Un flujo de bits puede comprender una secuencia de unidades de capa de abstracción de red (NAL). Una unidad de NAL es una estructura de sintaxis que contiene una indicación del tipo de datos en la unidad de NAL y bytes que contienen esos datos en forma de una carga útil de secuencia de bytes en bruto (RBSP) intercalada según sea necesario con bits de prevención de emulación. Cada una de las unidades de NAL puede incluir un encabezado de unidad de NAL y puede encapsular una RBSP. El encabezado de unidad de NAL puede incluir un elemento de sintaxis que indica un código de tipo de unidad de NAL. El código de tipo de unidad de NAL especificado por el encabezado de unidad de NAL de una unidad de NAL indica el tipo de unidad de NAL.

Una RBSP puede ser una estructura de sintaxis que contiene un número entero de bytes que se encapsula dentro de una unidad de NAL. En algunos casos, una RBSP incluye cero bits.

[0079] Como se indicó anteriormente, un flujo de bits puede incluir una representación de imágenes codificadas de los datos de video y datos asociados. Los datos asociados pueden incluir conjuntos de parámetros. Las unidades de NAL pueden encapsular RBSP para conjuntos de parámetros de video (VPS), conjuntos de parámetros de secuencia (SPS), conjuntos de parámetros de imagen (PPS) y conjuntos de parámetros de adaptación (APS). Un VPS es una estructura de sintaxis que incluye elementos de sintaxis que se aplican a cero o más secuencias de video codificadas (CVS) completas. Un SPS también es una estructura de sintaxis que incluye elementos de sintaxis que se aplican a cero o más CVS completas. Un SPS puede incluir un elemento de sintaxis que identifica un VPS que está activo cuando el SPS está activo. Por tanto, los elementos de sintaxis de un SPV pueden ser de aplicación más general que los elementos de sintaxis de un SPV. Un PPS es una estructura de sintaxis que incluye elementos de sintaxis que se aplican a cero o más imágenes codificadas. Un PPS puede incluir un elemento de sintaxis que identifica un SPS que está activo cuando el PPS está activo. Un encabezado de porción de una porción puede incluir un elemento de sintaxis que indica un PPS que está activo cuando se codifica la porción. Un APS es una estructura de sintaxis que contiene elementos de sintaxis que se aplican a cero o más porciones según se determina mediante cero o más elementos de sintaxis encontrados en los encabezados de porción. Un encabezado de porción de una porción puede incluir uno o más elementos de sintaxis que indican APS que están activos cuando se codifica la porción.

[0080] En VTM-5.0, los APS se usan para transportar coeficientes de filtro de ALF en flujo de bits. Un APS puede contener un conjunto de filtros de luminancia o un conjunto de filtros de crominancia o ambos. Un grupo de mosaicos, porción o imagen solo señala índices de APS que se utilizaron para el grupo de mosaicos actual en su grupo de mosaicos, porción o encabezado de imagen.

[0081] En VTM-5.0, los filtros generados a partir de grupos de mosaicos, porciones o imágenes previamente codificados se pueden usar para un grupo de mosaicos, porción o imagen actual para guardar la sobrecarga para la señalización del filtro. El codificador de video 200 puede elegir, para un CTB de luminancia, un conjunto de filtros entre conjuntos de filtros fijos y conjuntos de filtros de APS. El codificador de video 200 puede señalar el índice del conjunto de filtros elegido. Todos los CTB de crominancia usan

un filtro del mismo APS. En un grupo de mosaicos, porción o encabezado de imagen, el codificador de video 200 señala los APS utilizados para los CTB de luminancia o crominancia de un grupo de mosaicos, porción o imagen actual. Un mosaico es una región rectangular de CTB dentro de una columna de mosaicos en particular y una fila de mosaicos en particular en una imagen.

[0082] En el decodificador de video de VTM-5.0 (por ejemplo, el decodificador de video 300), los coeficientes de filtro del ALF se reconstruyen primero. Luego, los índices de recorte se decodifican para coeficientes de filtro distintos de cero. Para coeficientes de filtro con valores de cero, el decodificador de video infiere que los índices de recorte son cero. La codificación de Golomb exponencial se utiliza para la señalización de índices de recorte. El orden de un código de Golomb exponencial para un índice de recorte depende de su posición en la plantilla de filtro.

[0083] Para ser específico, en VTM-5.0, un APS puede incluir índices de recorte para el componente de luminancia que se analizan de la siguiente manera, donde `alf_luma_clip_idx` especifica un índice de recorte:

<code>si(alf_luma_clip_flag) {</code>	
<code> alf_luma_clip_min_eg_order_minus1</code>	<code>ue(v)</code>
<code> para(i = 0; i < 3; i++)</code>	
<code> alf_luma_clip_eg_order_increase_flag[i]</code>	<code>u(1)</code>
<code> para(sflidx = 0; sflidx <=</code>	
<code> alf_luma_num_filters_signalled_minus1; sflidx++) {</code>	
<code> si(alf_luma_coeff_flag[sflidx]) {</code>	
<code> para (j = 0; j < 12; j++) {</code>	
<code> si(filtCoeff[sflidx][j])</code>	
<code> alf_luma_clip_idx[sflidx][j]</code>	<code>uek(v)</code>
<code> }</code>	
<code> }</code>	
<code> }</code>	
<code> }</code>	

[0084] El Borrador 5 de VVC proporciona la siguiente semántica para los elementos de sintaxis que se muestran en la tabla de sintaxis anterior:

alf_luma_clip_min_eg_order_minus1 más 1 especifica el orden mínimo del código de Golomb exponencial para la señalización del índice de recorte de luminancia. El valor de `alf_luma_clip_min_eg_order_minus1` debe estar en el rango de 0 a 6, inclusive.

alf_luma_clip_eg_order_increase_flag[i] igual a 1 especifica que el orden mínimo del código de Golomb exponencial para la señalización del índice de recorte de luminancia aumenta en 1. `alf_luma_clip_eg_order_increase_flag[ i ]`

igual a 0 especifica que el orden mínimo del código de Golomb exponencial para la señalización del índice de recorte de luminancia no aumenta en 1.

El orden $kClipY[i]$ del código de Golomb exponencial utilizado para decodificar los valores de $alf_luma_clip_idx[sfIdx][j]$ deriva de la siguiente manera:

$$kClipY[i] = (i == 0 ? alf_luma_clip_min_eg_order_minus1 + 1 : kClipY[i - 1]) + alf_luma_clip_eg_order_increase_flag[i] \quad (7-74)$$

$alf_luma_clip_idx[sfIdx][j]$ especifica el índice de recorte del valor de recorte que se utilizará antes de multiplicar por el j -ésimo coeficiente del filtro de luminancia señalado indicado mediante $sfIdx$. Cuando $alf_luma_clip_idx[sfIdx][j]$ no está presente, se infiere que es igual a 0 (sin recorte). Es un requisito de la conformidad de flujo de bits que los valores de $alf_luma_clip_idx[sfIdx][j]$ con $sfIdx = 0..alf_luma_num_filters_signalled_minus1$ y $j = 0..11$ estén en el rango de 0 a 3, inclusive.

El orden k de la binarización de Golomb exponencial $uek(v)$ deriva de la siguiente manera:

$$k = kClipY[golombOrderIdxY[j]] \quad (7-75)$$

La variable $filterClips[sfIdx][j]$ con $sfIdx = 0..alf_luma_num_filters_signalled_minus1$, $j = 0..11$ se inicializa de la siguiente manera:

$$filterClips[sfIdx][j] = Round(2^{(BitDepthY * (4 - alf_luma_clip_idx[sfIdx][j]) / 4)}) \quad (7-76)$$

Los valores de recorte de filtro de luminancia $AlfClip_L[adaptation_parameter_set_id]$ con elementos $AlfClip_L[adaptation_parameter_set_id][filtIdx][j]$, con $filtIdx = 0..NumAlfFilters - 1$ y $j = 0..11$ derivan de la siguiente manera:

$$AlfClip_L[adaptation_parameter_set_id][filtIdx][j] = filterClips[alf_luma_coeff_delta_idx[filtIdx]][j] \quad (7-77)$$

[0085] En las tablas de sintaxis de esta divulgación, $u(n)$ indica un número entero sin signo que usa n bits. Cuando la letra n en un descriptor de tipo $u(n)$ es "v" en una tabla de sintaxis, la cantidad de bits varía de una manera que depende del valor de otros elementos de sintaxis. El descriptor $tb(v)$ indica un valor binario truncado que usa hasta

maxVal bits con maxVal definido en la semántica del elemento de sintaxis. El descriptor $tu(v)$ indica un valor unario truncado que usa hasta maxVal bits con maxVal definido en la semántica del elemento de sintaxis. El descriptor $ue(v)$ indica un elemento de sintaxis codificado mediante Golomb exponencial de número entero de orden 0 sin signo con el bit de la izquierda en primer lugar. El descriptor $uek(v)$ indica un elemento de sintaxis codificado mediante Golomb exponencial de número entero de orden k sin signo con el bit de la izquierda en primer lugar. El descriptor $se(v)$ indica un elemento de sintaxis codificado mediante Golomb exponencial de número entero de orden 0 con signo con el bit de la izquierda en primer lugar.

[0086] El Borrador 5 de VVC proporciona el siguiente proceso de análisis para los elementos de sintaxis codificados mediante el descriptor $tb(v)$:

Este proceso se invoca cuando el descriptor de un elemento de sintaxis en las tablas de sintaxis de la subcláusula 7.3 es igual a $tb(v)$.

Las entradas a este proceso son bits de la RBSP y el valor máximo maxVal.

Las salidas de este proceso son valores de elementos de sintaxis.

Los elementos de sintaxis codificados como $tb(v)$ se codifican en binario truncado. Primero se determina el rango de valores posibles para el elemento de sintaxis. El rango de este elemento de sintaxis es de 0 a maxVal, inclusive, en donde maxVal es mayor o igual a 1. synVal, que es igual al valor del elemento de sintaxis, viene dado por un proceso especificado de la siguiente manera:

```

thVal = 1
th = -1
mientras( thVal <= maxVal ) {
    th++
    thVal <<= 1
}
val = 1 << th
b = maxVal - val
synVal = read_bits( th )
si( synVal >= val - b ) {
    synVal <<= 1
    synVal += read_bits( 1 )
    synVal -= val - b
}

```

(9-4)

donde el valor devuelto por `read_bits( th )` se interpreta como una representación binaria de un número entero sin signo con el bit más significativo escrito primero.

[0087] El Borrador 5 de VVC proporciona el siguiente proceso de análisis para los elementos de sintaxis codificados mediante el descriptor `tu(v)`:

Este proceso se invoca cuando el descriptor de un elemento de sintaxis en las tablas de sintaxis de la subcláusula 7.3 es igual a `tu(v)`.

Las entradas a este proceso son bits de la RBSP y el valor máximo `maxVal`.

Las salidas de este proceso son valores de elementos de sintaxis.

Los elementos de sintaxis codificados como `tu(v)` se codifican en unario truncado. Primero se determina el rango de valores posibles para el elemento de sintaxis. El rango de este elemento de sintaxis es de 0 a `maxVal` inclusive, en donde `maxVal` es mayor o igual a 1. `codeNum`, que es igual al valor del elemento de sintaxis, viene dado por un proceso especificado de la siguiente manera:

```
codeNum = 0
keepGoing = 1
for(i = 0; i < maxVal && keepGoing; i++){
    keepGoing = read_bits( 1 )
    si( keepGoing )
        codeNum ++
}
```

(9-3)

[0088] El Borrador 5 de VVC proporciona el siguiente proceso de análisis para los elementos de sintaxis codificados mediante los descriptores `ue(v)`, `uek(v)` y `se(v)`:

Este proceso se invoca cuando el descriptor de un elemento de sintaxis en las tablas de sintaxis es igual a `ue(v)`, `uek(v)` o `se(v)`.

Las entradas a este proceso son bits de la RBSP.

Las salidas de este proceso son valores de elementos de sintaxis.

Los elementos de sintaxis codificados como `ue(v)` o `se(v)` se codifican mediante Golomb exponencial con el orden `k` igual a 0 y los elementos de sintaxis codificados como `uek(v)` se codifican mediante Golomb exponencial con el orden `k`. El proceso de análisis para estos elementos de sintaxis comienza con la lectura de los bits a partir de la ubicación actual en el flujo de bits hasta el primer

bit distinto de cero inclusive y el recuento de la cantidad de bits iniciales que son iguales a 0. Este proceso se especifica de la siguiente manera:

```

leadingZeroBits = -1
para( b = 0; !b; leadingZeroBits++ )
    b = read_bits( 1 )

```

(9-1)

La variable codeNum se asigna entonces de la siguiente manera:

$$\text{codeNum} = (2^{\text{leadingZeroBits} - 1}) * 2^k + \text{read_bits}(\text{leadingZeroBits} + k) \quad (9-2)$$

donde el valor devuelto por read_bits(leadingZeroBits) se interpreta como una representación binaria de un número entero sin signo con el bit más significativo escrito primero.

La Tabla 9-1 ilustra la estructura del código de Golomb exponencial de orden 0 al separar la cadena de bits en bits de "prefijo" y "sufijo". Los bits de "prefijo" son aquellos bits que se analizan como se especificó anteriormente para el cálculo de leadingZeroBits, y se muestran como 0 o 1 en la columna de cadena de bits de la Tabla 9-1. Los bits de "sufijo" son aquellos bits que se analizan en el cálculo de codeNum y se muestran como x_i en la Tabla 9-1, con i en el rango de 0 a $\text{leadingZeroBits} - 1$, inclusive. Cada x_i es igual a 0 o 1.

Tabla 9-1 – Cadenas de bits con bits de "prefijo" y "sufijo" y asignación a rangos de codeNum (informativo)

Forma de cadena de bits	Rango de codeNum
1	0
0 1 x_0	1..2
0 0 1 $x_1 x_0$	3..6
0 0 0 1 $x_2 x_1 x_0$	7..14
0 0 0 0 1 $x_3 x_2 x_1 x_0$	15..30
0 0 0 0 0 1 $x_4 x_3 x_2 x_1 x_0$	31..62
...	...

La Tabla 9-2 ilustra explícitamente la asignación de cadenas de bits a valores de codeNum.

Tabla 9-2 – Cadenas de bits de Golomb exponencial y codeNum en forma explícita y utilizados como ue(v) (informativo)

Cadena de bits	codeNum
1	0
0 1 0	1
0 1 1	2
0 0 1 0 0	3
0 0 1 0 1	4
0 0 1 1 0	5
0 0 1 1 1	6
0 0 0 1 0 0 0	7
0 0 0 1 0 0 1	8
0 0 0 1 0 1 0	9
...	...

Según el descriptor, el valor de un elemento de sintaxis deriva de la siguiente manera:

- Si el elemento de sintaxis está codificado como ue(v), el valor del elemento de sintaxis es igual a codeNum.
- De otro modo (el elemento de sintaxis está codificado como se(v)), el valor del elemento de sintaxis deriva al invocar el proceso de mapeo para los códigos de Golomb exponencial con signo como se especifica en la cláusula 9.2.2 con codeNum como entrada.

9.2.2 Proceso de mapeo para códigos de Golomb exponencial con signo

La entrada a este proceso es codeNum como se especifica en la cláusula 9.2. La salida de este proceso es un valor de un elemento de sintaxis codificado como se(v).

El elemento de sintaxis se asigna al codeNum al ordenar el elemento de sintaxis por su valor absoluto en orden creciente y representar el valor positivo para un valor absoluto dado con el codeNum más bajo. La Tabla 9-3 proporciona la regla de asignación.

Tabla 9-3 – Asignación de elemento de sintaxis a codeNum para elementos de sintaxis codificados mediante Golomb exponencial con signo se(v)

codeNum	valor de elemento de sintaxis
0	0
1	1
2	-1
3	2
4	-2
5	3
6	-3
k	$(-1)^{k+1} \text{Ceil}(k \div 2)$

[0089] En el Borrador 5 de VVC, los índices de recorte para los componentes de crominancia se analizan de la siguiente manera:

si(alf_chroma_clip_flag) {	
alf_chroma_clip_min_eg_order_minus1	ue(v)
para(i = 0; i < 2; i++)	
alf_chroma_clip_eg_order_increase_flag[i]	u(1)
para(j = 0; j < 6; j++) {	
si(alf_chroma_coeff_abs[j])	
alf_chroma_clip_idx[j]	uek(v)
}	
}	

[0090] El Borrador 5 de VVC proporciona la siguiente semántica para los elementos de sintaxis que se muestran en la tabla de sintaxis anterior:

alf_chroma_clip_flag igual a 0 especifica que el filtrado en bucle adaptativo lineal se aplica a los componentes de crominancia; **alf_chroma_clip_flag** igual a 1 especifica que el filtrado en bucle adaptativo no lineal se aplica al componente de crominancia. Cuando no está presente, se infiere que **alf_chroma_clip_flag** es igual a 0.

alf_chroma_min_eg_order_minus1 más 1 especifica el orden mínimo del código de Golomb exponencial para la señalización del coeficiente de filtro de crominancia. El valor de **alf_chroma_min_eg_order_minus1** debe estar en el rango de 0 a 6, inclusive.

alf_chroma_eg_order_increase_flag[i] igual a 1 especifica que el orden mínimo del código de Golomb exponencial para la señalización del coeficiente de filtro de crominancia aumenta en 1. **alf_chroma_eg_order_increase_flag[i]** igual a 0 especifica que el orden mínimo del código de Golomb exponencial para la señalización del coeficiente de filtro de crominancia no aumenta en 1.

El orden **expGoOrderC[i]** del código de Golomb exponencial utilizado para decodificar los valores de **alf_chroma_coeff_abs[j]** deriva de la siguiente manera:

$$\text{expGoOrderC}[i] = (i = 0 \quad ? \quad \text{alf_chroma_min_eg_order_minus1} + 1 : \text{expGoOrderC}[i - 1]) + \quad (7-78)$$

$$\text{alf_chroma_eg_order_increase_flag}[i] \quad (7-79)$$

alf_chroma_coeff_abs[j] especifica el valor absoluto del j-ésimo coeficiente de filtro de crominancia. Cuando **alf_chroma_coeff_abs[j]** no está presente, se infiere que es igual a 0. Es un requisito de la conformidad de flujo de bits que los valores de **alf_chroma_coeff_abs[j]** estén en el rango de 0 a $2^7 - 1$, inclusive.

El orden k de la binarización de Golomb exponencial $uek(v)$ deriva de la siguiente manera:

$$\text{golombOrderIdxC}[] = \{ 0, 0, 1, 0, 0, 1 \} \quad (7-80)$$

$$k = \text{expGoOrderC}[\text{golombOrderIdxC}[j]] \quad (7-81)$$

alf_chroma_coeff_sign[j] especifica el signo del j -ésimo coeficiente de filtro de crominancia de la siguiente manera:

–Si **alf_chroma_coeff_sign[j]** es igual a 0, el coeficiente de filtro de crominancia correspondiente tiene un valor positivo.

–De otro modo (**alf_chroma_coeff_sign[j]** es igual a 1), el coeficiente de filtro de crominancia correspondiente tiene un valor negativo.

Cuando **alf_chroma_coeff_sign[j]** no está presente, se infiere que es igual a 0.

Los coeficientes de filtro de crominancia **AlfCoeffC[adaptation_parameter_set_id]** con elementos **AlfCoeffC[adaptation_parameter_set_id][j]**, con $j = 0..5$ derivan de la siguiente manera:

$$\begin{aligned} \text{AlfCoeffC}[\text{adaptation_parameter_set_id}][j] &= \text{alf_chroma_coeff_abs}[j] * \\ &\quad (7-82) \\ &\quad (1 - 2 * \text{alf_chroma_coeff_sign}[j]) \end{aligned}$$

Es un requisito de la conformidad de flujo de bits que los valores de **AlfCoeffC[adaptation_parameter_set_id][j]** con $j = 0..5$ estén en el rango de $-2^7 - 1$ a $2^7 - 1$, inclusive.

alf_luma_clip_min_eg_order_minus1 más 1 especifica el orden mínimo del código de Golomb exponencial para la señalización del índice de recorte de crominancia. El valor de **alf_chroma_clip_min_eg_order_minus1** debe estar en el rango de 0 a 6, inclusive.

alf_chroma_clip_eg_order_increase_flag[i] igual a 1 especifica que el orden mínimo del código de Golomb exponencial para la señalización del índice de recorte de crominancia aumenta en 1. **alf_chroma_clip_eg_order_increase_flag[i]** igual a 0 especifica que el orden mínimo del código de Golomb exponencial para la señalización del índice de recorte de crominancia no aumenta en 1.

El orden **expGoOrderC[i]** del código de Golomb exponencial utilizado para decodificar los valores de **alf_chroma_clip_idx[j]** deriva de la siguiente manera:

$$\begin{aligned} k\text{ClipC}[i] &= (i == 0 ? \text{alf_chroma_clip_min_eg_order_minus1} + 1 : \\ &\quad k\text{ClipC}[i - 1]) + \end{aligned} \quad (7-83)$$

$$\text{alf_chroma_clip_eg_order_increase_flag}[i] \quad (7-84)$$

alf_chroma_clip_idx[j] especifica el índice de recorte del valor de recorte que se utilizará antes de multiplicar por el j-ésimo coeficiente del filtro de crominancia. Cuando **alf_chroma_clip_idx[j]** no está presente, se infiere que es igual a 0 (sin recorte). Es un requisito de la conformidad de flujo de bits que los valores de **alf_chroma_clip_idx[j]** con $j = 0..5$ estén en el rango de 0 a 3, inclusive.

El orden k de la binarización de Golomb exponencial $\text{uek}(v)$ deriva de la siguiente manera:

$$k = \text{kClipC}[\text{golombOrderIdxC}[j]] \quad (7-85)$$

Los valores de recorte de filtro de crominancia **AlfClipC[adaptation_parameter_set_id]** con elementos **AlfClipC[adaptation_parameter_set_id][j]**, con $j = 0..5$ derivan de la siguiente manera:

$$\text{AlfClipC}[\text{adaptation_parameter_set_id}][j] = \text{Round}(2^{(\text{BitDepthC} - 8)} * 2^{(8 * (3 - \text{alf_chroma_clip_idx}[j]) / 3)}) \quad (7-86)$$

[0091] Como se describió anteriormente, para analizar índices de recorte, primero se reconstruyen los coeficientes de filtro. Además, la codificación de Golomb exponencial recursiva se utiliza para analizar índices de recorte. La reconstrucción en primer lugar de los coeficientes de filtro y el uso de la codificación de Golomb exponencial recursiva pueden aumentar el retardo en el decodificador de video 300.

[0092] Los aspectos de esta divulgación describen ejemplos que pueden simplificar el análisis de índices de recorte para los componentes de luminancia y crominancia. Los aspectos y ejemplos de esta divulgación se pueden usar por separado o en combinación. Los aspectos de esta divulgación pueden disminuir el retardo en el decodificador de video 300 y, en algunos ejemplos, la trayectoria de decodificación del codificador de video 200.

[0093] En un primer aspecto de esta divulgación, un índice de recorte $\text{clipIdx}(k,l)$ siempre es señalado/analizado (por ejemplo, por el codificador de video 200 o el decodificador de video 300, respectivamente), incluso si el correspondiente coeficiente de filtro $f(k,l)$ es cero. En otras palabras, se puede eliminar la condición de que el correspondiente coeficiente de filtro no es cero. Por ejemplo, con referencia a las tablas de sintaxis anteriores, se pueden eliminar las líneas "si($\text{filtCoeff}[\text{sIdx}][j]$)" y "si($\text{alf_chroma_coeff_abs}[j]$)". De esta forma, un encriptador de video (por ejemplo, el codificador de video 200 o el decodificador de video 300) puede encriptar el índice de

recorte de ALF independientemente de un valor de un coeficiente de filtro correspondiente del ALF. Por ejemplo, el encriptador de video puede determinar que el correspondiente coeficiente de filtro es igual a 0 y aún encripta (por ejemplo, codifica o decodifica) el índice de recorte de ALF.

[0094] En un segundo aspecto de esta divulgación, se elimina la codificación de Golomb exponencial recursiva. Por ejemplo, en un ejemplo del segundo aspecto de esta divulgación, la codificación de longitud fija se puede utilizar para señalar índices de recorte, en donde x es la longitud de cada palabra del código, como se muestra en las tablas de sintaxis a continuación.

si(alf_luma_clip_flag) {	
para(sflidx = 0; sflidx <= alf_luma_num_filters_signalled_minus1;	
sflidx++) {	
para (j = 0; j < 12; j++) {	
alf_luma_clip_idx [sflidx][j]	u(x)
}	
}	
}	
si(alf_chroma_clip_flag) {	
para(j = 0; j < 6; j++) {	
alf_chroma_clip_idx [j]	u(x)
}	
}	

A diferencia del Borrador 5 de VVC, en donde alf_luma_clip_idx y alf_chroma_clip_idx tienen descriptores uek(v) (que indican una codificación de Golomb exponencial de orden k con número entero sin signo), alf_luma_clip_idx y alf_chroma_clip_idx tienen descriptores u(x) en este ejemplo (que indican un número entero sin signo que usa x bits). Las tablas de sintaxis de este ejemplo pueden formar parte de una tabla de sintaxis de un APS.

[0095] Por lo tanto, en este ejemplo, un encriptador de video (por ejemplo, el codificador de video 200 o el decodificador de video 300) puede encriptar un índice de recorte de ALF como un número entero sin signo de longitud fija. Además, el encriptador de video puede aplicar, en función del índice de recorte de ALF, un ALF a un bloque de una imagen de los datos de video.

[0096] En algunos ejemplos del segundo aspecto de esta divulgación, la codificación binaria truncada se puede utilizar para señalar índices de recorte. Por ejemplo, alf_luma_clip_idx y/o alf_chroma_clip_idx pueden tener descriptores tb(v).

[0097] En algunos ejemplos del segundo aspecto de esta divulgación, se puede utilizar Golomb de orden 0 para señalar índices de recorte. Por ejemplo, alf_luma_clip_idx y/o

alf_chroma_clip_idx pueden tener descriptores ue(v).

[0098] En algunos ejemplos del segundo aspecto de esta divulgación, se puede utilizar Golomb de orden x para señalar índices de recorte, donde x es el orden de codificación de Golomb. En algunos de tales ejemplos, x puede depender del componente. En otras palabras, pueden existir valores diferentes de x para los componentes de luminancia y crominancia. x puede ser igual para todos los índices de recorte en uno o múltiples componentes. En otros ejemplos, x puede depender de los índices de filtro y x puede ser igual para todos los índices de recorte en uno o múltiples filtros.

[0099] En algunos ejemplos del segundo aspecto de esta divulgación, la codificación unaria truncada se puede utilizar para señalar índices de recorte. Por ejemplo, alf_luma_clip_idx y/o alf_chroma_clip_idx pueden tener descriptores tu(v).

[0100] En algunos ejemplos del segundo aspecto de esta divulgación, el método de encriptación puede ser diferente entre los componentes de color. Por ejemplo, alf_luma_clip_idx y alf_chroma_clip_idx se pueden encriptar mediante el uso de diferentes codificaciones de binaria truncada, unidad truncada, número entero sin signo que usa n bits, Golomb exponencial de orden 0 con número entero sin signo y Golomb exponencial de orden k con número entero sin signo.

[0101] Por lo tanto, un encriptador de video (por ejemplo, el codificador de video 200 o el decodificador de video 300) puede encriptar (por ejemplo, codificar o decodificar) un índice de recorte de ALF como uno de: un número entero sin signo de longitud fija (descriptor u(x)), un valor binario truncado (descriptor tb(v)), un valor unario truncado (descriptor tu(v)) o un valor codificado mediante Golomb exponencial de orden 0 sin signo (descriptor ue(v)). El encriptador de video puede aplicar, en función del índice de recorte de ALF, un ALF a un bloque de una imagen de los datos de video. El encriptador de video puede realizar esto para cualquiera o ambos de los componentes de luminancia y crominancia. Por lo tanto, el índice de recorte de ALF puede ser un índice de recorte de ALF de luminancia o índice de recorte de ALF de crominancia. Con respecto al primer aspecto de esta divulgación, el encriptador de video puede encriptar el índice de recorte de ALF independientemente de un valor de un coeficiente de filtro correspondiente del ALF.

[0102] En algunos ejemplos, para aplicar el ALF en función del índice de recorte de ALF, el encriptador de video puede determinar un valor de recorte en función del índice de recorte como se indica en la ecuación (1') o (1'') anteriormente. Entonces, el encriptador de video puede usar este valor de recorte en la ecuación (1), como se indicó anteriormente. El encriptador de video puede determinar los coeficientes de filtro como

se establece en otra parte de esta divulgación o de otra manera.

[0103] Esta divulgación en general puede referirse a "señalizar" cierta información, tal como elementos de sintaxis. El término "señalización" puede referirse generalmente a la comunicación de valores para elementos de sintaxis y/u otros datos usados para decodificar datos de video codificados. Es decir, el codificador de video 200 puede señalar valores para elementos de sintaxis en el flujo de bits. En general, la señalización se refiere a generar un valor en el flujo de bits. Como se señaló anteriormente, el dispositivo fuente 102 puede transportar el flujo de bits al dispositivo de destino 116 sustancialmente en tiempo real, o no en tiempo real, como podría ocurrir cuando se almacenan elementos de sintaxis en el dispositivo de almacenamiento 112 para su posterior recuperación por el dispositivo de destino 116.

[0104] Las Figuras 5A y 5B son diagramas conceptuales que ilustran una estructura de árbol binario de árbol cuaternario (QTBT) 130 de ejemplo y una unidad de árbol de codificación (CTU) 132 correspondiente. Las líneas continuas representan la división de árbol cuaternario y las líneas de puntos indican la división de árbol binario. En cada nodo dividido (es decir, no hoja) del árbol binario, se señala una bandera para indicar qué tipo de división (es decir, horizontal o vertical) se utiliza, donde 0 indica división horizontal y 1 indica división vertical en este ejemplo. Para la división de árbol cuaternario, no es necesario indicar el tipo de división, ya que los nodos de árbol cuaternario dividen un bloque horizontal y verticalmente en 4 subbloques con el mismo tamaño. En consecuencia, el codificador de video 200 puede codificar, y el decodificador de video 300 puede decodificar, elementos de sintaxis (tal como dividir información) para un nivel de árbol de región (es decir, el primer nivel) de la estructura QTBT 130 (es decir, las líneas continuas) y elementos de sintaxis (tal como dividir información) para un nivel de árbol de predicción (es decir, el segundo nivel) de la estructura QTBT 130 (es decir, las líneas discontinuas). El codificador de video 200 puede codificar, y el decodificador de video 300 puede decodificar, datos de video, tales como datos de predicción y transformada, para CU representadas por nodos hoja terminal de la estructura QTBT 130.

[0105] En general, la CTU 132 de la Figura 5B puede estar asociada con parámetros que definen tamaños de bloques correspondientes a nodos de la estructura QTBT 130 en el primer y segundo nivel. Estos parámetros pueden incluir un tamaño de CTU (que representa un tamaño de CTU 132 en muestras), un tamaño mínimo de árbol cuaternario (MinQTSIZE, que representa un tamaño mínimo de nodo hoja de árbol cuaternario permitido), un tamaño máximo de árbol binario (MaxBTSIZE, que representa

un tamaño de nodo raíz de árbol binario máximo permitido), una profundidad máxima de árbol binario (MaxBTDepth, que representa una profundidad máxima permitida de árbol binario) y un tamaño mínimo de árbol binario (MinBTSize, que representa el tamaño mínimo permitido de nodo hoja de árbol binario).

[0106] El nodo raíz de una estructura QTBT correspondiente a una CTU puede tener cuatro nodos secundarios en el primer nivel de la estructura QTBT, cada uno de los cuales puede particionarse de acuerdo con la partición de árbol cuaternario. Es decir, los nodos del primer nivel son nodos hoja (que no tienen nodos secundarios) o tienen cuatro nodos secundarios. El ejemplo de la estructura QTBT 130 representa tales nodos que incluyen el nodo principal y los nodos secundarios que tienen líneas continuas para las ramas. Si los nodos del primer nivel no son más grandes que el tamaño máximo permitido del nodo raíz del árbol binario (MaxBTSize), entonces los nodos se pueden dividir aún más mediante los respectivos árboles binarios. La división del árbol binario de un nodo se puede iterar hasta que los nodos resultantes de la división alcancen el tamaño mínimo permitido del nodo hoja del árbol binario (MinBTSize) o la profundidad máxima permitida del árbol binario (MaxBTDepth). El ejemplo de la estructura QTBT 130 representa tales nodos que tienen líneas discontinuas para las ramas. El nodo hoja del árbol binario se denomina unidad de codificación (CU), que se utiliza para la predicción (por ejemplo, predicción intraimagen o interimagen) y transformada, sin más particiones. Como se analizó anteriormente, las CU también se pueden denominar "bloques de video" o "bloques".

[0107] En un ejemplo de la estructura de partición QTBT, el tamaño de CTU se establece en 128x128 (muestras de luminancia y dos muestras de crominancia de 64x64 correspondientes), MinQTSIZE se establece en 16x16, MaxBTSize se establece en 64x64, MinBTSize (para ancho y altura) se establece en 4 y MaxBTDepth se establece en 4. La partición del árbol cuaternario se aplica primero a la CTU para generar nodos hoja de árbol cuaternario. Los nodos hoja de árbol cuaternario pueden tener un tamaño de 16x16 (es decir, MinQTSIZE) a 128x128 (es decir, el tamaño de CTU). Si el nodo hoja de árbol cuaternario es 128x128, el árbol binario no lo dividirá más, ya que el tamaño excede el MaxBTSize (es decir, 64x64, en este ejemplo). De lo contrario, el árbol binario dividirá aún más el nodo hoja de árbol cuaternario. Por lo tanto, el nodo hoja de árbol cuaternario es también el nodo raíz del árbol binario y tiene la profundidad del árbol binario como 0. Cuando la profundidad del árbol binario alcanza MaxBTDepth (4, en este ejemplo), no se permite más división. Cuando el nodo de árbol binario tiene un ancho igual a MinBTSize (4, en este ejemplo), implica que no se permite más división

vertical. De manera similar, un nodo de árbol binario que tiene una altura igual a MinBTSIZE implica que no se permite más división horizontal para ese nodo de árbol binario. Como se señaló anteriormente, los nodos hoja de árbol binario se denominan CU y se procesan adicionalmente de acuerdo con la predicción y la transformada sin más particiones.

[0108] La Figura 6 es un diagrama de bloques que ilustra un codificador de video 200 de ejemplo que puede realizar las técnicas de la presente divulgación. La Figura 6 se proporciona para los propósitos de explicación y no se debe considerar limitativa de las técnicas de acuerdo con lo ejemplificado y descrito de manera más amplia en esta divulgación. Para fines de explicación, esta divulgación describe el codificador de video 200 en el contexto de estándares de codificación de video tales como el estándar de codificación de video H.265 (HEVC) y el estándar de codificación de video H.266 (VVC) en desarrollo. Sin embargo, las técnicas de esta divulgación no se limitan a estos estándares de codificación de video y son aplicables generalmente a la codificación y decodificación de video.

[0109] En el ejemplo de la Figura 6, el codificador de video 200 incluye la memoria de datos de video 230, la unidad de selección de modo 202, la unidad de generación residual 204, la unidad de procesamiento de transformada 206, la unidad de cuantificación 208, la unidad de cuantificación inversa 210, la unidad de procesamiento de transformada inversa 212, la unidad de reconstrucción 214, la unidad de filtro 216, el búfer de imagen decodificada (DPB) 218 y la unidad de codificación por entropía 220. Cualquiera o todos de la memoria de datos de video 230, la unidad de selección de modo 202, la unidad de generación residual 204, la unidad de procesamiento de transformada 206, la unidad de cuantificación 208, la unidad de cuantificación inversa 210, la unidad de procesamiento de transformada inversa 212, la unidad de reconstrucción 214, la unidad de filtro 216, el DPB 218 y la unidad de codificación por entropía 220 pueden implementarse en uno o más procesadores o en circuitos de procesamiento. Además, el codificador de video 200 puede incluir procesadores o circuitos de procesamiento adicionales o alternativos para realizar estas y otras funciones.

[0110] La memoria de datos de video 230 puede almacenar los datos de video que van a codificarse por los componentes del codificador de video 200. El codificador de video 200 puede recibir los datos de video almacenados en la memoria de datos de video 230 desde, por ejemplo, la fuente de video 104 (Figura 1). El DPB 218 puede actuar como una memoria de imágenes de referencia que almacena datos de video de referencia

para su uso en la predicción de datos de video posteriores por el codificador de video 200. La memoria de datos de video 230 y el DPB 218 se pueden formar mediante cualquiera de una variedad de dispositivos de memoria, tales como memoria dinámica de acceso aleatorio (DRAM), incluso DRAM síncrona (SDRAM), RAM magnetorresistiva (MRAM), RAM resistiva (RRAM) u otros tipos de dispositivos de memoria. La memoria de datos de video 230 y el DPB 218 pueden ser proporcionados por el mismo dispositivo de memoria o por dispositivos de memoria separados. En diversos ejemplos, la memoria de datos de video 230 puede estar en un chip con otros componentes del codificador de video 200, como se ilustra, o fuera del chip con respecto a esos componentes.

[0111] En esta divulgación, la referencia a la memoria de datos de video 230 no debe interpretarse como limitada a la memoria interna del codificador de video 200, a menos que se describa específicamente como tal, o la memoria externa al codificador de video 200, a menos que se describa específicamente como tal. Más bien, la referencia a la memoria de datos de video 230 debe entenderse como una memoria de referencia que almacena datos de video que el codificador de video 200 recibe para codificar (por ejemplo, datos de video para un bloque actual que se va a codificar). La memoria 106 de la Figura 1 también puede proporcionar almacenamiento temporal de salidas de las diversas unidades del codificador de video 200.

[0112] Las diversas unidades de la Figura 6 se ilustran para ayudar a comprender las operaciones realizadas por el codificador de video 200. Las unidades pueden implementarse como circuitos de función fija, circuitos programables o una combinación de estos. Los circuitos de función fija se refieren a los circuitos que proporcionan una funcionalidad particular y están preestablecidos en las operaciones que se pueden realizar. Los circuitos programables se refieren a los circuitos que se pueden programar para realizar diversas tareas y proporcionan una funcionalidad flexible en las operaciones que se pueden realizar. Por ejemplo, los circuitos programables pueden ejecutar *software* o *firmware* que hacen que los circuitos programables funcionen de la manera definida por las instrucciones del *software* o *firmware*. Los circuitos de función fija pueden ejecutar instrucciones de *software* (por ejemplo, para recibir parámetros o parámetros de salida), pero los tipos de operaciones que realizan los circuitos de función fija son generalmente inmutables. En algunos ejemplos, una o más de las unidades pueden ser bloques de circuitos distintos (de función fija o programables) y, en algunos ejemplos, la una o más unidades pueden ser circuitos integrados.

[0113] El codificador de video 200 puede incluir unidades lógicas aritméticas (ALU), unidades de función elemental (EFU), circuitos digitales, circuitos analógicos y/o núcleos

programables, formados a partir de circuitos programables. En los ejemplos donde las operaciones del codificador de video 200 se realizan mediante el uso de *software* ejecutado por los circuitos programables, la memoria 106 (Figura 1) puede almacenar el código objeto del *software* que el codificador de video 200 recibe y ejecuta, u otra memoria dentro del codificador de video 200 (no mostrado) puede almacenar tales instrucciones.

[0114] La memoria de datos de video 230 está configurada para almacenar datos de video recibidos. El codificador de video 200 puede recuperar una imagen de los datos de video de la memoria de datos de video 230 y proporcionar los datos de video a la unidad de generación residual 204 y a la unidad de selección de modo 202. Los datos de video en la memoria de datos de video 230 pueden ser datos de video sin procesar que se van a codificar.

[0115] La unidad de selección de modo 202 incluye una unidad de estimación de movimiento 222, una unidad de compensación de movimiento 224 y una unidad de intrapredicción 226. La unidad de selección de modo 202 puede incluir unidades funcionales adicionales para realizar la predicción de video de acuerdo con otros modos de predicción. Como ejemplos, la unidad de selección de modo 202 puede incluir una unidad de paleta, una unidad de copia intrabloque (que puede ser parte de la unidad de estimación de movimiento 222 y/o la unidad de compensación de movimiento 224), una unidad afín, una unidad de modelo lineal (LM) o similares.

[0116] La unidad de selección de modo 202 generalmente coordina múltiples pasadas de codificación para probar combinaciones de parámetros de codificación y valores de distorsión de velocidad resultantes para tales combinaciones. Los parámetros de codificación pueden incluir la partición de CTU en CU, modos de predicción para las CU, tipos de transformada para datos residuales de las CU, parámetros de cuantificación para datos residuales de las CU, etc. La unidad de selección de modo 202 puede finalmente seleccionar la combinación de parámetros de codificación que tienen valores de distorsión de velocidad que son mejores que las otras combinaciones probadas.

[0117] El codificador de video 200 puede dividir una imagen recuperada de la memoria de datos de video 230 en una serie de CTU y encapsular una o más CTU dentro de un segmento. La unidad de selección de modo 202 puede dividir una CTU de la imagen de acuerdo con una estructura de árbol, tal como la estructura QTBT o la estructura de árbol cuaternario de HEVC descrita anteriormente. Como se describió anteriormente, el codificador de video 200 puede formar una o más CU a partir de la partición de una CTU de acuerdo con la estructura de árbol. Tal CU también puede denominarse

generalmente "bloque de video" o "bloque".

[0118] En general, la unidad de selección de modo 202 también controla los componentes de esta (por ejemplo, unidad de estimación de movimiento 222, unidad de compensación de movimiento 224 y unidad de intrapredicción 226) para generar un bloque de predicción para un bloque actual (por ejemplo, una CU actual, o en HEVC, la parte superpuesta de una PU y una TU). Para la interpredicción de un bloque actual, la unidad de estimación de movimiento 222 puede realizar una búsqueda de movimiento para identificar uno o más bloques de referencia que coincidan estrechamente en una o más imágenes de referencia (por ejemplo, una o más imágenes codificadas previamente almacenadas en el DPB 218). En particular, la unidad de estimación de movimiento 222 puede calcular un valor representativo de qué tan similar es un bloque de referencia potencial al bloque actual, por ejemplo, de acuerdo con la suma de diferencia absoluta (SAD), la suma de diferencias cuadradas (SSD), la diferencia absoluta media (MAD), las diferencias cuadráticas medias (MSD) o similares. La unidad de estimación de movimiento 222 generalmente puede realizar estos cálculos mediante el uso de las diferencias muestra por muestra entre el bloque actual y el bloque de referencia que se está considerando. La unidad de estimación de movimiento 222 puede identificar un bloque de referencia que tiene un valor más bajo resultante de estos cálculos, lo que indica un bloque de referencia que se asemeja más al bloque actual.

[0119] La unidad de estimación de movimiento 222 puede formar uno o más vectores de movimiento (MV) que definen las posiciones de los bloques de referencia en las imágenes de referencia con respecto a la posición del bloque actual en una imagen actual. La unidad de estimación de movimiento 222 puede entonces proporcionar los vectores de movimiento a la unidad de compensación de movimiento 224. Por ejemplo, para la interpredicción unidireccional, la unidad de estimación de movimiento 222 puede proporcionar un único vector de movimiento, mientras que para la interpredicción bidireccional, la unidad de estimación de movimiento 222 puede proporcionar dos vectores de movimiento. La unidad de compensación de movimiento 224 puede generar entonces un bloque de predicción mediante los vectores de movimiento. Por ejemplo, la unidad de compensación de movimiento 224 puede recuperar datos del bloque de referencia mediante el vector de movimiento. Como otro ejemplo, si el vector de movimiento tiene una precisión de muestra fraccionada, la unidad de compensación de movimiento 224 puede interpolar valores para el bloque de predicción de acuerdo con uno o más filtros de interpolación. Además, para la interpredicción bidireccional, la unidad de compensación de movimiento 224 puede recuperar datos para dos bloques

de referencia identificados por vectores de movimiento respectivos y combinar los datos recuperados, por ejemplo, mediante promediado muestra por muestra o promediado ponderado.

[0120] Como otro ejemplo, para la intrapredicción o codificación de intrapredicción, la unidad de intrapredicción 226 puede generar el bloque de predicción a partir de muestras vecinas al bloque actual. Por ejemplo, para los modos direccionales, la unidad de intrapredicción 226 generalmente puede combinar matemáticamente valores de muestras vecinas y completar estos valores calculados en la dirección definida a través del bloque actual para producir el bloque de predicción. Como otro ejemplo, para el modo DC, la unidad de intrapredicción 226 puede calcular un promedio de las muestras vecinas al bloque actual y generar el bloque de predicción para incluir este promedio resultante para cada muestra del bloque de predicción.

[0121] La unidad de selección de modo 202 proporciona el bloque de predicción a la unidad de generación residual 204. La unidad de generación residual 204 recibe una versión en bruto, no codificada del bloque actual desde la memoria de datos de video 230 y el bloque de predicción desde la unidad de selección de modo 202. La unidad de generación residual 204 calcula las diferencias muestra por muestra entre el bloque actual y el bloque de predicción. Las diferencias muestra por muestra resultantes definen un bloque residual para el bloque actual. En algunos ejemplos, la unidad de generación residual 204 también puede determinar diferencias entre valores de muestra en el bloque residual para generar un bloque residual mediante el uso de modulación por impulsos codificados diferencial residual (RDPCM). En algunos ejemplos, la unidad de generación residual 204 puede formarse mediante el uso de uno o más circuitos sustractores que realizan una resta binaria.

[0122] En los ejemplos donde la unidad de selección de modo 202 divide las CU en PU, cada PU puede estar asociada con una unidad de predicción de luminancia y las correspondientes unidades de predicción de crominancia. El codificador de video 200 y el decodificador de video 300 pueden soportar las PU que tienen diversos tamaños. De acuerdo con lo indicado con anterioridad, el tamaño de una CU se puede referir al tamaño del bloque de codificación de luminancia de la CU y el tamaño de una PU se puede referir al tamaño de una unidad de predicción de luminancia de la PU. Suponiendo que el tamaño de una CU particular es $2N \times 2N$, el codificador de video 200 puede soportar tamaños de PU de $2N \times 2N$ o $N \times N$ para la intrapredicción y tamaños simétricos de PU de $2N \times 2N$, $2N \times N$, $N \times 2N$, $N \times N$ o similares para la interpretación. El codificador de video 200 y el decodificador de video 300 también pueden soportar la partición

asimétrica para los tamaños de PU de $2N \times nU$, $2N \times nD$, $nL \times 2N$ y $nR \times 2N$ para la interpredicción.

[0123] En los ejemplos en donde la unidad de selección de modo no divide adicionalmente una CU en PU, cada CU puede estar asociada con un bloque de codificación de luminancia y los correspondientes bloques de codificación de crominancia. De acuerdo con lo anterior, el tamaño de una CU se puede referir al tamaño del bloque de codificación de luminancia de la CU. El codificador de video 200 y el decodificador de video 300 pueden soportar tamaños de CU de $2N \times 2N$, $2N \times N$ o $N \times 2N$.

[0124] Para otras técnicas de codificación de video, tal como una codificación de modo de copia intrabloque, una codificación de modo afín y una codificación de modo de modelo lineal (LM), como algunos ejemplos, la unidad de selección de modo 202, a través de las unidades respectivas asociadas con las técnicas de codificación, genera un bloque de predicción para el bloque actual que se está codificando. En algunos ejemplos, tal como la codificación del modo de paleta, es posible que la unidad de selección de modo 202 no genere un bloque de predicción y, en su lugar, genere elementos de sintaxis que indican la manera de reconstruir el bloque en función de una paleta seleccionada. En tales modos, la unidad de selección de modo 202 puede proporcionar estos elementos de sintaxis a la unidad de codificación por entropía 220 a codificar.

[0125] Como se describió anteriormente, la unidad de generación residual 204 recibe los datos de video para el bloque actual y el bloque de predicción correspondiente. La unidad de generación residual 204 genera entonces un bloque residual para el bloque actual. Para generar el bloque residual, la unidad de generación residual 204 calcula las diferencias muestra por muestra entre el bloque de predicción y el bloque actual.

[0126] La unidad de procesamiento de transformada 206 aplica una o más transformadas al bloque residual para generar un bloque de coeficientes de transformada (denominado en la presente "bloque de coeficientes de transformada"). La unidad de procesamiento de transformada 206 puede aplicar diversas transformadas a un bloque residual para formar el bloque de coeficientes de transformada. Por ejemplo, la unidad de procesamiento de transformada 206 puede aplicar una transformada de coseno discreta (DCT), una transformada direccional, una transformada de Karhunen-Loeve (KLT) o una transformada conceptualmente similar a un bloque residual. En algunos ejemplos, la unidad de procesamiento de transformada 206 puede realizar múltiples transformadas en un bloque residual, por ejemplo, una transformada primaria y una transformada secundaria, tal como una transformada rotacional. En algunos

ejemplos, la unidad de procesamiento de transformada 206 no aplica transformadas a un bloque residual.

[0127] La unidad de cuantificación 208 puede cuantificar los coeficientes de transformada en un bloque de coeficientes de transformada, para producir un bloque de coeficientes de transformada cuantificado. La unidad de cuantificación 208 puede cuantificar los coeficientes de transformada de un bloque de coeficientes de transformada de acuerdo con un valor de parámetro de cuantificación (QP) asociado con el bloque actual. El codificador de video 200 (por ejemplo, mediante la unidad de selección de modo 202) puede ajustar el grado de cuantificación aplicado a los bloques de coeficientes de transformada asociados con el bloque actual al ajustar el valor de QP asociado con la CU. La cuantificación puede introducir pérdida de información y, por lo tanto, los coeficientes de transformada cuantificados pueden tener una precisión menor que los coeficientes de transformada originales producidos por la unidad de procesamiento de transformada 206.

[0128] La unidad de cuantificación inversa 210 y la unidad de procesamiento de transformada inversa 212 pueden aplicar cuantificación inversa y transformadas inversas a un bloque de coeficientes de transformada cuantificado, respectivamente, para reconstruir un bloque residual a partir del bloque de coeficientes de transformada. La unidad de reconstrucción 214 puede producir un bloque reconstruido correspondiente al bloque actual (aunque potencialmente con algún grado de distorsión) en función del bloque residual reconstruido y un bloque de predicción generado por la unidad de selección de modo 202. Por ejemplo, la unidad de reconstrucción 214 puede agregar muestras del bloque residual reconstruido a las muestras correspondientes del bloque de predicción generado por la unidad de selección de modo 202 para producir el bloque reconstruido.

[0129] La unidad de filtro 216 puede realizar una o más operaciones de filtro en bloques reconstruidos. Por ejemplo, la unidad de filtro 216 puede realizar operaciones de desbloqueo para reducir los artefactos de bloqueo a lo largo de los bordes de las CU. Las operaciones de la unidad de filtro 216 pueden omitirse, en algunos ejemplos. De acuerdo con un ejemplo de esta divulgación, la unidad de filtro 216 puede aplicar, en función de un índice de recorte de ALF codificado como un número entero sin signo de longitud fija, un ALF a un bloque de una imagen de los datos de video.

[0130] El codificador de video 200 almacena bloques reconstruidos en el DPB 218. Por ejemplo, en los ejemplos en los que no se necesitan las operaciones de la unidad de filtro 216, la unidad de reconstrucción 214 puede almacenar bloques reconstruidos en

el DPB 218. En los ejemplos en los que se necesitan las operaciones de la unidad de filtro 216, la unidad de filtro 216 puede almacenar los bloques reconstruidos filtrados en el DPB 218. La unidad de estimación de movimiento 222 y la unidad de compensación de movimiento 224 pueden recuperar una imagen de referencia del DPB 218, formada a partir de los bloques reconstruidos (y potencialmente filtrados), para interprededir bloques de imágenes codificadas posteriormente. Además, la unidad de intrapredicción 226 puede usar bloques reconstruidos en el DPB 218 de una imagen actual para intraprededir otros bloques en la imagen actual.

[0131] En general, la unidad de codificación por entropía 220 puede codificar por entropía elementos de sintaxis recibidos de otros componentes funcionales del codificador de video 200. Por ejemplo, la unidad de codificación por entropía 220 puede codificar por entropía bloques de coeficientes de transformada cuantificados a partir de la unidad de cuantificación 208. Como otro ejemplo, la unidad de codificación por entropía 220 puede codificar por entropía elementos de sintaxis de predicción (por ejemplo, información de movimiento para la interpredicción o información de intramodo para la intrapredicción) a partir de la unidad de selección de modo 202. La unidad de codificación por entropía 220 puede realizar una o más operaciones de codificación por entropía en los elementos de sintaxis, que son otro ejemplo de datos de video, para generar datos codificados por entropía. Por ejemplo, la unidad de codificación por entropía 220 puede llevar a cabo una operación de codificación adaptativa según el contexto de longitud variable (CAVLC), una operación de CABAC, una operación de codificación de longitud de variable a variable (V2V), una operación de codificación aritmética binaria adaptativa según el contexto con base en la sintaxis (SBAC), una operación de codificación por Entropía de Partición de Intervalo de Probabilidad (PIPE), una operación de exponencial de codificación Golomb, u otro tipo de operación de codificación por entropía sobre los datos. En algunos ejemplos, la unidad de codificación por entropía 220 puede funcionar en modo de derivación donde los elementos de sintaxis no están codificados por entropía.

[0132] El codificador de video 200 puede generar un flujo de bits que incluye los elementos de sintaxis codificados por entropía necesarios para reconstruir bloques de un corte o imagen. En particular, la unidad de codificación por entropía 220 puede generar el flujo de bits.

[0133] Las operaciones descritas anteriormente se describen con respecto a un bloque. Tal descripción debe entenderse como operaciones para un bloque de codificación de luminancia y/o bloques de codificación de crominancia. Como se describió

anteriormente, en algunos ejemplos, el bloque de codificación de luminancia y los bloques de codificación de crominancia son componentes de luminancia y crominancia de una CU. En algunos ejemplos, el bloque de codificación de luminancia y los bloques de codificación de crominancia son componentes de luminancia y crominancia de una PU.

[0134] En algunos ejemplos, las operaciones realizadas con respecto a un bloque de codificación de luminancia no necesitan repetirse para los bloques de codificación de crominancia. Como ejemplo, las operaciones para identificar un vector de movimiento (MV) y una imagen de referencia para un bloque de codificación de luminancia no necesitan repetirse para identificar un MV y una imagen de referencia para los bloques de crominancia. Más bien, el MV para el bloque de codificación de luminancia se puede escalar para determinar el MV para los bloques de crominancia, y la imagen de referencia puede ser la misma. Como otro ejemplo, el proceso de intrapredicción puede ser el mismo para el bloque de codificación de luminancia y los bloques de codificación de crominancia.

[0135] El codificador de video 200 representa un ejemplo de un dispositivo configurado para codificar datos de video que incluye una memoria configurada para almacenar datos de video, y una o más unidades de procesamiento implementadas en circuitería y configuradas para codificar un índice de recorte de ALF como uno de: un número entero sin signo de longitud fija, un valor binario truncado, un valor unario truncado o un valor codificado mediante Golomb exponencial de orden 0 sin signo. En otras palabras, el codificador de video 200 puede incluir, en un flujo de bits que incluye una representación codificada de datos de video, un elemento de sintaxis del índice de recorte de ALF, donde el elemento de sintaxis del índice de recorte de ALF se formatea como uno de estos tipos de datos. En algunos ejemplos, el índice de recorte de ALF es un índice de recorte de ALF de luminancia (por ejemplo, `alf_luma_clip_idx` u otro elemento de sintaxis) o un índice de recorte de ALF de crominancia (por ejemplo, `alf_chroma_clip_idx` u otro elemento de sintaxis). Además, las unidades de procesamiento del codificador de video 200 pueden aplicar, en función del índice de recorte de ALF, un ALF a un bloque de una imagen de los datos de video. Por ejemplo, la unidad de filtro 216 del codificador de video 200 puede aplicar el ALF.

[0136] La Figura 7 es un diagrama de bloques que ilustra un decodificador de video 300 de ejemplo que puede realizar las técnicas de la presente divulgación. La Figura 7 se proporciona con fines explicativos y no se limita a las técnicas que se ejemplifican y describen ampliamente en esta divulgación. A efectos de explicación, esta divulgación

describe el decodificador de video 300 de acuerdo con las técnicas de VVC y HEVC. Sin embargo, las técnicas de esta divulgación pueden realizarse mediante dispositivos de codificación de video que están configurados para otros estándares de codificación de video.

[0137] En el ejemplo de la Figura 7, el decodificador de video 300 incluye memoria de búfer de imagen codificada (CPB) 320, unidad de decodificación por entropía 302, unidad de procesamiento de predicción 304, unidad de cuantificación inversa 306, unidad de procesamiento de transformada inversa 308, unidad de reconstrucción 310, unidad de filtro 312 y búfer de imagen decodificada (DPB) 314. Cualquiera o todos de la memoria de CPB 320, la unidad de decodificación por entropía 302, la unidad de procesamiento de predicción 304, la unidad de cuantificación inversa 306, la unidad de procesamiento de transformada inversa 308, la unidad de reconstrucción 310, la unidad de filtro 312 y el DPB 314 pueden implementarse en uno o más procesadores o en circuitos de procesamiento. Además, el decodificador de video 300 puede incluir procesadores o circuitos de procesamiento adicionales o alternativos para realizar estas y otras funciones.

[0138] La unidad de procesamiento de predicción 304 incluye la unidad de compensación de movimiento 316 y la unidad de intrapredicción 318. La unidad de procesamiento de predicción 304 puede incluir unidades adicionales para realizar la predicción de acuerdo con otros modos de predicción. Como ejemplos, la unidad de procesamiento de predicción 304 puede incluir una unidad de codificación de paleta, una unidad de codificación de copia intrabloque (que puede formar parte de la unidad de compensación de movimiento 316), una unidad de codificación afín, una unidad de codificación de modelo lineal (LM) o similares. En otros ejemplos, el decodificador de video 300 puede incluir más, menos o diferentes componentes funcionales.

[0139] La memoria de CPB 320 puede almacenar datos de video, tales como un flujo de bits de video codificado, para ser decodificado por los componentes del decodificador de video 300. Los datos de video almacenados en la memoria de CPB 320 pueden obtenerse, por ejemplo, del medio legible por computadora 110 (Figura 1). La memoria de CPB 320 puede incluir un CPB que almacena datos de video codificados (por ejemplo, elementos de sintaxis) de un flujo de bits de video codificado. Además, la memoria de CPB 320 puede almacenar datos de video distintos de los elementos de sintaxis de una imagen codificada, tales como datos temporales que representan salidas de las diversas unidades del decodificador de video 300. El DPB 314 generalmente almacena imágenes decodificadas, cuyo decodificador de video 300 puede emitir y/o

usar como datos de video de referencia al decodificar datos o imágenes subsiguientes del flujo de bits de video codificado. La memoria de CPB 320 y el DPB 314 se pueden formar mediante cualquiera de una variedad de dispositivos de memoria, tal como DRAM, que incluye SDRAM, MRAM, RRAM u otros tipos de dispositivos de memoria. La memoria de CPB 320 y el DPB 314 pueden ser proporcionados por el mismo dispositivo de memoria o por dispositivos de memoria separados. En diversos ejemplos, la memoria de CPB 320 puede estar en un chip con otros componentes del decodificador de video 300, o fuera del chip con respecto a esos componentes.

[0140] Adicional o alternativamente, en algunos ejemplos, el decodificador de video 300 puede recuperar datos de video codificados de la memoria 120 (Figura 1). Es decir, la memoria 120 puede almacenar datos como se analizó anteriormente con la memoria de CPB 320. Del mismo modo, la memoria 120 puede almacenar instrucciones que el decodificador de video 300 ejecutará, cuando algunas o todas las funciones del decodificador de video 300 se implementan en *software* para que la circuitería de procesamiento del decodificador de video 300 las ejecute.

[0141] Las diversas unidades que se muestran en la Figura 7 se ilustran para ayudar a comprender las operaciones realizadas por el decodificador de video 300. Las unidades pueden implementarse como circuitos de función fija, circuitos programables o una combinación de estos. Similar a la Figura 6, los circuitos de función fija se refieren a los circuitos que proporcionan una funcionalidad particular y están preestablecidos en las operaciones que se pueden realizar. Los circuitos programables se refieren a los circuitos que se pueden programar para realizar diversas tareas y proporcionan una funcionalidad flexible en las operaciones que se pueden realizar. Por ejemplo, los circuitos programables pueden ejecutar *software* o *firmware* que hacen que los circuitos programables funcionen de la manera definida por las instrucciones del *software* o *firmware*. Los circuitos de función fija pueden ejecutar instrucciones de *software* (por ejemplo, para recibir parámetros o parámetros de salida), pero los tipos de operaciones que realizan los circuitos de función fija son generalmente inmutables. En algunos ejemplos, una o más de las unidades pueden ser bloques de circuitos distintos (de función fija o programables) y, en algunos ejemplos, la una o más unidades pueden ser circuitos integrados.

[0142] El decodificador de video 300 puede incluir ALU, EFU, circuitos digitales, circuitos analógicos y/o núcleos programables formados a partir de circuitos programables. En los ejemplos en los que las operaciones del decodificador de video 300 se realizan mediante *software* que se ejecuta en los circuitos programables, la

memoria dentro o fuera del chip puede almacenar instrucciones (por ejemplo, código objeto) del *software* que el decodificador de video 300 recibe y ejecuta.

[0143] La unidad de decodificación por entropía 302 puede recibir datos de video codificados desde el CPB y decodificar por entropía los datos de video para reproducir elementos de sintaxis. La unidad de procesamiento de predicción 304, la unidad de cuantificación inversa 306, la unidad de procesamiento de transformada inversa 308, la unidad de reconstrucción 310, y la unidad de filtro 312 pueden generar datos de video decodificados en función de los elementos de sintaxis extraídos del flujo de bits.

[0144] En general, el decodificador de video 300 reconstruye una imagen bloque por bloque. El decodificador de video 300 puede realizar una operación de reconstrucción en cada bloque individualmente (donde el bloque que se está reconstruyendo actualmente, es decir, decodificado, puede denominarse "bloque actual").

[0145] La unidad de decodificación por entropía 302 puede decodificar por entropía elementos de sintaxis que definen coeficientes de transformada cuantificados de un bloque de coeficientes de transformada cuantificados, así como información de transformada, tal como un parámetro de cuantificación (QP) y/o indicaciones de modo de transformada. La unidad de cuantificación inversa 306 puede usar el QP asociado con el bloque de coeficientes de transformada cuantificado para determinar un grado de cuantificación e, igualmente, un grado de cuantificación inversa para que se aplique la unidad de cuantificación inversa 306. La unidad de cuantificación inversa 306 puede, por ejemplo, realizar una operación de desplazamiento a la izquierda bit a bit para cuantificar de forma inversa los coeficientes de transformada cuantificados. La unidad de cuantificación inversa 306 puede formar de ese modo un bloque de coeficientes de transformada que incluye coeficientes de transformada.

[0146] Después de que la unidad de cuantificación inversa 306 forme el bloque de coeficientes de transformada, la unidad de procesamiento de transformada inversa 308 puede aplicar una o más transformadas inversas al bloque de coeficiente de transformada para generar un bloque residual asociado con el bloque actual. Por ejemplo, la unidad de procesamiento de transformada inversa 308 puede aplicar una DCT inversa, una transformada inversa de números enteros, una transformada inversa de Karhunen-Loeve (KLT), una transformada inversa de rotación, una transformada direccional inversa u otra transformada inversa al bloque de coeficientes de transformada.

[0147] Además, la unidad de procesamiento de predicción 304 genera un bloque de predicción de acuerdo con los elementos de sintaxis de información de predicción que

fueron decodificados por entropía por la unidad de decodificación por entropía 302. Por ejemplo, si los elementos de sintaxis de información de predicción indican que el bloque actual está interpredicho, la unidad de compensación de movimiento 316 puede generar el bloque de predicción. En este caso, los elementos de sintaxis de información de predicción pueden indicar una imagen de referencia en el DPB 314 de la cual recuperar un bloque de referencia, así como un vector de movimiento que identifica una ubicación del bloque de referencia en la imagen de referencia en relación con la ubicación del bloque actual en la imagen actual. La unidad de compensación de movimiento 316 generalmente puede realizar el proceso de interpredicción de una manera que es sustancialmente similar a la descrita con respecto a la unidad de compensación de movimiento 224 (Figura 6).

[0148] Como otro ejemplo, si los elementos de sintaxis de información de predicción indican que el bloque actual está intrapredicho, la unidad de intrapredicción 318 puede generar el bloque de predicción de acuerdo con un modo de intrapredicción indicado por los elementos de sintaxis de información de predicción. De nuevo, la unidad de intrapredicción 318 puede realizar generalmente el proceso de intrapredicción de una manera que es sustancialmente similar a la descrita con respecto a la unidad de intrapredicción 226 (Figura 6). La unidad de intrapredicción 318 puede recuperar datos de muestras vecinas al bloque actual a partir del DPB 314.

[0149] La unidad de reconstrucción 310 puede reconstruir el bloque actual mediante el uso del bloque de predicción y el bloque residual. Por ejemplo, la unidad de reconstrucción 310 puede agregar muestras del bloque residual a las muestras correspondientes del bloque de predicción para reconstruir el bloque actual.

[0150] La unidad de filtro 312 puede realizar una o más operaciones de filtro en bloques reconstruidos. Por ejemplo, la unidad de filtro 312 puede realizar operaciones de desbloqueo para reducir los artefactos de bloqueo a lo largo de los bloques reconstruidos. Las operaciones de la unidad de filtro 312 no se realizan necesariamente en todos los ejemplos. De acuerdo con un ejemplo de esta divulgación, la unidad de filtro 312 puede aplicar, en función de un índice de recorte de ALF codificado como un número entero sin signo de longitud fija, un ALF a un bloque de una imagen de los datos de video.

[0151] El decodificador de video 300 puede almacenar los bloques reconstruidos en el DPB 314. Por ejemplo, en los ejemplos en los que no se realizan las operaciones de la unidad de filtro 312, la unidad de reconstrucción 310 puede almacenar bloques reconstruidos en el DPB 314. En los ejemplos en los que se realizan las operaciones de

la unidad de filtro 312, la unidad de filtro 312 puede almacenar los bloques reconstruidos filtrados en el DPB 314. Como se analizó anteriormente, el DPB 314 puede proporcionar información de referencia, tal como muestras de una imagen actual para intrapredicción e imágenes previamente decodificadas para compensación de movimiento subsiguiente, a la unidad de procesamiento de predicción 304. Además, el decodificador de video 300 puede emitir imágenes decodificadas a partir del DPB 314 para su presentación posterior en un dispositivo de visualización, tal como el dispositivo de visualización 118 de la Figura 1.

[0152] De esta forma, el decodificador de video 300 representa un ejemplo de un dispositivo de decodificación de video que incluye una memoria configurada para almacenar datos de video, y una o más unidades de procesamiento implementadas en circuitería y configuradas para decodificar un índice de recorte de ALF como uno de: un número entero sin signo de longitud fija, un valor binario truncado, un valor unario truncado o un valor codificado mediante Golomb exponencial de orden 0 sin signo. En otras palabras, el decodificador de video 300 puede obtener un elemento de sintaxis del índice de recorte de ALF de un flujo de bits e interpretar el elemento de sintaxis del índice de recorte de ALF como uno de estos tipos de datos. En algunos ejemplos, el índice de recorte de ALF es un índice de recorte de ALF de luminancia (por ejemplo, `alf_luma_clip_idx` u otro elemento de sintaxis) o un índice de recorte de ALF de crominancia (por ejemplo, `alf_chroma_clip_idx` u otro elemento de sintaxis). Además, las unidades de procesamiento del decodificador de video 300 pueden aplicar, en función del índice de recorte de ALF, un ALF a un bloque de una imagen de los datos de video. Por ejemplo, la unidad de filtro 312 del decodificador de video 300 puede aplicar el ALF.

[0153] La Figura 8 es un diagrama de flujo que ilustra un método de ejemplo para codificar un bloque actual. El bloque actual puede comprender una CU actual. Aunque se describe con respecto al codificador de video 200 (Figura 1 y 6), debe entenderse que se pueden configurar otros dispositivos para realizar un método similar al de la Figura 8.

[0154] En este ejemplo, el codificador de video 200 predice inicialmente el bloque actual (350). Por ejemplo, el codificador de video 200 puede formar un bloque de predicción para el bloque actual. El codificador de video 200 puede entonces calcular un bloque residual para el bloque actual (352). Para calcular el bloque residual, el codificador de video 200 puede calcular una diferencia entre el bloque original no codificado y el bloque de predicción para el bloque actual. El codificador de video 200 puede entonces

transformar y cuantificar los coeficientes de transformada del bloque residual (354). A continuación, el codificador de video 200 puede explorar los coeficientes de transformada cuantificados del bloque residual (356). Durante la exploración, o después de la exploración, el codificador de video 200 puede codificar por entropía los coeficientes de transformada (358). Por ejemplo, el codificador de video 200 puede codificar los coeficientes de transformada mediante el uso de CAVLC o CABAC. El codificador de video 200 puede generar entonces los datos codificados por entropía del bloque (360).

[0155] Además, en el ejemplo de la Figura 8, para soportar la predicción de los bloques posteriores, el codificador de video 200 puede reconstruir el bloque actual (362). Por ejemplo, el codificador de video 200 puede cuantificar de forma inversa los coeficientes de transformada del bloque actual, aplicar una transformada inversa a los coeficientes de transformada para generar datos residuales y agregar los datos residuales para el bloque actual al bloque de predicción del bloque actual. Además, el codificador de video 200 puede aplicar uno o más filtros a bloques reconstruidos de la imagen actual (364). Por ejemplo, el codificador de video 200 puede aplicar un ALF a bloques reconstruidos de la imagen actual. De acuerdo con una técnica de esta divulgación, para soportar la correspondiente aplicación del ALF en el decodificador de video 300, el codificador de video 200 puede codificar un índice de recorte de ALF. El codificador de video 200 (por ejemplo, la unidad de filtro 216 del codificador de video 200) puede aplicar, en función del índice de recorte de ALF, un ALF a un bloque (por ejemplo, un bloque reconstruido) de la imagen actual. De acuerdo con una técnica de esta divulgación, el codificador de video 200 puede codificar el índice de recorte de ALF como un número entero sin signo de longitud fija, un valor binario truncado, un valor unario truncado o un valor codificado mediante Golomb exponencial de orden 0 sin signo.

[0156] La Figura 9 es un diagrama de flujo que ilustra un método de ejemplo para decodificar un bloque actual de una imagen actual de datos de video. El bloque actual puede comprender una CU actual. Aunque se describe con respecto al decodificador de video 300 (Figuras 1 y 7), debe entenderse que se pueden configurar otros dispositivos para realizar un método similar al de la Figura 9.

[0157] El decodificador de video 300 puede recibir datos codificados por entropía para el bloque actual, tales como información de predicción codificada por entropía y datos codificados por entropía para coeficientes de transformada de un bloque residual correspondiente al bloque actual (370). El decodificador de video 300 también puede recibir datos no codificados por entropía en un flujo de bits. El decodificador de video

300 puede decodificar por entropía los datos codificados por entropía para determinar la información de predicción para el bloque actual y reproducir los coeficientes de transformada del bloque residual (372). El decodificador de video 300 puede predecir el bloque actual (374), por ejemplo, mediante el uso de un modo de intrapredicción o interpredicción, como lo indica la información de predicción para el bloque actual, para calcular un bloque de predicción para el bloque actual. El decodificador de video 300 puede entonces explorar de forma inversa los coeficientes de transformada reproducidos (376), para crear un bloque de coeficientes de transformada cuantificados. El decodificador de video 300 (por ejemplo, la unidad de cuantificación inversa 306 y la unidad de procesamiento de transformada inversa 308) puede entonces cuantificar de forma inversa y transformar de forma inversa los coeficientes de transformada para producir un bloque residual (378). El decodificador de video 300 puede decodificar el bloque actual al combinar el bloque de predicción y el bloque residual (380).

[0158] Además, en el ejemplo de la Figura 9, después de combinar el bloque de predicción y el bloque residual para reconstruir el bloque actual, el decodificador de video 300 (por ejemplo, la unidad de filtro 312 del decodificador de video 300) puede aplicar uno o más filtros a bloques reconstruidos de la imagen actual (382). Por ejemplo, el decodificador de video 300 puede aplicar un ALF a bloques reconstruidos de la imagen actual. El decodificador de video 300 puede decodificar un índice de recorte de ALF y aplicar, en función del índice de recorte de ALF, un ALF a un bloque (por ejemplo, un bloque reconstruido) de la imagen actual. De acuerdo con una técnica de esta divulgación, el decodificador de video 300 puede decodificar el índice de recorte de ALF como un número entero sin signo de longitud fija, un valor binario truncado, un valor unario truncado o un valor codificado mediante Golomb exponencial de orden 0 sin signo.

[0159] La Figura 10 es un diagrama de flujo que ilustra una operación de ejemplo para codificar datos de video de acuerdo con una o más técnicas de la presente divulgación. En el ejemplo de la Figura 10, un encriptador de video (por ejemplo, el codificador de video 200 o el decodificador de video 300) puede encriptar (por ejemplo, codificar o decodificar) un índice de recorte de filtro en bucle adaptativo (ALF) como un número entero sin signo de longitud fija (400). Por ejemplo, cuando el encriptador de video es un codificador de video tal como el codificador de video 200, el codificador de video puede codificar el índice de recorte de ALF mediante la inclusión de un número entero sin signo de longitud fija que representa el índice de recorte de ALF en un flujo de bits. En un ejemplo donde el encriptador de video es un decodificador de video tal como el

decodificador de video 300, el decodificador de video puede decodificar el índice de recorte de ALF mediante el análisis del número entero sin signo de longitud fija que representa el índice de recorte de ALF del flujo de bits.

[0160] Además, en el ejemplo de la Figura 10, el encriptador de video puede aplicar, en función del índice de recorte de ALF, un ALF a un bloque de una imagen de los datos de video (402). Por ejemplo, el encriptador de video puede utilizar el índice de recorte de ALF para buscar o calcular un conjunto de valores de recorte (por ejemplo, $-c(k, l)$ y $c(k, l)$), por ejemplo, mediante el uso de la ecuación (1') o la ecuación (1'') anteriores. El encriptador de video puede entonces utilizar el conjunto de valores de recorte cuando aplica un ALF a muestras reconstruidas del bloque (por ejemplo, como se muestra en la ecuación (1) anterior). El encriptador de video puede aplicar el ALF como parte de la acción 364 de la Figura 8 o la acción 382 de la Figura 9. El índice de recorte de ALF puede ser un índice de recorte de ALF de luminancia (por ejemplo, `alf_luma_clip_idx`), en cuyo caso el encriptador de video utiliza el índice de recorte de ALF de luminancia para determinar valores de recorte para su uso en la aplicación de un ALF a muestras de luminancia. En algunos ejemplos, el índice de recorte de ALF es un índice de recorte de ALF de luminancia, en cuyo caso el encriptador de video utiliza el índice de recorte de ALF de luminancia (por ejemplo, `alf_chroma_clip_idx`) para determinar valores de recorte para su uso en la aplicación de un ALF a muestras de luminancia. La aplicación de un ALF a un bloque (por ejemplo, un bloque de 4x4) de una imagen puede incluir determinar coeficientes de filtro de ALF para el bloque, usar el índice de recorte de ALF para determinar valores de recorte para al menos una muestra del bloque y usar los valores de recorte y los coeficientes de filtro de ALF, por ejemplo, como se describe en la ecuación (1).

[0161] Además, en algunos ejemplos, el encriptador de video puede encriptar un índice de recorte de ALF de luminancia como un primer número entero sin signo de longitud fija y puede encriptar un índice de recorte de ALF de crominancia como un segundo número entero sin signo de longitud fija. En tales ejemplos, el encriptador de video puede aplicar, en función del índice de recorte de ALF de luminancia, un ALF a un bloque de luminancia de la imagen y puede aplicar, en función del índice de recorte de ALF de crominancia, un ALF a un bloque de crominancia de la imagen.

[0162] La siguiente es una lista no limitativa de ejemplos de acuerdo con una o más técnicas de esta divulgación.

[0163] Ejemplo 1. Un método para encriptar datos de video, en donde el método incluye: encriptar un índice de recorte de filtro en bucle adaptativo (ALF) como uno de:

un número entero sin signo de longitud fija, un valor binario truncado, un valor unario truncado o un valor codificado mediante Golomb exponencial de orden 0 sin signo; y aplicar, en función del índice de recorte de ALF, un filtro ALF a un bloque de una imagen de los datos de video.

[0164] Ejemplo 2. El método de acuerdo con el Ejemplo 1, en donde el índice de recorte de ALF es un índice de recorte de ALF de luminancia.

[0165] Ejemplo 3. El método de acuerdo con el Ejemplo 1, en donde el índice de recorte de ALF es un índice de recorte de ALF de crominancia.

[0166] Ejemplo 4. El método de acuerdo con el Ejemplo 1, en donde: el índice de recorte de ALF es un índice de recorte de ALF de luminancia y el bloque de la imagen es un bloque de luminancia, y el método comprende además: encriptar un índice de recorte de ALF de crominancia como uno de: un número entero sin signo de longitud fija, un valor binario truncado, un valor unario truncado o un valor codificado mediante Golomb exponencial de orden 0 sin signo; y aplicar, en función del índice de recorte de ALF, un filtro ALF a un bloque de crominancia de una imagen de los datos de video.

[0167] Ejemplo 5. El método de acuerdo con el Ejemplo 4, en donde el índice de recorte de ALF de luminancia y el índice de recorte de ALF de crominancia se encriptan como diferentes de un número entero sin signo de longitud fija, un valor binario truncado, un valor unario truncado o un valor codificado mediante Golomb exponencial de orden 0 sin signo.

[0168] Ejemplo 6. El método de acuerdo con cualquiera de los Ejemplos 1-5, en donde encriptar el índice de recorte de ALF comprende encriptar el índice de recorte de ALF independientemente de un valor de un correspondiente coeficiente de filtro del filtro ALF.

[0169] Ejemplo 7. El método de acuerdo con cualquiera de los Ejemplos 1-6, en donde encriptar comprende decodificar.

[0170] Ejemplo 8. El método de acuerdo con cualquiera de los Ejemplos 1-6, en donde encriptar comprende codificar.

[0171] Ejemplo 9. Un dispositivo para encriptar datos de video, en donde el dispositivo incluye uno o más medios para realizar el método de acuerdo con cualquiera de los Ejemplos 1-8.

[0172] Ejemplo 10. El dispositivo de acuerdo con el Ejemplo 9, en donde el uno o más medios comprenden uno o más procesadores implementados en circuitería.

[0173] Ejemplo 11. El dispositivo de acuerdo con cualquiera de los Ejemplos 9 y 10, que incluye además una memoria para almacenar los datos de video.

[0174] Ejemplo 12. El dispositivo de acuerdo con cualquiera de los Ejemplos 9–11, que incluye además una pantalla configurada para mostrar datos de video decodificados.

[0175] Ejemplo 13. El dispositivo de acuerdo con cualquiera de los Ejemplos 9–12, en donde el dispositivo comprende uno o más de una cámara, una computadora, un dispositivo móvil, un dispositivo receptor de transmisión o un decodificador.

[0176] Ejemplo 14. El dispositivo de acuerdo con cualquiera de los Ejemplos 9–13, en donde el dispositivo comprende un decodificador de video.

[0177] Ejemplo 15. El dispositivo de acuerdo con cualquiera de los Ejemplos 9–14, en donde el dispositivo comprende un codificador de video.

[0178] Ejemplo 16. Un medio de almacenamiento legible por computadora que tiene almacenadas instrucciones que, cuando se ejecutan, hacen que uno o más procesadores realicen el método de acuerdo con cualquiera de los Ejemplos 1–8.

[0179] Ejemplo 17. Un dispositivo para codificar datos de video, en donde el dispositivo incluye medios para realizar los métodos de acuerdo con cualquiera de los Ejemplos 1–8.

[0180] Ejemplo 18. Un medio de almacenamiento de datos legible por computadora que tiene instrucciones almacenadas que, cuando se ejecutan, hacen que un dispositivo informático realice los métodos de acuerdo con cualquiera de los Ejemplos 1–8.

[0181] Cabe aclarar que según el ejemplo, algunos actos o eventos de cualquiera de las técnicas descritas en la presente se pueden realizar en una secuencia diferente, se pueden agregar, fusionar o excluir completamente (por ejemplo, no todos los actos o eventos descritos son necesarios para la práctica de las técnicas). Asimismo, en algunos ejemplos, los actos o eventos se pueden realizar de manera concurrente, por ejemplo, a través de procesamiento multihilo, procesamiento interrumpido o procesadores múltiples, en lugar de secuencialmente.

[0182] En uno o más ejemplos, las funciones descritas pueden implementarse en *hardware*, *software*, *firmware* o cualquier combinación de estos. Si se implementan en *software*, las funciones pueden almacenarse o transmitirse como una o más instrucciones o código en un medio legible por computadora y pueden ejecutarse mediante una unidad de procesamiento de *hardware*. Los medios legibles por computadora pueden incluir medios de almacenamiento legibles por computadora, que corresponden a un medio tangible tal como un medio de almacenamiento de datos, o medios de comunicación, incluso cualquier medio que facilite la transferencia de un programa informático de un lugar a otro, por ejemplo, de acuerdo con un protocolo de

comunicación. De esta manera, los medios legibles por computadora generalmente pueden corresponder a (1) medios de almacenamiento legibles por computadora tangibles no transitorios o (2) un medio de comunicación tal como una onda de señal o portadora. Los medios de almacenamiento de datos pueden ser cualquier medio disponible al que se puede acceder mediante una o más computadoras o uno o más procesadores para obtener instrucciones, códigos y/o estructuras de datos para la implementación de las técnicas descritas en esta divulgación. Un producto de programa informático puede incluir un medio legible por computadora.

[0183] A modo ejemplificativo, y no limitativo, tal medio de almacenamiento legible por computadora puede comprender RAM, ROM, EEPROM, CD-ROM u otros dispositivos de almacenamiento en disco óptico, de almacenamiento en disco magnético u otro almacenamiento magnético, o cualquier otro medio que pueda utilizarse para almacenar el código de programación deseado en forma de instrucciones o estructuras de datos y al que puede accederse por medio de una computadora. Además, cualquier conexión se denomina de manera adecuada medio legible por computadora. Por ejemplo, si las instrucciones se transmiten desde un sitio web, servidor u otra fuente remota mediante un cable coaxial, cable de fibra óptica, par trenzado, línea de abonado digital (DSL) o tecnologías inalámbricas tales como infrarroja, radio y microondas, entonces el cable coaxial, cable de fibra óptica, par trenzado, DSL o tecnologías inalámbricas tales como infrarroja, radio y microondas están incluidas en la definición de medio. Sin embargo, cabe aclarar que los medios de almacenamiento legibles por computadora y los medios de almacenamiento de datos no incluyen conexiones, ondas portadoras, señales u otros medios transitorios, sino que, en su lugar, se refieren a medios de almacenamiento tangibles no transitorios. Como se usa en la presente, disco incluye disco compacto (CD), disco láser, disco óptico, disco versátil digital (DVD), disquete y disco Blu-ray, en donde los discos reproducen datos magnéticamente y datos ópticamente con láseres. También deberían incluirse combinaciones de los anteriores dentro del alcance de medios legibles por computadora.

[0184] Las instrucciones pueden ser ejecutadas por uno o más procesadores, tales como uno o más DSP, microprocesadores de propósito general, ASIC, FPGA u otra circuitería lógica integrada o separada equivalente. En consecuencia, los términos "procesador" y "circuitería de procesamiento", como se usan en la presente, pueden referirse a cualquiera de las estructuras anteriores o cualquier otra estructura adecuada para la aplicación de las técnicas descritas en la presente. Además, en algunos aspectos, la funcionalidad descrita en la presente puede proporcionarse dentro de los

módulos de *hardware* y/o *software* dedicados configurados para codificar y decodificar, o incorporarse en un códec combinado. Además, las técnicas podrían implementarse totalmente en uno o más circuitos o elementos lógicos.

[0185] Las técnicas de esta divulgación se pueden implementar en una amplia variedad de dispositivos o aparatos, que incluyen un microteléfono inalámbrico, un circuito integrado (IC) o un conjunto de IC (por ejemplo, un conjunto de chips). En esta divulgación, se describen distintos componentes, módulos o unidades para enfatizar los aspectos funcionales de los dispositivos configurados para realizar las técnicas divulgadas, pero no es necesario que sean realizadas por diferentes unidades de *hardware*. Más bien, como se ha descrito anteriormente, las diversas unidades pueden ser combinadas en una unidad de *hardware* de códec o proporcionadas por una colección de unidades de *hardware* interoperativas, lo que incluye uno o más procesadores como se describe anteriormente, en conjunción con el *software* y/o *firmware* adecuado.

[0186] Se han descrito distintos ejemplos. Estos y otros ejemplos se encuentran dentro del alcance de las siguientes reivindicaciones.