

INICIALIZACIÓN DE PROBABILIDAD PARA CODIFICACIÓN DE VIDEO

[0001] La presente solicitud reivindica el beneficio de la solicitud provisional US N.º **16/816.016**, presentada el 11 de marzo de 2020, que reivindica el beneficio de la solicitud provisional US N.º **62/817.475**, presentada el 12 de marzo de 2019, la solicitud provisional US N.º **62/834.297**, presentada el 15 de abril de 2019 y la solicitud provisional US N.º **62/871.519**, presentada el 8 de julio de 2019, cuyo contenido completo se incorpora aquí por referencia.

CAMPO TÉCNICO

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

ANTECEDENTES

[0003] Las capacidades de video digital se pueden incorporar en una amplia gama de dispositivos, incluidos televisores digitales, sistemas de transmisión digital directa, sistemas de transmisión inalámbrica, asistentes digitales personales (PDA), computadoras portátiles o de escritorio, tabletas, lectores de libros electrónicos, cámaras digitales, dispositivos de grabación digital, reproductores de medios digitales, dispositivos de videojuegos, consolas de videojuegos, teléfonos celulares o de radio satelital, los llamados “teléfonos inteligentes”, dispositivos de videoconferencia, dispositivos de transmisión de video en continuo y similares. Los dispositivos de video digital implementan técnicas de codificación de video, como las 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 extensiones de dichos estándares. Los dispositivos de video pueden transmitir, recibir, codificar, decodificar y/o almacenar información de video digital de manera más eficiente mediante la implementación de tales técnicas de codificación de video.

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

imagen codificada (P o B) pueden usar predicción espacial con respecto a muestras de referencia en bloques vecinos en la misma imagen o predicción temporal con respecto a 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 la inicialización de probabilidad para la codificación aritmética para la compresión de video. Por ejemplo, un codificador de video puede establecer los estados de probabilidad iniciales de cada modelo de contexto utilizado en un codificador aritmético binario. El proceso puede invocarse al comienzo de la codificación de entropía o la decodificación de una unidad decodificable de forma independiente, como un segmento o un mosaico.

[0006] Como un ejemplo, un método para la codificación de entropía de datos de video incluye obtener un valor de inicialización predefinido para un contexto de una pluralidad de contextos utilizados en un proceso de codificación de entropía adaptable al contexto para codificar por entropía un valor para un elemento de sintaxis para una unidad codificable independientemente de datos de video; determinar, basándose en el valor de inicialización predefinido y en un dominio lineal, un estado de probabilidad inicial del contexto; y codificar por entropía, basándose en el estado de probabilidad inicial del contexto, un bin del valor para el elemento de sintaxis.

[0007] Como otro ejemplo, un dispositivo incluye una memoria que almacena datos de video; y uno o más procesadores implementados en circuitos y configurados para: obtener un valor de inicialización predefinido para un contexto de una pluralidad de contextos utilizados en un proceso de codificación de entropía adaptable al contexto para codificar por entropía un valor para un elemento de sintaxis para una unidad codificable independientemente de datos de video; determinar, basándose en el valor de inicialización predefinido y en un dominio lineal, un estado de probabilidad inicial del contexto; y codificar por entropía, basándose en el estado de probabilidad inicial del contexto, un bin del valor para el elemento de sintaxis.

[0008] Como otro ejemplo, un dispositivo incluye medios para obtener un valor de inicialización predefinido para un contexto de una pluralidad de contextos utilizados en un proceso de codificación de entropía adaptable al contexto para codificar por entropía un valor para un elemento de sintaxis para una unidad codificable independientemente de datos de video; medios para determinar, basados en el valor de inicialización predefinido y en un dominio lineal, un estado de probabilidad inicial del contexto; y

medios para la codificación de entropía, basados en el estado de probabilidad inicial del contexto, un bin del valor para el elemento de sintaxis.

[0009] Como otro ejemplo, un medio de almacenamiento legible por computadora almacena instrucciones que, cuando se ejecutan, hacen que uno o más procesadores de un codificador de video: obtengan un valor de inicialización predefinido para un contexto de una pluralidad de contextos utilizados en un proceso de codificación de entropía adaptable al contexto para codificar por entropía un valor para un elemento de sintaxis para una unidad codificable independientemente de datos de video; determinen, basándose en el valor de inicialización predefinido y en un dominio lineal, un estado de probabilidad inicial del contexto; y codifiquen por entropía, basándose en el estado de probabilidad inicial del contexto, un bin del valor para el elemento de sintaxis.

[0010] Los detalles de uno o más aspectos de la divulgación se exponen en los dibujos adjuntos y la descripción a continuación. Otras características, objetos y ventajas de las técnicas descritas en esta divulgación serán evidentes a partir de la descripción y los dibujos, y de las reivindicaciones.

BREVE DESCRIPCIÓN DE LOS DIBUJOS

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

[0012] Las FIG. 2A y 2B son diagramas conceptuales que ilustran un proceso de actualización de rango en codificación aritmética binaria.

[0013] La FIG. 3 es un diagrama conceptual que ilustra un proceso de salida en codificación aritmética binaria.

[0014] Las FIG. 4A y 4B son diagramas conceptuales que ilustran un ejemplo de estructura de árbol binario cuaternario (QTBT) y una unidad de árbol de codificación (CTU) correspondiente.

[0015] La FIG. 5 es un diagrama de bloques que ilustra un codificador de video de ejemplo que puede realizar las técnicas de esta divulgación.

[0016] La FIG. 6 es un diagrama de bloques que ilustra un codificador aritmético binario adaptativo al contexto en un codificador de video.

[0017] La FIG. 7 es un diagrama de bloques que ilustra un decodificador de video de ejemplo que puede realizar las técnicas de esta divulgación.

[0018] La FIG. 8 es un diagrama de bloques que ilustra un codificador aritmético binario adaptativo al contexto en un decodificador de video.

[0019] La FIG. 9 es un gráfico que ilustra los estados de probabilidad para representar las probabilidades de ser 1, como en HEVC.

[0020] Las FIG. 10A y 10B son gráficos que ilustran los estados de probabilidad para representar las probabilidades de ser 1, como en VVC, con la FIG. 10A que tiene 1024 estados de probabilidad (precisión de 10 bits) y la FIG. 10B que tiene 16384 estados de probabilidad (precisión de 14 bits).

[0021] Las FIG. 11A y 11B son gráficos que ilustran un mapeo de InitProbState al estado de probabilidad que se puede usar en el motor de codificación aritmética de la codificación de video versátil (VVC), con la FIG. 11A que tiene 1024 estados de probabilidad (precisión de 10 bits) y la FIG. 11B que tiene 16.384 estados de probabilidad (precisión de 14 bits).

[0022] La FIG. 12 es un gráfico que ilustra el conocimiento previo de la distribución de probabilidad del elemento de sintaxis SaoMergeFlag (1 bin) utilizado en el segmento I.

[0023] La FIG. 13A es un gráfico que ilustra el conocimiento previo de la distribución de probabilidad en la FIG. 12 representado por el estado de probabilidad en el dominio logarítmico, y la FIG. 13B es un gráfico que ilustra el estado de probabilidad del modelo y SliceQP_Y con una función lineal.

[0024] La FIG. 14A es un gráfico que ilustra el conocimiento previo de la distribución de probabilidad en la FIG. 12 representado por el estado de probabilidad en el dominio lineal; la FIG. 14B es un gráfico que ilustra el estado de probabilidad del modelo y SliceQP_Y con una función lineal sin aumentar los rangos de pendiente y desplazamiento definidos en HEVC y la VVC actual, y la FIG. 14C es un gráfico que ilustra un estado de probabilidad del modelo y SliceQP_Y con una función lineal con rangos aumentados de pendiente y desplazamiento.

[0025] La FIG. 15A es un gráfico que ilustra el conocimiento previo de la distribución de probabilidad del elemento de sintaxis SplitFlag utilizado en el segmento B; la FIG. 15B es un gráfico que ilustra el modelo de estado de probabilidad logarítmica y SliceQP_Y con una función lineal; la FIG. 15C es un gráfico que ilustra el estado de probabilidad lineal del modelo y SliceQP_Y con una función lineal que utiliza las pendientes y los desplazamientos definidos en HEVC y la VVC actual, y la FIG. 15D es un gráfico que ilustra el estado de probabilidad lineal del modelo y SliceQP_Y con una función lineal con intersección definida en SliceQP_Y igual a 32.

[0026] Las FIG. 16A – 16C son gráficos que ilustran el uso de una función lineal por partes determinada por múltiples intersecciones para ajustar las distribuciones de probabilidad.

[0027] Las FIG. 17A y 17B son gráficos que ilustran un ejemplo de mapeo de dominio cuadrático InitProbState al estado de probabilidad que puede usarse en el motor de

codificación aritmética de VVC, con la FIG. 17A que tiene 1024 estados de probabilidad (precisión de 10 bits) y la FIG. 17B que tiene 16.384 estados de probabilidad (precisión de 14 bits).

[0028] Las FIG. 18A y 18B son gráficos que ilustran ejemplos de distribuciones de probabilidad extrema.

[0029] La FIG. 19 es un diagrama de flujo que ilustra un método de ejemplo para codificar un bloque actual.

[0030] La FIG. 20 es un diagrama de flujo que ilustra un proceso de ejemplo para realizar la codificación de entropía basada en el contexto, de acuerdo con una o más técnicas de esta divulgación.

[0031] La FIG. 21 es un diagrama de flujo que ilustra un método de ejemplo para decodificar un bloque actual.

[0032] La FIG. 22 es un diagrama de flujo que ilustra un proceso de ejemplo para realizar la decodificación de entropía basada en el contexto, de acuerdo con una o más técnicas de esta divulgación.

DESCRIPCIÓN DETALLADA

[0033] En general, esta divulgación describe técnicas para la inicialización de probabilidad para la codificación aritmética para la compresión de video. Por ejemplo, un codificador de video puede establecer los estados de probabilidad iniciales de cada modelo de contexto utilizado en un codificador aritmético binario. El proceso puede invocarse al comienzo de la codificación o decodificación de entropía de una unidad decodificable independientemente, como un segmento.

[0034] Para realizar la inicialización, el codificador de video puede obtener un valor de inicialización predefinido (por ejemplo, `initValue`) para un contexto de una pluralidad de contextos utilizados en un proceso de codificación de entropía adaptable al contexto para codificar por entropía un valor para un elemento de sintaxis para segmento de datos de video. El codificador de video puede determinar, basándose en el valor de inicialización predefinido y en un dominio logarítmico, un estado de probabilidad inicial del contexto (por ejemplo, `pStateldx`). El codificador de video puede codificar por entropía, basándose en el estado de probabilidad inicial del contexto, un bin del valor para el elemento de sintaxis.

[0035] Al codificar datos de video de acuerdo con algunos estándares de video, el codificador de video puede mantener los estados de probabilidad en el dominio lineal. Por ejemplo, como se analiza con más detalle a continuación, el codificador aritmético binario adaptativo al contexto (CABAC) adoptado en la codificación de video versátil

(VVC) puede mantener estados de probabilidad de contextos en el dominio lineal. Cuando el estado de probabilidad inicial se determina en el dominio logarítmico, los codificadores de video pueden realizar varias operaciones para convertir el estado de probabilidad inicial determinado en el dominio lineal. Por ejemplo, un codificador de video puede usar una tabla de búsqueda (LUT) para convertir el estado de probabilidad inicial del dominio logarítmico al dominio lineal.

[0036] De acuerdo con una o más técnicas de esta divulgación, un codificador de video puede determinar directamente un estado de probabilidad inicial en el dominio lineal. Por ejemplo, el video puede determinar, en base al valor de inicialización predefinido, un estado de probabilidad inicial del contexto (por ejemplo, PStateldx) en el dominio lineal sin usar una LUT para convertir el estado de probabilidad inicial del dominio logarítmico al dominio lineal. De esta manera, el codificador de video puede evitar tener que almacenar y/o acceder a una LUT que traduce entre estados de probabilidad iniciales en los dominios lineal y logarítmico.

[0037] La FIG. 1 es un diagrama de bloques que ilustra un ejemplo de sistema de codificación y decodificación de video 100 que puede realizar las técnicas de esta divulgación. Las técnicas de esta divulgación están generalmente dirigidas a la codificación (codificación y/o decodificación) de 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, sin codificar, video codificado, video decodificado (por ejemplo, reconstruido) y metadatos de video, como datos de señalización.

[0038] Como se muestra en la FIG. 1, el sistema 100 incluye un dispositivo de origen 102 que proporciona datos de video codificados para ser decodificados y visualizados por un dispositivo de destino 116, en este ejemplo. En particular, el dispositivo de origen 102 proporciona los datos de video al dispositivo de destino 116 a través de un medio legible por computadora 110. El dispositivo de origen 102 y el dispositivo de destino 116 pueden comprender cualquiera de una amplia gama de dispositivos, que incluye computadoras de escritorio, computadoras portátiles (es decir, *laptops*), tabletas, decodificadores, microteléfonos como teléfonos inteligentes, televisores, cámaras, dispositivos de visualización, reproductores de medios digitales, consolas de videojuegos, dispositivos de transmisión de video en continuo o similares. En algunos casos, el dispositivo de origen 102 y el dispositivo de destino 116 pueden estar equipados para la comunicación inalámbrica y, por lo tanto, pueden denominarse dispositivos de comunicación inalámbrica.

[0039] En el ejemplo de la FIG. 1, el dispositivo de origen 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 esta divulgación, el codificador de video 200 del dispositivo de origen 102 y el decodificador de video 300 del dispositivo de destino 116 pueden configurarse para aplicar las técnicas para inicialización de probabilidad para codificación aritmética para compresión de video. Por lo tanto, el dispositivo de origen 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 de origen y un dispositivo de destino pueden incluir otros componentes o disposiciones. Por ejemplo, el dispositivo de origen 102 puede recibir datos de video desde una fuente de video externa, tal como una cámara externa. Del mismo modo, el dispositivo de destino 116 puede interactuar con un dispositivo de visualización externo, en lugar de incluir un dispositivo de visualización integrado.

[0040] El sistema 100 como se muestra en la FIG. 1 es simplemente un ejemplo. En general, cualquier dispositivo de codificación y/o decodificación de video digital puede realizar técnicas para la inicialización de probabilidad para la codificación aritmética para la compresión de video. El dispositivo de origen 102 y el dispositivo de destino 116 son simplemente ejemplos de tales dispositivos de codificación en los que el dispositivo de origen 102 genera datos de video codificados para su transmisión al dispositivo de destino 116. Esta divulgación se refiere a un dispositivo de “codificación” como un dispositivo que realiza la codificación (codificación y/o decodificación) de datos. Así, 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 funcionar de una 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 admitir la transmisión de video unidireccional o bidireccional entre los dispositivos de video 102, 116, por ejemplo, para transmisión de video en continuo, reproducción de video, transmisión de video o telefonía de video.

[0041] En general, la fuente de video 104 representa una fuente de datos de video (es decir, datos de video sin procesar, sin codificar) y proporciona una serie secuencial de imágenes (también denominadas “cuadros”) de los datos de video al codificador de video 200, que codifica datos para las imágenes. La fuente de video 104 del dispositivo de origen 102 puede incluir un dispositivo de captura de video, tal como una cámara de

video, un archivo de video que contiene video sin procesar previamente capturado y/o una interfaz de alimentación de video para recibir video de un proveedor de contenido de video. Como alternativa adicional, la fuente de video 104 puede generar datos basados en gráficos de computadora 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 capturados, capturados previamente o generados 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 de origen 102 puede entonces emitir los datos de video codificados a través de la interfaz de salida 108 en un medio legible por computadora 110 para recepción y/o recuperación por parte de, por ejemplo, la interfaz de entrada 122 del dispositivo de destino 116.

[0042] La memoria 106 del dispositivo de origen 102 y la memoria 120 del dispositivo de destino 116 representan memorias de uso 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 decodificar y sin procesar del decodificador de video 300. Adicional o alternativamente, las memorias 106, 120 pueden almacenar instrucciones de *software* ejecutables mediante, por ejemplo, el codificador de video 200 y decodificador de video 300, respectivamente. Aunque 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, partes 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.

[0043] 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 de origen 102 al dispositivo de destino 116. En un ejemplo, el medio legible por computadora 110 representa un medio de comunicación para habilitar al dispositivo de origen 102 a transmitir 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 informática. 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 inalámbrico o cableado, como un espectro de radiofrecuencia (RF) o una o más líneas de transmisión física. El medio de comunicación puede formar parte de una red basada en paquetes, como una red de área local, una red de área amplia o una red global como Internet. El medio de comunicación puede incluir enrutadores, conmutadores, estaciones base o cualquier otro equipo que pueda ser útil para facilitar la comunicación desde el dispositivo de origen 102 al dispositivo de destino 116.

[0044] En algunos ejemplos, el medio legible por computadora 110 puede incluir el dispositivo de almacenamiento 112. El dispositivo de origen 102 puede emitir datos codificados desde la interfaz de salida 108 al dispositivo de almacenamiento 112. De manera similar, el dispositivo de destino 116 puede acceder a los 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 de acceso local, 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 almacenar datos de video codificados.

[0045] En algunos ejemplos, el medio legible por computadora 110 puede incluir el servidor de archivos 114 u otro dispositivo de almacenamiento intermedio que puede almacenar los datos de video codificados generados por el dispositivo de origen 102. El dispositivo de origen 102 puede enviar datos de video codificados al servidor de archivos 114 u otro dispositivo de almacenamiento intermedio que puede almacenar los datos de video codificados generados por el dispositivo de origen 102. El dispositivo de destino 116 puede acceder a los datos de video almacenados desde el servidor de archivos 114 mediante transmisión en continuo o descarga. El servidor de archivos 114 puede ser cualquier tipo de dispositivo de servidor capaz de almacenar datos de video codificados y transmitir esos 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 datos de video codificados desde el servidor de archivos 114 a través de cualquier conexión de datos estándar, incluida una conexión a Internet. Esto puede incluir un canal inalámbrico (por ejemplo, una conexión Wi-Fi),

una conexión por cable (por ejemplo, DSL, módem de cable, etc.) o una combinación de ambos que sea adecuada para acceder a los datos de video codificados almacenados en el servidor de archivos 114. El servidor de archivos 114 y la interfaz de entrada 122 pueden configurarse para funcionar de acuerdo con un protocolo de transmisión en continuo, un protocolo de transmisión de descarga o una combinación de estos.

[0046] 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 funcionan 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 se pueden configurar para transferir datos, como datos de video codificados, de acuerdo con un estándar de comunicación celular, como 4G, 4G-LTE (Evolución a largo plazo), LTE Advanced, 5G o similar. En algunos ejemplos donde la interfaz de salida 108 comprende un transmisor inalámbrico, la interfaz de salida 108 y la interfaz de entrada 122 se pueden configurar para transferir datos, como datos de video codificados, de acuerdo con otros estándares inalámbricos, como una especificación IEEE 802.11, una especificación IEEE 802.15 (por ejemplo, ZigBee™), un estándar Bluetooth™ o similar. En algunos ejemplos, el dispositivo de origen 102 y/o el dispositivo de destino 116 pueden incluir dispositivos respectivos de sistema en un chip (SoC). Por ejemplo, el dispositivo de origen 102 puede incluir un dispositivo 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 SoC para realizar la funcionalidad atribuida al decodificador de video 300 y/o la interfaz de entrada 122.

[0047] Las técnicas de esta divulgación se pueden aplicar a la codificación de video en apoyo de cualquiera de una variedad de aplicaciones multimedia, tales como transmisiones de televisión por aire, transmisiones de televisión por cable, transmisiones de televisión por satélite, transmisiones de video de transmisión en continuo por Internet, tales como transmisión en continuo dinámica adaptativa sobre 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.

[0048] La interfaz de entrada 122 del dispositivo de destino 116 recibe un flujo de bits de video codificado del medio legible por computadora 110 (por ejemplo, dispositivo de almacenamiento 112, servidor de archivos 114 o similar). El flujo de bits de video

codificado puede incluir información de señalización definida por el codificador de video 200, que también es utilizado por el decodificador de video 300, como elementos de sintaxis que tienen valores que describen características y/o el procesamiento de bloques de video u otras unidades codificadas (por ejemplo, segmentos, 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 una variedad de 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 emisor de luz orgánica (OLED) u otro tipo de dispositivo de visualización.

[0049] Aunque no se muestra en la FIG. 1, en algunos ejemplos, el codificador de video 200 y el decodificador de video 300 pueden estar integrados con un codificador de audio y/o decodificador de audio, y pueden incluir unidades MUX-DEMUX apropiadas, u otro *hardware* y/o *software*, para manejar flujos multiplexados que incluyen audio y video en un flujo de datos común. Si corresponde, las unidades MUX-DEMUX pueden cumplir con el protocolo multiplexor ITU H.223 u otros protocolos como el protocolo de datagramas de usuario (UDP).

[0050] El codificador de video 200 y el decodificador de video 300 pueden implementarse como cualquiera de una variedad de circuitos de codificador y/o decodificador adecuados, tales como uno o más microprocesadores, procesadores de señal digital (DSP), circuitos integrados específicos de aplicación (ASIC), matrices de puerta programables en campo (FPGA), lógica discreta, *software*, *hardware*, *firmware* o cualquier combinación de estos. Cuando las técnicas se implementan parcialmente en el *software*, un dispositivo puede almacenar instrucciones para el *software* en un medio legible por computadora adecuado, no transitorio y ejecutar las instrucciones en hardware utilizando uno o más procesadores para realizar las técnicas de esta descripción. Cada codificador de video 200 y decodificador de video 300 puede incluirse en uno o más codificadores o decodificadores, cualquiera de los cuales puede integrarse 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 inalámbrico, tal como un teléfono celular.

[0051] El codificador de video 200 y el decodificador de video 300 pueden funcionar de acuerdo con un estándar de codificación de video, tal como ITU-T H.265, también denominado Codificación de video de alta eficiencia (HEVC) o extensiones a estos, tales

como la vista múltiple y/o extensiones de codificación de video escalables. Alternativamente, el codificador de video 200 y el decodificador de video 300 pueden funcionar de acuerdo con otros estándares patentados o industriales, como el Modelo de Prueba de Exploración Conjunta (JEM) o ITU-T H.266, también conocido como Codificación de Video Versátil (VVC). Un borrador reciente del estándar VVC se describe en Bross, et al. “*Versatile Video Coding (Draft 4)*,” Grupo Conjunto de Expertos de Video (JVET) de ITU-T SG 16 WP 3 e ISO/IEC JTC 1/SC 29/WG 11, 13^{er} Encuentro: Marrakech, MA, 9–18 de enero de 2019, JVET-M1001-v6 (en adelante “VVC Draft 4”). Sin embargo, las técnicas de esta divulgación no se limitan a ningún estándar de codificación particular.

[0052] 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 a procesar (por ejemplo, codificados, decodificados o utilizados 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 rojos, verdes y azules (RGB) para muestras de una imagen, el codificador de video 200 y el decodificador de video 300 pueden codificar componentes de luminancia y crominancia, 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.

[0053] 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 la partición de imágenes en bloques. Por lo tanto, las referencias a la codificación de una imagen o un bloque generalmente deben entenderse

como valores de codificación para elementos de sintaxis que forman la imagen o el bloque.

[0054] La HEVC define varios bloques, que incluyen unidades de codificación (CU), unidades de predicción (PU) y unidades de transformación (TU). De acuerdo con HEVC, un codificador de video (como el codificador de video 200) divide una unidad de árbol de codificación (CTU) en CU según una estructura de árbol cuaternario. Es decir, el codificador de video divide las CTU y CU en cuatro cuadrados iguales y no superpuestos, 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 codificador de video puede dividir más PU y 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 se interpredicen incluyen información de interpredicción, como una indicación de modo intra.

[0055] Como otro ejemplo, el codificador de video 200 y el decodificador de video 300 pueden configurarse para funcionar de acuerdo con JEM o VVC. De acuerdo con JEM o VVC, un codificador de video (como el codificador de video 200) divide una imagen en una pluralidad de unidades de árbol de codificación (CTU). El codificador de video 200 puede dividir una CTU de acuerdo con una estructura de árbol, tal como una estructura de árbol binario cuaternario (QTBT) o una estructura de árbol de tipo múltiple (MTT). La estructura QTBT elimina los conceptos de múltiples tipos de partición, como la separación entre CU, PU y TU de HEVC. Una estructura QTBT incluye dos niveles: un primer nivel particionado de acuerdo con la partición de árbol cuaternario y un segundo nivel particionado 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).

[0056] En una estructura de partición MTT, los bloques pueden dividirse mediante 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 donde 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 a través del centro. Los tipos de partición en MTT (por ejemplo, QT, BT y TT) pueden ser simétricos o asimétricos.

[0057] 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, 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 componentes de crominancia respectivos).

[0058] El codificador de video 200 y el decodificador de video 300 pueden configurarse para usar la partición de árbol cuaternario por 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 pueden aplicarse a los codificadores de video configurados para usar la partición de árbol cuaternario u otros tipos de partición.

[0059] Esta divulgación puede usar “NxN” y “N por N” indistintamente para referirse a las dimensiones de muestra de un bloque (como un 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 ($x = 16$). Del mismo modo, una CU de NxN generalmente tiene N muestras en dirección vertical y N muestras en dirección horizontal, donde N representa un valor entero no negativo. Las muestras en una CU pueden organizarse en filas y columnas. Además, las CU no necesitan necesariamente tener el mismo número de muestras en la dirección horizontal que en la dirección vertical. Por ejemplo, las CU pueden comprender muestras de NxM, donde M no es necesariamente igual a N.

[0060] El codificador de video 200 codifica datos de video para CU que representan predicción y/o información residual, y otra información. La información de predicción indica cómo se debe predecir la CU para formar un bloque de predicción para la CU. La información residual generalmente representa diferencias muestra por muestra entre muestras de CU antes de la codificación y el bloque de predicción.

[0061] Para predecir una CU, el codificador de video 200 generalmente puede formar un bloque de predicción para la CU a través de la interpredicción o la intrapredicción. La interpredicción generalmente se refiere a la predicción de la CU a partir de datos de una imagen previamente codificada, mientras que la intrapredicción generalmente se refiere a la predicción de 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 usando 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 coincida estrechamente con 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 utilizando una suma de diferencia absoluta (SAD), suma de diferencias cuadradas (SSD), diferencia absoluta media (MAD), diferencias cuadradas medias (MSD) u otros cálculos de diferencia similares para determinar si un bloque de referencia coincide estrechamente con la CU actual. En algunos ejemplos, el codificador de video 200 puede predecir la CU actual mediante predicción unidireccional o predicción bidireccional.

[0062] Algunos ejemplos de JEM y 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 movimiento no traslacional, tal como acercamiento o alejamiento, rotación, movimiento en perspectiva u otros tipos de movimiento irregular.

[0063] Para realizar la intrapredicción, el codificador de video 200 puede seleccionar un modo de intrapredicción para generar el bloque de predicción. Algunos ejemplos de JEM y VVC proporcionan sesenta y siete modos de intrapredicción, incluidos varios modos direccionales, así como el modo plano y el modo DC. En general, el codificador de video 200 selecciona un modo de intrapredicción que describe las muestras vecinas a un bloque actual (por ejemplo, un bloque de una CU) desde el cual predecir muestras del bloque actual. Tales muestras generalmente pueden estar arriba, arriba y a la izquierda, o a la izquierda del bloque actual en la misma imagen que el bloque actual, suponiendo que el codificador de video 200 codifique CTU y CU en orden de escaneo de trama (de izquierda a derecha, de arriba a abajo).

[0064] 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 representan 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 utilizando 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.

[0065] Después de la predicción, tal como la intrapredicción o la interpredicción de un

bloque, el codificador de video 200 puede calcular datos residuales para el bloque. Los datos residuales, como un bloque residual, representan diferencias muestra por muestra entre el bloque y un bloque de predicción para el bloque, formado mediante el 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 discreta de coseno (DCT), una transformada entera, una transformada ondícula o una transformada conceptualmente similar a los datos de video residual. 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 Karhunen-Loeve (KLT) o similar. El codificador de video 200 produce coeficientes de transformación después de la aplicación de una o más transformadas.

[0066] Como se indicó anteriormente, después de cualquier transformada para producir coeficientes de transformación, el codificador de video 200 puede realizar la cuantificación de los coeficientes de transformación. La cuantificación generalmente se refiere a un proceso en el que los coeficientes de transformación se cuantifican para posiblemente reducir la cantidad de datos utilizados para representar los coeficientes, 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. 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 a nivel de bit del valor a cuantificar.

[0067] Después de la cuantificación, el codificador de video 200 puede escanear los coeficientes de transformación, produciendo un vector unidimensional a partir de la matriz bidimensional que incluye los coeficientes de transformación cuantificados. El escaneo puede estar diseñado para colocar coeficientes de energía más alta (y, por lo tanto, de frecuencia más baja) en la parte delantera del vector y para colocar coeficientes de transformación 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 escaneo predefinido para escanear los coeficientes de transformación cuantificados para producir un vector serializado, y luego codificar por entropía los coeficientes de transformación cuantificados del vector. En otros ejemplos, el codificador

de video 200 puede realizar un escaneo adaptativo. Después de escanear los coeficientes de transformación 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 uso por el decodificador de video 300 en la decodificación de los datos de video.

[0068] 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 relacionarse, por ejemplo, si los valores vecinos del símbolo tienen valor cero o no. La determinación de probabilidad puede basarse en un contexto asignado al símbolo.

[0069] 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, para el decodificador de video 300, por ejemplo, en un encabezado de imagen, un encabezado de bloque, un encabezado de segmento u otros datos de sintaxis, 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 también puede decodificar dichos datos de sintaxis para determinar cómo decodificar los datos de video correspondientes.

[0070] 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) y predicción y/o información residual para los bloques. Finalmente, el decodificador de video 300 puede recibir el flujo de bits y decodificar los datos de video codificados.

[0071] La siguiente sección describirá la codificación aritmética binaria (BAC) y las técnicas CABAC con más detalle. BAC, en general, es un procedimiento recursivo de subdivisión de intervalos. BAC se utiliza para codificar bins en el proceso CABAC en los estándares de codificación de video H.264/AVC y H.265/HEVC. La salida del codificador BAC es un flujo binario que representa un valor o puntero a una probabilidad dentro de un intervalo de probabilidad codificado final. El intervalo de probabilidad se especifica mediante un rango (“rango”) y un valor final más bajo (“bajo”). El rango es la extensión del intervalo de probabilidad. Bajo es el límite inferior del intervalo de codificación/probabilidad.

[0072] La aplicación de la codificación aritmética a la codificación de video se describe

en D. Marpe, H. Schwarz, y T. Wiegand “Context-Based Adaptive Binary Arithmetic Coding in the H.264/AVC Video Compression Standard,” IEEE Trans. Circuits and Systems for Video Technology, vol. 13, no. 7, julio de 2003. La CABAC implica tres funciones principales, a saber, binarización, modelado de contexto y codificación aritmética. La binarización se refiere a la función de mapear elementos de sintaxis a símbolos binarios (o “bins”). Los símbolos binarios también pueden denominarse “cadenas bin”. El modelado de contexto se refiere a la función de estimar la probabilidad de los diferentes bins. La codificación aritmética se refiere a la función posterior de comprimir los bins en bits, en base a la probabilidad estimada. Varios dispositivos y/o módulos de estos, como un codificador aritmético binario, pueden realizar la función de codificación aritmética.

[0073] Se utilizan varios procesos de binarización diferentes en HEVC, que incluyen unario (U), unario truncado (TU), Exp-Golomb de orden k (EGk) y longitud fija (FL). Los detalles de varios procesos de binarización se describen en V. Sze y M. Budagavi, “High throughput CABAC entropy coding in HEVC,” IEEE Transactions on Circuits and Systems for Video Technology (TCSVT), vol. 22, nro. 12, págs. 1778–1791, diciembre de 2012.

[0074] Cada contexto (es decir, modelo de probabilidad) en CABAC está representado por un estado. Cada estado (σ) representa implícitamente una probabilidad (p_σ) de que un símbolo particular (por ejemplo, un bin) sea el símbolo menos probable (LPS). Un símbolo puede ser un LPS o un símbolo más probable (MPS). Los símbolos son binarios, y como tales, el MPS y el LPS pueden ser 0 o 1. La probabilidad se estima para el contexto correspondiente y se usa (implícitamente) para codificar por entropía el símbolo mediante el codificador aritmético.

[0075] El proceso de BAC es manejado por una máquina de estado que cambia sus valores internos “rango” y “bajo” según el contexto para codificar y el valor del bin que se codifica. Según el estado de un contexto (es decir, su probabilidad), el rango se divide en $rangeMPS_\sigma$ (rango del símbolo más probable en el estado $_\sigma$) y $rangeLPS_\sigma$ (rango del símbolo menos probable en el estado $_\sigma$). En teoría, el valor $rangeLPS_\sigma$ de un estado de probabilidad $_\sigma$ se deriva de una multiplicación:

$$rangeLPS_\sigma = range \times p_\sigma,$$

donde p_σ es la probabilidad de seleccionar el LPS. Por supuesto, la probabilidad de MPS es $1-p_\sigma$. De manera equivalente, $rangeMPS_\sigma$ es igual a $range$ menos $rangeLPS_\sigma$. BAC actualiza iterativamente el rango según el estado del bin de contexto a codificar, el rango actual y el valor del bin que se codifica (es decir, es el bin igual al LPS o al MPS).

[0076] Las FIG. 2A y 2B muestran ejemplos de este proceso en bin n . En el ejemplo 201 de la FIG. 2A, el rango en bin n incluye RangeMPS y RangeLPS dado por la probabilidad de que el LPS (p_σ) tenga un cierto estado de contexto (σ). El ejemplo 201 muestra la actualización del rango en bin $n + 1$ cuando el valor de bin n es igual al MPS. En este ejemplo, el valor bajo permanece igual, pero el valor del rango en bin $n + 1$ se reduce al valor de RangeMPS en bin n . El ejemplo 203 de la FIG. 2B muestra la actualización del rango en bin $n + 1$ cuando el valor de bin n no es igual al MPS (es decir, igual al LPS). En este ejemplo, el valor bajo se mueve al valor de rango inferior de RangeLPS en bin n . Además, el valor del rango en bin $n + 1$ se reduce al valor de RangeLPS en bin n .

[0077] En HEVC, el rango se expresa con 9 bits y el valor bajo con 10 bits. Hay un proceso de renormalización para mantener el rango y los valores bajos con suficiente precisión. La renormalización ocurre siempre que el rango sea menor que 256. Por lo tanto, el rango siempre es igual o mayor que 256 después de la renormalización. Según los valores de rango y bajos, se emite la BAC al flujo de bits, un “0”, o un “1”, o actualiza una variable interna (llamada BO: bits pendientes) para mantener para salidas futuras. La FIG. 3 muestra ejemplos de emisión de BAC según el rango. Por ejemplo, se emite un ‘1’ al flujo de bits cuando el rango y el valor bajo están por encima de cierto umbral (por ejemplo, 512). Se emite un “0” al flujo de bits cuando el rango y el valor bajo están por debajo de cierto umbral (por ejemplo, 512). No se emite nada al flujo de bits cuando el rango y valor más bajo se encuentran entre ciertos umbrales. En cambio, el valor BO se incrementa y se codifica el siguiente bin.

[0078] Las técnicas descritas en esta divulgación pueden realizarse, por ejemplo, dentro de un codificador de video, decodificador de video o codificador-decodificador de video combinado (CODEC). En particular, tales técnicas pueden realizarse en una unidad de codificación de entropía de un codificador de video y/o una unidad de decodificación de entropía de un decodificador de video. Las técnicas se pueden realizar, por ejemplo, dentro de un proceso CABAC, que se puede configurar para admitir la codificación de video, como la codificación de video de acuerdo con los aspectos de las unidades de codificación y decodificación de entropía estándar de HEVC puede aplicar procesos de codificación de manera recíproca o inversa, por ejemplo, para codificar o decodificar cualquiera de una variedad de datos de video, tales como coeficientes de transformación cuantificados asociados con datos de video residuales, información de vectores de movimiento, elementos de sintaxis y otros tipos de información que pueden ser útiles en un proceso de codificación de video y/o decodificación de video.

[0079] 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 usando CABAC de una manera sustancialmente similar, aunque recíproca, al proceso de codificación de CABAC del codificador de video 200. Los elementos de sintaxis pueden definir información de partición de una imagen en CTU, y la partición de cada CTU de acuerdo con una estructura de partición correspondiente, como una estructura QTBT, para definir las CU de la CTU. Los elementos de sintaxis pueden definir aún más la predicción y la información residual para bloques (por ejemplo, CU) de datos de video.

[0080] La información residual puede estar representada, por ejemplo, por coeficientes de transformación cuantificados. El decodificador de video 300 puede cuantificar y transformar inversamente los coeficientes de transformación cuantificados de un bloque para reproducir un bloque residual para el bloque. El decodificador de video 300 usa un modo de predicción señalado (intrapredicción o interpredicción) e información de predicción relacionada (por ejemplo, información de movimiento para interpredicción) para formar un bloque de predicción para el bloque. El decodificador de video 300 puede entonces combinar 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.

[0081] De acuerdo con las técnicas de esta divulgación, el codificador de video 200 y/o el decodificador de video 300 pueden obtener un índice de pendiente predefinido (por ejemplo, Slopeldx) y un índice de desplazamiento predefinido (por ejemplo, OffsetIdx) para un contexto de una pluralidad de contextos utilizados en un proceso de codificación de entropía adaptable al contexto para codificar por entropía un valor para un elemento de sintaxis en un segmento de los datos de video; determinar, basándose en el índice de pendiente predefinido y el índice de desplazamiento predefinido, un estado de probabilidad inicial del contexto para el segmento de los datos de video en el dominio lineal; y codificar por entropía, basado en el estado de probabilidad inicial del contexto, un bin del valor para el elemento de sintaxis.

[0082] Esta divulgación generalmente puede referirse a “señalizar” cierta información, tal como elementos de sintaxis. El término “señalización” puede referirse generalmente a la comunicación de valores de elementos de sintaxis y/u otros datos utilizados 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 indicó anteriormente, el dispositivo de origen 102 puede transportar el flujo de bits al dispositivo de destino 116 sustancialmente en tiempo real, o no en tiempo real, tal como podría ocurrir al almacenar elementos de sintaxis en el dispositivo de almacenamiento 112 para su posterior recuperación por parte del dispositivo de destino 116.

[0083] Las técnicas de esta divulgación son adecuadas para los codificadores aritméticos binarios donde el estado de probabilidad representa la probabilidad real en el dominio lineal, tal como el codificador aritmético binario adaptativo al contexto (CABAC) adoptado en la codificación de video versátil (VVC).

[0084] En la codificación aritmética binaria, una secuencia de video se convierte primero en elementos de datos (o *elementos de sintaxis*) con la redundancia espacio-temporal eliminada; los elementos de sintaxis se convierten entonces sin pérdida en representaciones binarias (o *flujos de bits*) mediante codificación de entropía. Los estándares de codificación de video recientes (por ejemplo, HEVC y VVC) pueden utilizar codificación aritmética binaria para realizar la codificación de entropía.

[0085] El proceso de codificación aritmética binaria tiene tres etapas principales, es decir, binarización, estimación de probabilidad adaptativa y codificación aritmética. En la etapa de binarización, cada elemento de sintaxis no binario a codificar se convierte en una cadena de símbolos de datos binarios (o *bins*).

[0086] En la etapa de estimación de probabilidad adaptativa, cada bin, sin importar si es un elemento de sintaxis binario o un elemento de la cadena binaria convertido de un elemento de sintaxis no binario, tiene una estimación de su distribución de probabilidad (es decir, la probabilidad de ser 0 o 1). Las distribuciones pueden clasificarse en dos categorías: (1) distribución estacionaria y uniforme (es decir, siempre $p = 0,5$) y (2) distribución variable o no uniforme en el tiempo. Al bin con distribución de categoría (2) se le puede asignar un modelo de probabilidad (o *modelo de contexto*) que rastrea la distribución en tiempo real de ese bin en base a sus valores de bin anteriores y otras estadísticas de contexto.

[0087] En la etapa de codificación aritmética, el bin con distribución de categoría (1) puede codificarse en el modo de derivación, un modo de baja complejidad y altamente paralelizable; el bin con distribución de categoría (2) se codifica en modo regular, donde se utiliza el valor del bin y su probabilidad estimada por el modelo de contexto asociado.

[0088] Cuando se usa en codificadores aritméticos binarios para codificación de video, la probabilidad, teóricamente real y con un rango de 0 a 1, se digitaliza y, por lo tanto,

generalmente se conoce como estado de probabilidad. Por ejemplo, en HEVC, la probabilidad tiene una precisión de 7 bits, correspondiente a 128 estados de probabilidad. La FIG. 9 muestra el mapeo entre probabilidad y estado de probabilidad. Como se puede ver, el estado de probabilidad en HEVC representa la probabilidad real en el dominio logarítmico. Para otro ejemplo, en VVC, la estimación de probabilidad de un determinado bin es el promedio de las dos probabilidades rastreadas en el modelo de contexto asociado del bin y actualizadas a una tasa rápida y una tasa lenta, respectivamente. La que se actualiza a una velocidad rápida tiene una precisión de 10 bits, correspondiente a 1024 estados de probabilidad; la otra, actualizada a una velocidad lenta, tiene una precisión de 14 bits, correspondiente a 16.384 estados de probabilidad. A diferencia de HEVC, VVC adoptó el mapeo lineal entre el estado de probabilidad y la probabilidad (ver Figuras 10A y 10B).

[0089] En HEVC y VVC, un flujo de bits de video comprende o consiste en múltiples unidades decodificables independientemente (por ejemplo, segmentos), lo que implica que, al comienzo de dicha unidad, los estados de probabilidad de todos los modelos de contexto deben restablecerse a algunos valores predefinidos. Típicamente, sin ningún conocimiento previo de la naturaleza estadística de la fuente, cada modelo de contexto debe asumir la distribución uniforme ($p = 0,5$). Sin embargo, para unir la fase de aprendizaje de la estimación de probabilidad adaptativa y permitir una preadaptación en diferentes condiciones de codificación, se descubre que es beneficioso proporcionar un estado de probabilidad inicial más apropiado (*proceso de inicialización*) que el estado equiprobable para cada modelo de probabilidad.

[0090] La CABAC en HEVC tiene un proceso de inicialización dependiente del parámetro de cuantificación (QP) invocado al comienzo de cada segmento. Dado el valor inicial de luma QP para el segmento, SliceQP_Y , el estado de probabilidad inicial de un cierto modelo de contexto, denotado como InitProbState , es generado por Ec. (1) a (3),

$$m = \text{Slopeldx} * 5 - 45 \quad (1)$$

$$n = (\text{OffsetIdx} \ll 3) - 16 \quad (2)$$

$$\text{InitProbState} = \text{Clip3}(1, 127, ((m * \text{SliceQP}_Y) \gg 4) + n) \quad (3)$$

donde Slopeldx y OffsetIdx (ambos enteros que van de 0 a 15 inclusive) son los *parámetros de inicialización* predefinidos y almacenados para cada modelo de contexto. La Ec. (3) significa que InitProbState está modelado por una función lineal de SliceQP_Y , con la pendiente aproximadamente $m \gg 4$ y la intersección n en $\text{SliceQP}_Y = 0$. El mapeo de Slopeldx a pendiente y de OffsetIdx a intersección se puede encontrar en las Tablas

1 y 2, respectivamente.

Tabla 1 - Mapeo de Slopeldx a pendiente

Slopeldx	0	1	2	3	4	5	6	7
pendiente	-2,81	-2,50	-2,19	-1,88	-1,56	-1,25	-0,94	-0,63
Slopeldx	8	9	10	11	12	13	14	15
pendiente	-0,31	0	0,31	0,63	0,94	1,25	1,56	1,88

Tabla 2 - Mapeo de Offsetldx a intersección

Offsetldx	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
intersección	-16	-8	0	8	16	24	32	40	48	56	64	72	80	88	96	104

En otras palabras, un modelo de contexto puede no almacenar directamente el estado de probabilidad inicial; en su lugar, almacena dos parámetros de inicialización que determinan conjuntamente una función lineal que, al comienzo de cada segmento, usa SliceQP_Y como argumento para derivar el estado de probabilidad.

[0091] Slopeldx y Offsetldx, ambos con precisión de 4 bits, se empaquetan en un único *valor de inicialización* de 8 bits, de los cuales el cuarteto alto y bajo es Slopeldx y Offsetldx, respectivamente.

[0092] La CABAC de la VVC actual usa básicamente la misma forma de derivar InitProbState que en HEVC, excepto que en la Ec. (3) el recorte es de 0 a 127. Sin embargo, después de derivar InitProbState, que representa la probabilidad en el dominio logarítmico (ver FIG. 9), el proceso de inicialización en VVC necesita un paso más para convertir InitProbState en un estado de probabilidad que represente la probabilidad en el dominio lineal (ver FIG. 10A y 10B), para ser utilizado en el motor de codificación aritmética de VVC. La conversión (o mapeo), mostrada en las FIG. 11A y 11B, se implementa utilizando una tabla de búsqueda (LUT) (ver Tabla 3), como se muestra a continuación:

1. Usar InitProbState como el índice de búsqueda para encontrar el valor de probability_state correspondiente en la LUT.
2. - El estado de probabilidad en menor precisión (10 bits), denotado como ProbabilityStateL, se deriva en la Ec. (4)

$$\text{ProbabilityStateL} = \text{probability_state} \gg 5 \quad (4)$$

- El estado de probabilidad en mayor precisión (14 bits), denotado como ProbabilityStateH, se deriva en la Ec. (5)

$$\text{ProbabilityStateH} = \text{probability_state} \gg 1 \quad (5)$$

Tabla 3 - Mapeo de InitProbState a probability_state

InitProbState	0	1	2	3	4	5	6	7
probability_state	614	647	681	718	756	797	839	884
InitProbState	8	9	10	11	12	13	14	15
probability_state	932	982	1034	1089	1148	1209	1274	1342
InitProbState	16	17	18	19	20	21	22	23
probability_state	1414	1490	1569	1653	1742	1835	1933	2037
InitProbState	24	25	26	27	28	29	30	31
probability_state	2146	2261	2382	2509	2643	2785	2934	3091
InitProbState	32	33	34	35	36	37	38	39
probability_state	3256	3430	3614	3807	4011	4225	4452	4690
InitProbState	40	41	42	43	44	45	46	47
probability_state	4941	5205	5483	5777	6086	6412	6755	7116
InitProbState	48	49	50	51	52	53	54	55
probability_state	7497	7898	8320	8766	9235	9729	10249	10798
InitProbState	56	57	58	59	60	61	62	63
probability_state	11375	11984	12625	13300	14012	14762	15551	16384
InitProbState	64	65	66	67	68	69	70	71
probability_state	16384	17216	18005	18755	19467	20142	20783	21392
InitProbState	72	73	74	75	76	77	78	79
probability_state	21969	22518	23038	23532	24001	24447	24869	25270
InitProbState	80	81	82	83	84	85	86	87
probability_state	25651	26012	26355	26681	26990	27284	27562	27826
InitProbState	88	89	90	91	92	93	94	95
probability_state	28077	28315	28542	28756	28960	29153	29337	29511
InitProbState	96	97	98	99	100	101	102	103
probability_state	29676	29833	29982	30124	30258	30385	30506	30621
InitProbState	104	105	106	107	108	109	110	111
probability_state	30730	30834	30932	31025	31114	31198	31277	31353
InitProbState	112	113	114	115	116	117	118	119
probability_state	31425	31493	31558	31619	31678	31733	31785	31835
InitProbState	120	121	122	123	124	125	126	127
probability_state	31883	31928	31970	32011	32049	32086	32120	32153

[0093] Como se discutió anteriormente, la CABAC de la VVC actual adopta los estados de probabilidad que presentan las probabilidades en el dominio lineal (ver Figuras 10A y 10B), pero aún usa el proceso de inicialización heredado en HEVC, del cual los estados de probabilidad inicial derivados representan las probabilidades en el dominio logarítmico. Para unir la salida del proceso de inicialización y la entrada del motor de codificación aritmética, se utiliza una LUT de 128 entradas para el mapeo.

[0094] De acuerdo con una o más técnicas de esta divulgación, el proceso de inicialización puede modificarse para VVC (por ejemplo, estas técnicas pueden mejorar este proceso de búsqueda de tabla modificando adecuadamente el proceso de inicialización en VVC). Se proponen dos métodos para la modificación. En un método, el estado de probabilidad de salida InitProbState representa la probabilidad en el dominio lineal. Al hacer esto, el proceso de mapeo se reduce a operaciones de

desplazamiento, solo para fines de ajuste de precisión. En el otro método, el estado de probabilidad de salida `InitProbState` representa la probabilidad en el dominio cuadrático, y el mapeo se realiza utilizando ecuaciones, no LUT.

[0095] Aumentar el rango de pendiente e intersección. Para encontrar los parámetros de inicialización apropiados para un modelo de contexto dado, es necesario adquirir el conocimiento previo de la distribución de probabilidad de los bins que utilizan ese modelo de contexto, de alguna manera, como el análisis estadístico de la fuente. La FIG. 12 muestra un ejemplo para el bin del elemento de sintaxis `SaoMergeFlag` utilizado en el segmento I.

[0096] En HEVC y la VVC actual, el conocimiento previo de probabilidad se convierte en estado de probabilidad en el dominio logarítmico (la FIG. 13A se convierte de la FIG. 12), y los parámetros de inicialización óptimos, mediante los cuales la función lineal determinada se ajusta mejor a la correlación entre el estado de probabilidad y `SliceQPY`, se encuentran mediante la búsqueda de fuerza bruta u otras formas más inteligentes. Para el mismo ejemplo, `Slopeldx` y `Offsetldx` son 13 y 6, respectivamente (pendiente aproximadamente 1,25 e intersección 32), y la función lineal de mejor ajuste correspondiente se representa en la FIG. 13B.

[0097] De acuerdo con las técnicas de esta divulgación, el estado de probabilidad inicial está en el dominio lineal, para ser armonizado con el motor de codificación aritmética. Para el mismo ejemplo, el conocimiento previo representado por el estado de probabilidad en el dominio lineal se muestra en la FIG. 14A, que es simplemente una versión a escala de la FIG. 12. Se debe tener en cuenta que la precisión de 7 bits se usa tanto para los estados de probabilidad lineal como logarítmico, por la simplicidad de la aclaración. Sin embargo, en otros ejemplos, se pueden usar otras precisiones de bits.

[0098] Una comparación de la FIG. 13A y la FIG. 14A muestra que el estado de probabilidad cambia mucho más suavemente junto con `SliceQPY` en el dominio logarítmico que en el dominio lineal, lo que significa que una línea plana que se ajusta bien a los estados de probabilidad en el dominio logarítmico puede volverse bastante empinada en el dominio lineal. Con respecto al estado de probabilidad mostrado en la FIG. 14A, sin cambiar el proceso de inicialización definido en HEVC y la VVC actual (ver las Ecuaciones (1) a (3)), los mejores parámetros de inicialización que se pueden encontrar mediante la búsqueda de fuerza bruta son `Slopeldx` igual a 15 y `Offsetldx` igual a 4, con el cual se alcanza la pendiente válida más grande 1,88, como se muestra en la FIG. 14B. Sin embargo, es posible que la línea aún no sea lo suficientemente empinada para lograr un buen ajuste. Con mayores rangos de búsqueda de pendiente e

intersección, se puede encontrar un mejor ajuste, como se muestra en la FIG. 14C, donde la pendiente es 3,25 y la intersección es -23, ambos exceden los límites que HEVC y la VVC actual permiten.

[0099] Obsérvese que el ejemplo (Figuras 12 a 14C) es típico más que excepcional: un buen número de bins tiene este tipo de distribuciones de probabilidad. Por lo tanto, los rangos de pendiente y desplazamiento deben aumentarse para usar un modelo lineal que se ajuste a la correlación de SliceQP_Y y el estado de probabilidad que ahora está en el dominio lineal.

[0100] En un ejemplo, los números de posibles valores de pendiente e intersección no cambian (es decir, Slopeldx y Offsetldx todavía varían de 0 a 15), pero el tamaño del paso entre ellos aumenta, lo que significa que la constante *a* en la Ec. (6) es mayor que 5 en la Ec. (1) y *b* en la Ec. (7) es mayor que 3 en la Ec. (2), respectivamente.

$$m = \text{Slopeldx} * a - c \quad (6)$$

$$n = (\text{Offsetldx} << b) - d \quad (7)$$

[0101] En otro ejemplo, se incrementan los números de posibles valores de pendiente e intersección. Por ejemplo, Slopeldx puede variar de 0 a 31, lo que lleva a 32 valores de pendiente posibles. Los dos ejemplos anteriores pueden usarse conjuntamente o pueden usarse independientemente.

[0102] Definir la intersección en SliceQP_Y no igual a 0. Además de satisfacer el estado de probabilidad lineal con mayores rangos de pendiente y desplazamiento, definir la intersección en SliceQP_Y no igual a 0 es una forma alternativa o adicional. Por ejemplo, la FIG. 15A muestra la distribución de probabilidad de SplitFlag utilizada en el segmento B. En el dominio logarítmico, como se muestra en la FIG. 15B, el estado de probabilidad con respecto a SliceQP_Y se puede modelar bien mediante una función lineal con pendiente igual a -0,63 e intersección igual a 88 (es decir, Slopeldx 7 y Offsetldx 13 como se define en HEVC y la VVC actual). Sin embargo, en el dominio lineal, el estado de probabilidad no se puede modelar bien por las combinaciones válidas de pendiente/intersección definidas en HEVC y la VVC actual. La FIG. 15C muestra el ajuste más cercano encontrado por la búsqueda de fuerza bruta, donde la pendiente es igual a -0,94 y la intersección es igual a 104 (es decir, Slopeldx 6 y Offsetldx 15). Como se muestra en la FIG. 15D, otra pendiente válida -1,88 (Slopeldx 3) logra un mejor ajuste, pero la proyección al eje y (es decir, la intersección en SliceQP_Y igual a 0) es 130, mucho más allá del límite superior de 104.

[0103] Como se muestra arriba en la Ec. (3), en algunos ejemplos, el estado de probabilidad inicial puede determinarse en parte multiplicando la variable *m* por

SliceQP_Y. En el ejemplo de la Ec. (3), el modelo lineal del estado de probabilidad se define en el punto donde SliceQP_Y es cero.

[0104] De acuerdo con una o más técnicas de esta divulgación, un codificador de video puede utilizar un desplazamiento tal que el modelo lineal del estado de probabilidad se defina en un valor de SliceQP_Y distinto de cero. El punto en el modelo lineal en el que se define el estado de probabilidad puede denominarse un punto de anclaje del parámetro de cuantificación (QP_{anchor}). Por ejemplo, el codificador de video puede restar el desplazamiento de SliceQP_Y y multiplicar el resultado por la variable m. En un ejemplo, esto se puede lograr al modificar la Ec. (3) en la Ec. (8) como se muestra a continuación.

$$\text{InitProbState} = \text{Clip3}(1, 127, ((m * (\text{SliceQP}_Y - \text{QP}_{\text{anchor}})) >> 4) + n) \quad (8)$$

donde QP_{anchor} es el valor de QP en el que se define la intersección. De esta manera, la intersección no puede exceder el rango válido de los estados de probabilidad (1 a 127 en este ejemplo), y se elimina la consideración de una gran proyección en el eje y, especialmente cuando la pendiente es empinada. En la FIG. 15D, Slopeldx es igual a 3, QP_{anchor} es 32 y la intersección en QP_{anchor} es 72. Otros valores para QP_{anchor} incluyen, entre otros, 8, 16, 64, 128, etc.

[0105] Usar representación alternativa para la función lineal. En los ejemplos discutidos anteriormente, la función lineal utilizada para modelar el estado de probabilidad con SliceQP_Y se determina por un par de pendiente e intersección, que es lo mismo que HEVC y la VVC actual. Esta sección propone representaciones alternativas para funciones lineales, donde una función de línea está determinada por dos intersecciones en dos QP diferentes. En comparación con la representación de pendiente/intersección, esta representación de doble intersección proporciona más flexibilidad en el ajuste, ya que permite muchos más valores de pendiente que no están en forma de secuencia aritmética.

[0106] Denotar las dos intersecciones y las dos QP correspondientes como int1, int2, QP1 y QP2, respectivamente. QP1 y QP2 pueden ser arbitrarios, siempre que estén dentro del rango válido definido en la VVC actual (0 a 63). Sin embargo, dado que InitProbState se calcula mediante la Ec. (9) o la Ec. (10) como a continuación (tener en cuenta que, por simplicidad de aclaración, aquí no se consideran las operaciones intermedias de recorte y desplazamiento),

$$\frac{\text{int1} - \text{int2}}{\text{QP1} - \text{QP2}} (\text{SliceQP}_Y - \text{QP1}) + \text{int1} \quad (9)$$

$$\frac{\text{int1} - \text{int2}}{\text{QP1} - \text{QP2}} (\text{SliceQP}_Y - \text{QP2}) + \text{int2} \quad (10)$$

la diferencia de QP1 y QP2 es mejor que sea una potencia entera de 2, de modo que las divisiones en la Ec. (9) y la Ec. (10) puedan ser reemplazadas por desplazamiento a la derecha.

[0107] La representación de doble intersección puede extenderse aún más a la representación de intersección múltiple que determina funciones lineales por partes. Uno de estos ejemplos es un bin de luma SigFlag, ver FIG. 16A-16C, donde se produce un punto de inflexión en QP alrededor de 30 o 31 y una línea recta determinada por un par de pendiente/intersección o intersecciones dobles no es lo suficientemente buena para el ajuste (ver FIG. 16A). En este caso, se propone la representación de triple intersección, que utiliza tres parámetros de inicialización: intersección en QP igual a 31 y otras dos intersecciones en QP más pequeñas y mayores que 31, respectivamente (ver Figura 16B). Para ajustar la distribución de probabilidad aún más precisamente, se puede usar una representación de intersección cuádruple, donde se usa un parámetro de inicialización adicional que representa la intersección en QP igual a 37 para capturar otro punto de inflexión menor (ver Figura 16C).

[0108] Mantener una precisión intermedia más alta. Los ejemplos anteriores introducen las modificaciones en las Ec. (1) a (3), y derivar InitProbState en el dominio lineal. InitProbState todavía tiene una precisión de 7 bits (0 a 127), debido al legado de HEVC, y debe dejarse desplazado 3 bits y 7 bits, para alinearse con las precisiones de ProbabilityStateL y ProbabilityStateH definidas en la VVC actual, como en las Ec. (11) y (12).

$$\text{ProbabilityStateL} = \text{InitProbState} \ll 3 \quad (11)$$

$$\text{ProbabilityStateH} = \text{InitProbState} \ll 7 \quad (12)$$

[0109] Téngase en cuenta que el paso de búsqueda de la tabla justo antes de las Ec. (4) y (5) se guarda en esta técnica.

[0110] La aplicación de desplazamiento a la derecha para derivar InitProbState (ver la Ecuación (3)) seguido de desplazamientos a la izquierda (ver las Ec. (11) y (12)) sin duda pierde más información que mover todos (o parte) de los desplazamientos a la derecha al último paso, porque este último conserva una mayor precisión intermedia. Los siguientes dos ejemplos mueven todos los desplazamientos a la derecha al último paso.

En un ejemplo:

$$\text{InitProbState} = \text{Clip3}(1, 2047, ((m * \text{SliceQP}_Y) + (n \ll 4))) \quad (13)$$

$$\text{ProbabilityStateL} = \text{InitProbState} \gg 1 \quad (14)$$

$$\text{ProbabilityStateH} = \text{InitProbState} \ll 3 \quad (15)$$

En otro ejemplo:

$$\text{InitProbState} = \text{Clip3}(0, 2047, ((m * \text{SliceQP}_Y) + (n \ll 4))) \quad (16)$$

$$\text{ProbabilityStateL} = \text{InitProbState} \gg 1 \quad (17)$$

$$\text{ProbabilityStateH} = (\text{InitProbState} \ll 3) + 4 \quad (18)$$

[0111] Relacionado con CE1: la simplificación de JVET-O0191 que utiliza 4 o 6 bits por valor de inicialización, JVET-O0946, Gotemburgo, Suecia, del 3 al 12 de julio de 2019 (en adelante, “JVET-O0946”) propone la siguiente técnica para la inicialización:

$$q = \text{Clip3}(9, 23, \text{SliceQP}_Y \gg 1) - 8$$

$$a = (\text{initValue} \gg 3) * 9$$

$$b = (\text{initValue} \& 7) * 9$$

$$\text{ProbabilityStateL} = 16 * a + (b - a) * q + 8$$

$$\text{ProbabilityStateH} = \text{ProbabilityStateL} \ll 4$$

[0112] Sin embargo, la técnica propuesta en JVET-O0946 puede presentar una o más desventajas. Por ejemplo, la técnica propuesta en JVET-O0946 puede perder algo de precisión intermedia.

[0113] De acuerdo con una o más técnicas de esta divulgación, un codificador de video puede realizar la inicialización de la siguiente manera:

$$q = \text{Clip3}(18, 46, \text{SliceQP}_Y) - 16$$

$$a = (\text{initValue} \gg 3) * 9$$

$$b = (\text{initValue} \& 7) * 9$$

$$\text{ProbabilityStateL} = 16 * a + ((b - a) * q \gg 1) + 8$$

$$\text{ProbabilityStateH} = \text{ProbabilityStateL} \ll 4$$

[0114] Como se puede ver arriba, al mover el desplazamiento a la derecha hasta el paso final (es decir, al calcular ProbabilityStateL), la técnica anterior permite una alta precisión intermedia.

[0115] Inicialización en el dominio cuadrático. Los ejemplos anteriores pueden ser para escenarios en los que la salida del proceso de inicialización representa el estado de probabilidad en el dominio lineal. Sin embargo, para algunas fuentes de entrada, puede ser deseable hacer que el estado de probabilidad represente la probabilidad en el dominio logarítmico, correspondiente a la representación de la entropía en el dominio lineal. Para lograr la representación en el dominio logarítmico, esta divulgación propone utilizar los estados de probabilidad en el dominio cuadrático. Las FIG. 17A y 17B muestran un ejemplo del mapeo desde el dominio cuadrático InitProbState a las probabilidades lineales utilizadas en el motor de codificación aritmética de VVC, que se parece al mapeo para InitProbState logarítmico (ver FIG. 11A-11B), pero puede

implementarse mediante multiplicaciones (se debe tener en cuenta que LUT que almacena todos los resultados de multiplicación posibles se puede generar de antemano para su uso si lo prefiere alguna implementación). A continuación, hay un ejemplo,

```
slope      = int(initld & 0x0F) - 8;
offset     = int(initld & 0xF0) - 128;
shp        = offset + ((slope * (qp - 32)) >> 1);
idx        = Clip3(-127, 127, shp);
val        = 128 - abs(idx);
p1         = (idx < 0 ? 32768 - val2 : val2);
ProbabilityStateL = p1 >> 1
ProbabilityStateH = p1 >> 5
```

donde initld es el valor de inicialización con el cuarteto alto y bajo que representa la pendiente y el desplazamiento, respectivamente, e idx significa el estado de probabilidad con precisión de 8 bits y centrado en 0.

[0116] Téngase en cuenta que el método propuesto en esta sección se puede combinar con el uso de mayor precisión intermedia como se introdujo anteriormente.

[0117] Inicialización para distribuciones de probabilidad extrema. Un gran número de bins puede tener distribuciones de probabilidad extrema, lo que significa que estos bins tienen una probabilidad muy alta de ser 1 o 0, independientemente de SliceQP_Y. Las FIG. 18A y 18B son gráficos que ilustran ejemplos de distribuciones de probabilidad extrema.

[0118] Para modelar tales distribuciones de probabilidad extrema, la pendiente puede tener que ser 0, y el estado de probabilidad inicial puede depender solo de n (ver las Ec. (2) y (7)). Como tal, el estado de probabilidad inicial puede no cambiar junto con el parámetro de cuantificación (QP). La fórmula que deriva n, como en la Ec. (2) o la Ec. (7), puede diseñarse de modo que las probabilidades que se aproximen a 0 y 1 puedan representarse de manera eficiente. Los siguientes ejemplos se basan en la representación de 7 bits (0 a 127) de estados de probabilidad y se pueden ampliar a otras precisiones.

[0119] Tomar la Ec. (2) como ejemplo. Con OffsetIdx que varía de 0 a 15, el valor máximo de n que se puede alcanzar es 104, que representa la probabilidad 0,8189 (es decir, 104/128), que está lejos de 1,0.

[0120] En “*Simplification of the initialization process for context variables*”, JVET-N0301, Geneva, CH, 19-27 de marzo de 2019, n se deriva con precisión de 10 bits, con OffsetIdx que varía de 0 a 7, como en la Ec. (19).

$$n = ((\text{OffsetIdx} * 5) \ll 5) + 32 \quad (19)$$

[0121] Su representación equivalente de 7 bits de n se muestra en la Ec. (20)

$$n = ((\text{OffsetIdx} * 5) \ll 2) + 4 \quad (20)$$

donde la probabilidad más cercana a 0 que puede alcanzar es 0,0313 (OffsetIdx igual a 0 y n igual a 4, $4/128 = 0,0313$) y la probabilidad más cercana a 1,0 que puede alcanzar es 0,9688 (OffsetIdx igual a 6 y n igual a 124, $124/128 = 0,9688$). Sin embargo, tales aproximaciones pueden no ser lo suficientemente precisas como para representar las distribuciones de probabilidad extrema.

[0122] De acuerdo con una o más técnicas de esta divulgación, un codificador de video puede utilizar una o más de las siguientes fórmulas para derivar un parámetro de inicialización (por ejemplo, n).

1. Si OffsetIdx es de 3 bits (de 0 a 7), n se puede derivar como se muestra en la Ec. (21).

$$n = (\text{OffsetIdx} * 18) + 1 \quad (21)$$

2. Si OffsetIdx es de 4 bits (de 0 a 15), n se puede derivar como se muestra en la Ec. (22).

$$n = (\text{OffsetIdx} * 9) + 1 \quad (22)$$

[0123] Como se puede ver, en cualquier caso, n puede alcanzar 1 y 127, lo que significa que se pueden alcanzar las probabilidades de 0,0078 (es decir, $1/128$) y 0,9922 (es decir, $127/128$). Dichas probabilidades pueden estar lo suficientemente cerca de 0 y 1, respectivamente. De esta manera, las técnicas de esta divulgación permiten que un codificador de video inicialice distribuciones de probabilidad extrema con mayor precisión, lo que puede reducir la velocidad de bits de los datos de video codificados.

[0124] En algunos ejemplos, m, n, y/o initState se pueden determinar de acuerdo con una de las siguientes ecuaciones:

(23)

$$m = \text{slopIdx} * 4 - 36$$

$$n = \text{OffsetIdx} * 9 + 1$$

$$\text{initState} = (m * (\text{qp} - 24) \gg 4) + n$$

(24)

$$m = \text{slopIdx} * 8 - 32$$

$$n = \text{OffsetIdx} * 18 + 1$$

$$\text{initState} = (m * (\text{qp} - 16) \gg 4) + n$$

(25)

$$m = \text{slopeldx} * 5 - 45$$
$$n = \text{OffsetIdx} * 8 + 7$$
$$\text{initState} = (m * (\text{qp} - 32) >> 4) + n$$

[0125] Las FIG. 4A y 4B son diagramas conceptuales que ilustran un ejemplo de estructura 130 de árbol binario cuaternario (QTBT), y una unidad de árbol de codificación correspondiente (CTU) 132. Las líneas continuas representan la división de árbol cuaternario y las líneas de puntos indican la división de árbol binario. En cada nodo de división (es decir, no hoja) del árbol binario, se señala un indicador 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 (como la división de 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 la división de 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 de transformada, para CU representadas por nodos hoja terminales de la estructura QTBT 130.

[0126] En general, la CTU 132 de la FIG. 4B 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 permitido de nodo hoja de árbol cuaternario), un tamaño máximo de árbol binario (MaxBTSIZE, que representa un tamaño máximo permitido de un nodo raíz de árbol binario), 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 del nodo hoja del árbol binario).

[0127] 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 dividirse de acuerdo con la partición de árbol cuaternario. Es decir, los

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 nodos tales como el nodo primario y los nodos secundarios que tienen líneas continuas para las ramas. Si los nodos del primer nivel no son mayores que el tamaño máximo permitido del nodo raíz del árbol binario (MaxBTSIZE), pueden dividirse aún más por los respectivos árboles binarios. La división del árbol binario de un nodo puede repetirse 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 los nodos que tienen líneas discontinuas para las ramas. El nodo hoja de árbol binario se conoce como una unidad de codificación (CU), que se utiliza para la predicción (por ejemplo, predicción intraimagen o interimagen) y transforma, sin ninguna partición adicional. Como se discutió anteriormente, las CU también pueden denominarse “bloques de video” o “bloques”.

[0128] En un ejemplo de la estructura de partición QTBT, el tamaño de la CTU se establece en 128x128 (muestras de luma y dos muestras de croma 64x64 correspondientes), el MinQTSIZE se establece en 16x16, el MaxBTSIZE se establece en 64x64, el MinBTSIZE (para ambos anchos y altura) se establece en 4, y MaxBTDepth se establece en 4. La partición de 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, el tamaño MinQTSIZE) a 128x128 (es decir, el tamaño de la CTU). Si el nodo hoja de árbol cuaternario es 128x128, no se dividirá más por parte del árbol binario, ya que el tamaño excede el tamaño MaxBT (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 del árbol binario tiene un ancho igual a MinBTSIZE (4, en este ejemplo), implica que no se permite más división horizontal. De manera similar, un nodo de árbol binario que tiene una altura igual a MinBTSIZE implica que no se permite más división vertical para ese nodo de árbol binario. Como se señaló anteriormente, los nodos hoja del árbol binario se denominan CU y se procesan más de acuerdo con la predicción y se transforman sin más partición.

[0129] La FIG. 5 es un diagrama de bloques que ilustra un codificador de video 200 de ejemplo que puede realizar las técnicas de esta divulgación. La FIG. 5 se proporciona

con fines de explicación y no debe considerarse que limita las técnicas como se ejemplifica y describe ampliamente 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 HEVC y el estándar de codificación de video H.266 en desarrollo. Sin embargo, las técnicas de esta descripción no se limitan a estos estándares de codificación de video, y son aplicables generalmente a la codificación y decodificación de video.

[0130] En el ejemplo de la FIG. 5, el codificador de video 200 incluye memoria de datos de video 230, unidad de selección de modo 202, unidad de generación residual 204, unidad de procesamiento de transformada 206, unidad de cuantificación 208, unidad de cuantificación inversa 210, unidad de procesamiento de transformada inversa 212, unidad de reconstrucción 214, unidad de filtro 216, búfer de imagen decodificada (DPB) 218 y unidad de codificación de entropía 220. Cualquiera o la totalidad de la memoria de datos de video 230, unidad de selección de modo 202, unidad de generación residual 204, unidad de procesamiento de transformada 206, unidad de cuantificación 208, unidad de cuantificación inversa 210, unidad de procesamiento de transformada inversa 212, unidad de reconstrucción 214, unidad de filtro 216, DPB 218 y unidad de codificación de 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.

[0131] La memoria de datos de video 230 puede almacenar datos de video para ser codificados 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 (FIG. 1). El DPB 218 puede actuar como una memoria de imagen 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 DPB 218 pueden estar formados por cualquiera de una variedad de dispositivos de memoria, como la memoria dinámica de acceso aleatorio (DRAM), que incluye 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 dispositivos de memoria separados. En varios ejemplos, la memoria de datos de video 230 puede estar en chip con otros componentes del codificador de video 200, como se ilustra, o fuera del chip en relación con esos componentes.

[0132] En esta divulgación, la referencia a la memoria de datos de video 230 no debe interpretarse como limitada a la memoria interna al 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 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 codificará). La memoria 106 de la FIG. 1 también puede proporcionar almacenamiento temporal de salidas de las diversas unidades del codificador de video 200.

[0133] Las diversas unidades de la FIG. 5 se ilustran para ayudar a comprender las funciones 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 circuitos que proporcionan una funcionalidad particular y están preestablecidos en las operaciones que se pueden realizar. Los circuitos programables se refieren a 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, recibir parámetros o emitir parámetros), pero los tipos de funciones que realizan los circuitos de función fija son generalmente inmutables. En algunos ejemplos, una o más unidades pueden ser bloques de circuitos distintos (función fija o programable), y en algunos ejemplos, la unidad o las unidades pueden ser circuitos integrados.

[0134] El codificador de video 200 puede incluir unidades lógicas aritméticas (ALU), unidades de funciones elementales (EFU), circuitos digitales, circuitos analógicos y/o núcleos programables, formados a partir de circuitos programables. En los ejemplos en los que las funciones del codificador de video 200 se realizan mediante el uso de un *software* ejecutado por los circuitos programables, la memoria 106 (FIG. 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.

[0135] 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 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 codificarán.

[0136] 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 predicciones de video de acuerdo con otros modos de predicción. Como ejemplos, la unidad de selección de modo 202 puede incluir una unidad 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 similar.

[0137] La unidad de selección de modo 202 generalmente coordina múltiples pases de codificación para probar combinaciones de parámetros de codificación y los 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.

[0138] 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 de la partición de una CTU de acuerdo con la estructura de árbol. Tal CU también se puede denominar generalmente como un “bloque de video” o “bloque”.

[0139] En general, la unidad de selección de modo 202 también controla sus componentes (por ejemplo, la unidad de estimación de movimiento 222, la unidad de compensación de movimiento 224 y la 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 DPB 218). En particular, la unidad de estimación de movimiento 222 puede calcular un valor representativo de cuán similar es un bloque de referencia potencial al bloque actual, por ejemplo, de acuerdo con la suma de la diferencia absoluta (SAD), la suma de las 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 usando 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 coincide más estrechamente con el bloque actual.

[0140] 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 solo 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 usando los vectores de movimiento. Por ejemplo, la unidad de compensación de movimiento 224 puede recuperar datos del bloque de referencia usando el vector de movimiento. Como otro ejemplo, si el vector de movimiento tiene una precisión de muestra fraccional, 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 los respectivos vectores de movimiento y combinar los datos recuperados, por ejemplo, el promedio de muestra por muestra o el promedio ponderado.

[0141] Como otro ejemplo, para la intrapredicción o la 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 modos direccionales, la unidad de intrapredicción 226 generalmente puede combinar matemáticamente valores de muestras vecinas y llenar 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.

[0142] 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 sin procesar y sin codificar 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 las diferencias entre los valores de muestra en el bloque residual para generar un bloque residual utilizando la modulación de código de pulso diferencial residual (RDPCM). En algunos ejemplos, la unidad de generación residual 204 puede formarse usando uno o más circuitos sustractores que realizan sustracción binaria.

[0143] En los ejemplos en los que la unidad de selección de modo 202 divide las CU en PU, cada PU puede asociarse con una unidad de predicción de luma y las unidades de predicción de croma correspondientes. El codificador de video 200 y el decodificador de video 300 pueden admitir PU que tienen varios tamaños. Como se indicó anteriormente, el tamaño de una CU puede referirse al tamaño del bloque de codificación luma de la CU y el tamaño de una PU puede referirse al tamaño de una unidad de predicción de luma de la PU. Suponiendo que el tamaño de una CU particular es $2N \times 2N$, el codificador de video 200 puede admitir tamaños de PU de $2N \times 2N$ o $N \times N$ para intrapredicción, y tamaños de PU simétricos de $2N \times 2N$, $2N \times N$, $N \times 2N$, $N \times N$ o similares para interpredicción. El codificador de video 200 y el decodificador de video 300 también pueden admitir partición asimétrica para tamaños de PU de $2N \times nU$, $2N \times nD$, $nL \times 2N$ y $nR \times 2N$ para interpredicción.

[0144] En los ejemplos donde la unidad de selección de modo no divide más una CU en PU, cada CU puede estar asociada con un bloque de codificación luma y los bloques de codificación croma correspondientes. Como se indicó anteriormente, el tamaño de una CU puede referirse al tamaño del bloque de codificación luma de la CU. El codificador de video 200 y el decodificador de video 300 pueden admitir tamaños de CU de $2N \times 2N$, $2N \times N$ o $N \times 2N$.

[0145] Para otras técnicas de codificación de video tales como una codificación en modo de copia intrabloque, una codificación en modo afín y una codificación en 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 codifica. En algunos ejemplos, como la codificación de modo de paleta, la unidad de selección de modo 202 puede no generar un bloque de predicción, y en su lugar generar elementos de sintaxis que indican la manera de reconstruir el bloque en base a una paleta seleccionada. En tales modos, la unidad de selección de modo 202 puede proporcionar estos elementos de sintaxis a la unidad de codificación de entropía 220 a codificar.

[0146] 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.

[0147] La unidad de procesamiento de transformada 206 aplica una o más transformadas al bloque residual para generar un bloque de coeficientes de transformación (denominado en este documento “bloque de coeficiente de transformación”). La unidad de procesamiento de transformada 206 puede aplicar diversas transformadas a un bloque residual para formar el bloque de coeficiente de transformación. Por ejemplo, la unidad de procesamiento de transformada 206 puede aplicar una transformada de coseno discreto (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 a 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.

[0148] La unidad de cuantificación 208 puede cuantificar los coeficientes de transformación en un bloque de coeficiente de transformación, para producir un bloque de coeficiente de transformación cuantificado. La unidad de cuantificación 208 puede cuantificar los coeficientes de transformación de un bloque de coeficiente de transformación de acuerdo con un valor de parámetro de cuantificación (QP) asociado con el bloque actual. El codificador de video 200 (por ejemplo, a través de la unidad de selección de modo 202) puede ajustar el grado de cuantificación aplicado a los bloques de coeficientes de transformación asociados con el bloque actual ajustando el valor de QP asociado con la CU. La cuantificación puede introducir pérdida de información y, por

lo tanto, los coeficientes de transformación cuantificados pueden tener una precisión menor que los coeficientes de transformación originales producidos por la unidad de procesamiento de transformada 206.

[0149] 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 coeficiente de transformación cuantificado, respectivamente, para reconstruir un bloque residual del bloque de coeficiente de transformación. La unidad de reconstrucción 214 puede producir un bloque reconstruido correspondiente al bloque actual (aunque potencialmente con cierto grado de distorsión) basado en el 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 muestras correspondientes del bloque de predicción generado por la unidad de selección de modo 202 para producir el bloque reconstruido.

[0150] 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 las interferencias de bloqueo a lo largo de los bordes de las CU. Las operaciones de la unidad de filtro 216 pueden omitirse, en algunos ejemplos.

[0151] El codificador de video 200 almacena bloques reconstruidos en el DPB 218. Por ejemplo, en ejemplos donde no se necesitan operaciones de la unidad de filtro 216, la unidad de reconstrucción 214 puede almacenar bloques reconstruidos en el DPB 218. En ejemplos donde se necesitan operaciones de la unidad de filtro 216, la unidad de filtro 216 puede almacenar los bloques reconstruidos filtrados a DPB 218. La unidad de estimación de movimiento 222 y la unidad de compensación de movimiento 224 pueden recuperar una imagen de referencia de DPB 218, formada a partir de los bloques reconstruidos (y potencialmente filtrados), para interpredecir los bloques de imágenes posteriormente codificadas. 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.

[0152] En general, la unidad de codificación de 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 de entropía 220 puede codificar por entropía bloques de coeficientes de transformación cuantificados de la unidad de cuantificación 208. Como otro ejemplo, la unidad de codificación de entropía 220 puede codificar por entropía elementos de sintaxis de predicción (por ejemplo,

información de movimiento para interpredicción o información intramodal para intrapredicción) desde la unidad de selección de modo 202. La unidad de codificación de entropía 220 puede realizar una o más operaciones de codificación de 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 de entropía 220 puede realizar una operación de codificación de longitud variable adaptativa al contexto (CAVLC), una operación de CABAC, una operación de codificación de longitud variable a variable (V2V), una codificación aritmética binaria adaptativa al contexto basada en sintaxis (SBAC), una operación de codificación de Entropía de Particionamiento de Intervalo de Probabilidad (PIPE), una operación de codificación Exponencial-Golomb u otro tipo de operación de codificación de entropía en los datos. En algunos ejemplos, la unidad de codificación de entropía 220 puede funcionar en modo de derivación donde los elementos de sintaxis no están codificados por entropía.

[0153] La FIG. 6 es un diagrama de bloques de una unidad de codificación de entropía de ejemplo 220 que puede configurarse para realizar CABAC de acuerdo con las técnicas de esta divulgación. Se introduce un elemento de sintaxis 1180 en la unidad de codificación de entropía 220. Si el elemento de sintaxis ya es un elemento de sintaxis de valor binario (por ejemplo, un indicador u otro elemento de sintaxis que solo tiene un valor de 0 y 1), el paso de binarización puede ser omitido. Si el elemento de sintaxis es un elemento de sintaxis con valor no binario (por ejemplo, un elemento de sintaxis que puede tener valores distintos de 1 o 0), el elemento de sintaxis con valor no binario se binariza mediante el binarizador 1200. El binarizador 1200 realiza un mapeo del elemento de sintaxis con valor no binario en una secuencia de decisiones binarias. Estas decisiones binarias a menudo se llaman “bins”. Por ejemplo, para los niveles de coeficiente de transformación, el valor del nivel puede desglosarse en bins sucesivos, cada bin indica si el valor absoluto del nivel de coeficiente es o no mayor que algún valor. Por ejemplo, el bin 0 (a veces llamado indicador de significación) indica si el valor absoluto del nivel del coeficiente de transformación es mayor o no que 0. El bin 1 indica si el valor absoluto del nivel del coeficiente de transformación es mayor o no que 1, y así sucesivamente. Se puede desarrollar un mapeo único para cada elemento de sintaxis con valor no binario.

[0154] Cada bin producido por el binarizador 1200 se alimenta al lado de codificación aritmética binaria de la unidad de codificación de entropía 220. Es decir, para un conjunto predeterminado de elementos de sintaxis con valores no binarios, cada tipo de bin (por ejemplo, bin 0) se codifica antes del siguiente tipo de bin (por ejemplo, bin 1).

La codificación se puede realizar en modo regular o en modo de derivación. En el modo de derivación, el motor de codificación de derivación 1260 realiza una codificación aritmética utilizando un modelo de probabilidad fija, por ejemplo, utilizando Golomb-Rice o codificación exponencial de Golomb. El modo de derivación generalmente se usa para elementos de sintaxis más predecibles.

[0155] La codificación en modo regular implica realizar CABAC. El modo regular CABAC es para codificar valores de bin donde la probabilidad de un valor de un bin es predecible dados los valores de bins codificados previamente. La probabilidad de que un bin sea un LPS está determinada por el modelador de contexto 1220. El modelador de contexto 1220 emite el valor del bin y el estado de probabilidad para el contexto (por ejemplo, el estado de probabilidad σ , incluido el valor del LPS y la probabilidad de que ocurra el LPS). El contexto puede ser un contexto inicial para una serie de bins, o puede determinarse en base a los valores codificados de los bins codificados previamente. Como se describió anteriormente, el modelador de contexto 1220 puede actualizar el estado en función de si el bin recibido era o no el MPS o el LPS. Después de que el modelador de contexto 1220 determina el estado de probabilidad σ y contexto, el motor de codificación regular 1240 realiza BAC en el valor bin.

[0156] Volviendo a la FIG. 5, el codificador de video 200 puede emitir un flujo de bits que incluye los elementos de sintaxis codificados por entropía necesarios para reconstruir bloques de un segmento o una imagen. En particular, la unidad de codificación de entropía 220 puede emitir el flujo de bits.

[0157] Las operaciones descritas anteriormente se describen con respecto a un bloque. Dicha descripción debe entenderse como funciones para un bloque de codificación luma y/o bloques de codificación croma. Como se describió anteriormente, en algunos ejemplos, el bloque de codificación luma y los bloques de codificación croma son componentes de luma y croma de un CU. En algunos ejemplos, el bloque de codificación luma y los bloques de codificación croma son componentes de luma y croma de una PU.

[0158] En algunos ejemplos, las funciones realizadas con respecto a un bloque de codificación de luma no necesitan repetirse para los bloques de codificación de croma. Como un ejemplo, las funciones para identificar un vector de movimiento (MV) y una imagen de referencia para un bloque de codificación luma no necesitan repetirse para identificar un MV y una imagen de referencia para los bloques de croma. Por el contrario, el MV para el bloque de codificación luma se puede escalar para determinar el MV para los bloques de croma, y la imagen de referencia puede ser la misma. Como otro ejemplo,

el proceso de intrapredicción puede ser el mismo para los bloques de codificación de luma y los bloques de codificación de croma.

[0159] 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 circuitos y configuradas para obtener un índice de pendiente predefinido (por ejemplo, SlopeIdx) y un índice de desplazamiento predefinido (por ejemplo, OffsetIdx) para un contexto de una pluralidad de contextos utilizados en un proceso de codificación de entropía adaptable al contexto para codificar por entropía un valor para un elemento de sintaxis en un segmento de los datos de video; determinar, basándose en el índice de pendiente predefinido y el índice de desplazamiento predefinido, un estado de probabilidad inicial del contexto para el segmento de los datos de video; y codificar por entropía, basándose en el estado de probabilidad inicial del contexto, un bin del valor para el elemento de sintaxis.

[0160] La FIG. 7 es un diagrama de bloques que ilustra un decodificador de video 300 de ejemplo que puede realizar las técnicas de esta divulgación. La FIG. 7 se proporciona con fines de explicación y no limita las técnicas como se ejemplifica y describe ampliamente en esta divulgación. Para fines de explicación, esta divulgación describe el decodificador de video 300 de acuerdo con las técnicas de JEM, 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.

[0161] En el ejemplo de la FIG. 7, el decodificador de video 300 incluye memoria de búfer de imagen codificada (CPB) 320, unidad de decodificación de 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 la totalidad de la memoria de CPB 320, la unidad de decodificación de 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.

[0162] 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 de adición 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 paleta, una unidad de copia intrabloque (que puede formar parte de la unidad de compensación de movimiento 316), una unidad afín, una unidad de modelo lineal (LM) o similares. En otros ejemplos, el decodificador de video 300 puede incluir más, menos o componentes funcionales diferentes.

[0163] La memoria de CPB 320 puede almacenar datos de video, tales como un flujo de bits de video codificado, para ser decodificados 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 (FIG. 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 CPB 320 puede almacenar datos de video que no sean elementos de sintaxis de una imagen codificada, como datos temporales que representan salidas de las diversas unidades del decodificador de video 300. DPB 314 generalmente almacena imágenes decodificadas, que el decodificador de video 300 puede emitir y/o usar como datos de video de referencia al decodificar datos o imágenes posteriores del flujo de bits de video codificado. La memoria CPB 320 y DPB 314 pueden estar formadas por cualquiera de una variedad de dispositivos de memoria, como la memoria dinámica de acceso aleatorio (DRAM), incluida la DRAM síncrona (SDRAM), la RAM magnetorresistiva (MRAM), la RAM resistiva (RRAM) u otros tipos de dispositivos de memoria. La memoria CPB 320 y DPB 314 pueden ser proporcionadas por el mismo dispositivo de memoria o dispositivos de memoria separados. En varios ejemplos, la memoria de CPB 320 puede estar en chip con otros componentes del decodificador de video 300, o fuera de chip en relación con esos componentes.

[0164] Adicional o alternativamente, en algunos ejemplos, el decodificador de video 300 puede recuperar datos de video codificados de la memoria 120 (FIG. 1). Es decir, la memoria 120 puede almacenar datos como se discutió anteriormente con la memoria de CPB 320. Del mismo modo, la memoria 120 puede almacenar instrucciones para ser ejecutadas por el decodificador de video 300, cuando parte o la totalidad de la funcionalidad del decodificador de video 300 se implementa en el *software* para ser ejecutado por circuitos de procesamiento del decodificador de video 300.

[0165] Las diversas unidades mostradas en la FIG. 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 FIG. 5, los circuitos de función fija se refieren a circuitos que proporcionan una funcionalidad particular, y están preestablecidos en las operaciones que se pueden realizar. Los circuitos programables se refieren a 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 circuito distintos (función fija o programable), y en algunos ejemplos, la unidad o las unidades pueden ser circuitos integrados.

[0166] 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 en chip o fuera de chip puede almacenar instrucciones (por ejemplo, código de objeto) del *software* que el decodificador de video 300 recibe y ejecuta.

[0167] La unidad de decodificación de entropía 302 puede recibir datos de video codificados del 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 basados en los elementos de sintaxis extraídos del flujo de bits.

[0168] La FIG. 8 es un diagrama de bloques de una unidad de decodificación de entropía 302 de ejemplo que puede configurarse para realizar CABAC de acuerdo con las técnicas de esta divulgación. La unidad de decodificación de entropía 302 de la FIG. 8 realiza CABAC de manera inversa a la de la unidad de codificación de entropía 220 descrita en la FIG. 6. Los bits codificados del flujo de bits 2180 se introducen en la unidad de decodificación de entropía 302. Los bits codificados ingresan al modelador de contexto 2200 o al motor de decodificación de derivación 2220 en función de si se codificaron por entropía utilizando el modo regular o el modo de derivación. Si los bits

codificados se codificaron en modo de derivación, el motor de decodificación de derivación utilizará Golomb-Rice o la decodificación exponencial de Golomb, por ejemplo, para recuperar los elementos de sintaxis con valor binario o bins de elementos de sintaxis no binarios.

[0169] Si los bits codificados se codificaron en modo regular, el modelador de contexto 2200 puede determinar un modelo de probabilidad para los bits codificados y el motor de decodificación regular 2240 puede decodificar los bits codificados para producir bins de elementos de sintaxis no binarios (o los elementos de sintaxis propiamente dichos si tienen valor binario). Después de que el modelador de contexto 2200 determina el estado de probabilidad σ y contexto, el motor de decodificación regular 2240 realiza BAC para decodificar el valor de bin. En otras palabras, el motor de decodificación regular 2240 puede determinar un estado de probabilidad de un contexto, y decodificar un valor de bin basado en bins codificados previamente y un rango actual. Después de decodificar el bin, el modelador de contexto 2200 puede actualizar el estado de probabilidad del contexto en base al tamaño de la ventana y el valor del bin decodificado.

[0170] Volviendo a la FIG. 7, 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 un “bloque actual”).

[0171] La unidad de decodificación de entropía 302 puede decodificar por entropía elementos de sintaxis que definen coeficientes de transformación cuantificados de un bloque de coeficiente de transformación cuantificado, así como información de transformación, tal como un parámetro de cuantificación (QP) y/o indicaciones de modo de transformación. La unidad de cuantificación inversa 306 puede usar el QP asociado con el bloque de coeficiente de transformación cuantificado para determinar un grado de cuantificación y, asimismo, un grado de cuantificación inversa para aplicar la unidad de cuantificación inversa 306. La unidad de cuantificación inversa 306 puede, por ejemplo, realizar una operación de desplazamiento a la izquierda a nivel de bit para cuantificar inversamente los coeficientes de transformación cuantificados. La unidad de cuantificación inversa 306 puede formar así un bloque de coeficiente de transformación que incluye coeficientes de transformación.

[0172] Después de que la unidad de cuantificación inversa 306 forma el bloque de coeficiente de transformación, la unidad de procesamiento de transformada inversa 308 puede aplicar una o más transformadas inversas al bloque de coeficiente de

transformación 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 entera inversa, una transformada de Karhunen-Loeve inversa (KLT), una transformada rotacional inversa, una transformada direccional inversa u otra transformada inversa al bloque de coeficientes de transformación.

[0173] 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 de entropía 302. Por ejemplo, si los elementos de sintaxis de información de predicción indican que el bloque actual es 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 DPB 314 desde 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 (FIG. 5).

[0174] Como otro ejemplo, si los elementos de sintaxis de información de predicción indican que el bloque actual es 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. Nuevamente, la unidad de intrapredicción 318 generalmente puede realizar el proceso de intrapredicción de una manera que es sustancialmente similar a la descrita con respecto a la unidad de intrapredicción 226 (FIG. 5). La unidad de intrapredicción 318 puede recuperar datos de muestras vecinas al bloque actual del DPB 314.

[0175] La unidad de reconstrucción 310 puede reconstruir el bloque actual usando el 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.

[0176] 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 las interferencias de bloqueo a lo largo de los bordes de los bloques reconstruidos. Las operaciones de la unidad de filtro 312 no se realizan necesariamente en todos los ejemplos.

[0177] El decodificador de video 300 puede almacenar los bloques reconstruidos en DPB 314. Por ejemplo, en ejemplos donde no se realizan operaciones de la unidad de filtro 312, la unidad de reconstrucción 310 puede almacenar bloques reconstruidos en DPB 314. En ejemplos donde se realizan las funciones de la unidad de filtro 312, la unidad de filtro 312 puede almacenar los bloques reconstruidos filtrados en DPB 314. Como se discutió anteriormente, DPB 314 puede proporcionar información de referencia, tal como muestras de una imagen actual para intrapredicción e imágenes decodificadas previamente para compensación de movimiento posterior, a la unidad de procesamiento de predicción 304. Además, el decodificador de video 300 puede emitir imágenes decodificadas desde DPB para su posterior presentación en un dispositivo de visualización, tal como el dispositivo de visualización 118 de la FIG. 1.

[0178] De esta manera, 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 circuitos y configuradas para obtener un índice de pendiente predefinido (por ejemplo, Slopeldx) y un índice de desplazamiento predefinido (por ejemplo, OffsetIdx) para un contexto de una pluralidad de contextos utilizados en un proceso de codificación de entropía adaptable al contexto para codificar por entropía un valor para un elemento de sintaxis en un segmento de los datos de video; determinar, basándose en el índice de pendiente predefinido y el índice de desplazamiento predefinido, un estado de probabilidad inicial del contexto para el segmento de los datos de video; y decodificar por entropía, basándose en el estado de probabilidad inicial del contexto, un bin del valor para el elemento de sintaxis.

[0179] La FIG. 19 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 (FIG. 1 y 5), debe entenderse que otros dispositivos pueden configurarse para realizar un método similar al de la FIG. 19.

[0180] 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 calcular entonces 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 transformar y cuantificar los coeficientes del bloque residual (354). A continuación, el codificador de video 200 puede escanear los coeficientes de transformación cuantificados del bloque

residual (356). Durante el escaneo, o después del escaneo, el codificador de video 200 puede codificar por entropía los coeficientes (358). Por ejemplo, el codificador de video 200 puede codificar los coeficientes, y/u otros elementos de sintaxis, usando CAVLC o CABAC mediante las técnicas para la inicialización de probabilidad discutidas anteriormente y con referencia a la FIG. 20. El codificador de video 200 puede entonces emitir los datos codificados por entropía del bloque (360).

[0181] La FIG. 20 es un diagrama de flujo que ilustra un proceso de ejemplo para realizar la codificación de entropía basada en el contexto, de acuerdo con una o más técnicas de esta divulgación. Las técnicas de la FIG. 20 pueden realizarse mediante un codificador de video, tal como el codificador de video 200 ilustrado en las FIG. 1, 5 y 6. Para fines de ilustración, las técnicas de la FIG. 20 se describen dentro del contexto del codificador de video 200 de las FIG. 1, 5 y 6, aunque los codificadores de video que tienen configuraciones diferentes a la del codificador de video 200 pueden realizar las técnicas de la FIG. 20.

[0182] El codificador de video 200 puede obtener una cadena bin (por ejemplo, un vector binario unidimensional) para codificar usando codificación de entropía basada en contexto (por ejemplo, CABAC) (2002). Por ejemplo, la unidad de codificación de entropía 220 del codificador de video 200 puede obtener la cadena de bin al binarizar un elemento de sintaxis recibido de la unidad de selección de modo 202 del codificador de video 200.

[0183] El codificador de video 200 puede obtener un valor de inicialización predefinido para un contexto de una pluralidad de contextos (2004). Por ejemplo, la unidad de codificación de entropía 220 del codificador de video 200 puede obtener un valor de `initValue`, que puede ser una variable de seis bits.

[0184] El codificador de video 200 puede determinar, en base al valor de inicialización predefinido, un estado de probabilidad inicial del contexto para la unidad codificable independientemente (por ejemplo, segmento, mosaico, etc.) de los datos de video en el dominio lineal (2006). Por ejemplo, la unidad de codificación de entropía 220 puede determinar el estado de probabilidad inicial del contexto en el dominio lineal sin determinar de manera intermedia el estado de probabilidad inicial del contexto en el dominio logarítmico. En algunos ejemplos, la unidad de codificación de entropía 220 puede determinar el estado de probabilidad inicial sin usar una LUT para convertir el estado de probabilidad inicial del dominio logarítmico al dominio lineal.

[0185] Para determinar el estado de probabilidad inicial, la unidad de codificación de entropía 220 puede obtener un valor inicial de un parámetro de cuantificación asociado

con la unidad codificable independientemente (por ejemplo, SliceQP_Y), y obtener un valor de un punto de anclaje del parámetro de cuantificación que no sea cero (por ejemplo, $\text{QP}_{\text{anchor}}$). La unidad de codificación de entropía 220 puede determinar el estado de probabilidad inicial basándose en una diferencia entre el valor inicial del parámetro de cuantificación para el segmento y el valor del punto de anclaje del parámetro de cuantificación. Por ejemplo, la unidad de codificación de entropía 220 puede determinar el estado de probabilidad inicial de acuerdo con la siguiente ecuación:

$$\text{InitProbState} = ((m * (\text{SliceQP}_Y - \text{QP}_{\text{anchor}})) \gg r\text{shift}) + n.$$

donde InitProbState es el estado de probabilidad inicial, SliceQP_Y es el valor inicial del parámetro de cuantificación, $\text{QP}_{\text{anchor}}$ es el punto de anclaje del parámetro de cuantificación y $r\text{shift}$ es un valor de desplazamiento a la derecha.

[0186] Como se discutió anteriormente, en algunos ejemplos, la unidad de codificación de entropía 220 puede configurarse para realizar la inicialización con mayor precisión para distribuciones de probabilidad extrema. Por ejemplo, la unidad de codificación de entropía 220 puede determinar, en base al valor de inicialización predefinido, un valor de índice de pendiente y un valor de índice de desplazamiento. La unidad de codificación de entropía 220 puede determinar, en base al valor del índice de pendiente, un valor de m ; y determinar, en base al valor del índice de desplazamiento, un valor de n . En algunos ejemplos, la unidad de codificación de entropía 220 puede determinar el valor de n , de acuerdo con la siguiente ecuación $n = (\text{OffsetIdx} * 18) + 1$ donde OffsetIdx es el valor del índice de desplazamiento. Como se muestra arriba, en algunos ejemplos, la unidad de codificación de entropía 220 puede determinar el estado de probabilidad inicial basado en el valor de m y el valor de n .

[0187] El codificador de video 200 puede codificar, en un flujo de bits de video y en base al estado de probabilidad inicial del contexto, un bin de la cadena bin (2008). Por ejemplo, la unidad de codificación de entropía 220 puede generar un flujo binario que representa un valor o puntero a una probabilidad dentro de un intervalo de probabilidad codificado final del contexto.

[0188] La FIG. 21 es un diagrama de flujo que ilustra un método de ejemplo para decodificar un bloque actual de datos de video. El bloque actual puede comprender una CU actual. Aunque se describe con respecto al decodificador de video 300 (FIG. 1 y 7), debe entenderse que otros dispositivos pueden configurarse para realizar un método similar al de la FIG. 21.

[0189] 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 un bloque residual correspondiente al bloque actual (370). 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 para reproducir los coeficientes del bloque residual (372). Por ejemplo, el decodificador de video 300 puede decodificar los coeficientes y/u otros elementos de sintaxis, usando CAVLC o CABAC mediante las técnicas para la inicialización de probabilidad discutidas anteriormente y con referencia a la FIG. 22. El decodificador de video 300 puede predecir el bloque actual (374), por ejemplo, usando 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 escanear inversamente los coeficientes reproducidos (376), para crear un bloque de coeficientes de transformación cuantificados. El decodificador de video 300 puede cuantificar inversamente y transformar inversamente los coeficientes para producir un bloque residual (378). El decodificador de video 300 puede decodificar finalmente el bloque actual combinando el bloque de predicción y el bloque residual (380).

[0190] La FIG. 22 es un diagrama de flujo que ilustra un proceso de ejemplo para realizar la decodificación de entropía basada en el contexto, de acuerdo con una o más técnicas de esta divulgación. Las técnicas de la FIG. 22 pueden realizarse mediante un decodificador de video, tal como el decodificador de video 300 ilustrado en las FIG. 1, 7 y 8. Para fines de ilustración, las técnicas de la FIG. 22 se describen dentro del contexto del decodificador de video 300 de las FIG. 1, 7 y 8, aunque los decodificadores de video que tienen configuraciones diferentes a las del decodificador de video 300 pueden realizar las técnicas de la FIG. 22.

[0191] El decodificador de video 300 puede obtener, a partir de un flujo de bits de video, una cadena bin (por ejemplo, un vector binario unidimensional) para decodificar usando codificación de entropía basada en contexto (2202). Por ejemplo, la unidad de decodificación de entropía 302 del decodificador de video 300 puede obtener, a partir de la memoria CPB 320, la cadena bin. En algunos ejemplos, la cadena bin puede representar un valor o puntero a una probabilidad dentro de un intervalo de probabilidad codificado final de un contexto. En algunos ejemplos, la codificación de entropía basada en el contexto puede comprender la codificación aritmética binaria adaptativa al contexto (CABAC).

[0192] El decodificador de video 300 puede obtener un valor de inicialización predefinido para un contexto de una pluralidad de contextos (2204). Por ejemplo, la unidad de

decodificación de entropía 302 del decodificador de video 300 puede obtener un valor de initValue, que puede ser una variable de seis bits.

[0193] El decodificador de video 300 puede determinar, en base al valor de inicialización predefinido, un estado de probabilidad inicial del contexto para la unidad codificable independientemente de los datos de video (por ejemplo, segmento, mosaico, etc.) en el dominio lineal (2206). Por ejemplo, la unidad de decodificación de entropía 302 puede determinar el estado de probabilidad inicial del contexto en el dominio lineal sin determinar de manera intermedia el estado de probabilidad inicial del contexto en el dominio logarítmico. En algunos ejemplos, la unidad de decodificación de entropía 302 puede determinar el estado de probabilidad inicial sin usar una LUT para convertir el estado de probabilidad inicial del dominio logarítmico al dominio lineal.

[0194] Para determinar el estado de probabilidad inicial, la unidad de decodificación de entropía 302 puede obtener un valor inicial de un parámetro de cuantificación para la unidad codificable independientemente (por ejemplo, SliceQP_Y), y obtener un valor de un punto de anclaje del parámetro de cuantificación que no es cero (por ejemplo, QP_{anchor}). La unidad de decodificación de entropía 302 puede determinar el estado de probabilidad inicial basándose en una diferencia entre el valor inicial del parámetro de cuantificación para la unidad codificable independientemente y el valor del punto de anclaje del parámetro de cuantificación. Por ejemplo, la unidad de decodificación de entropía 302 puede determinar el estado de probabilidad inicial de acuerdo con la siguiente ecuación:

$$\text{InitProbState} = ((m * (\text{SliceQP}_Y - \text{QP}_{\text{anchor}})) \gg \text{rshift}) + n.$$

donde InitProbState es el estado de probabilidad inicial, SliceQP_Y es el valor inicial del parámetro de cuantificación, QP_{anchor} es el punto de anclaje del parámetro de cuantificación y rshift es un valor de desplazamiento a la derecha.

[0195] Como se discutió anteriormente, en algunos ejemplos, la unidad de decodificación de entropía 302 puede configurarse para realizar la inicialización con mayor precisión para distribuciones de probabilidad extrema. Por ejemplo, la unidad de decodificación de entropía 302 puede determinar, en base al valor de inicialización predefinido, un valor de índice de pendiente y un valor de índice de desplazamiento. La unidad de decodificación de entropía 302 puede determinar, en base al valor del índice de pendiente, un valor de m ; y determinar, en base al valor del índice de desplazamiento, un valor de n . En algunos ejemplos, la unidad de decodificación de entropía 302 puede determinar el valor de n , de acuerdo con la siguiente ecuación $n = (\text{OffsetIdx} * 18) + 1$ donde OffsetIdx es el valor del índice de desplazamiento. Como se muestra arriba, en

algunos ejemplos, la unidad de decodificación de entropía 302 puede determinar el estado de probabilidad inicial basado en el valor de m y el valor de n .

[0196] El decodificador de video 300 puede decodificar, basándose en el estado de probabilidad inicial del contexto, un bin de la cadena bin (2208). El decodificador de video 300 puede determinar, basándose en el bin decodificado y el estado de probabilidad inicial del contexto, un estado de probabilidad actualizado del contexto. El decodificador de video 300 puede decodificar, en base al estado de probabilidad actualizado del contexto, otro bin (2206).

[0197] Los siguientes ejemplos numerados pueden ilustrar uno o más aspectos de la divulgación:

[0198] Ejemplo 1A. Un método para la codificación de entropía de datos de video, que comprende: obtener un índice de pendiente predefinido y un índice de desplazamiento predefinido para un contexto de una pluralidad de contextos utilizados en un proceso de codificación de entropía adaptable al contexto para codificar por entropía un valor para un elemento de sintaxis en un segmento de los datos de video; determinar, en base al índice de pendiente predefinido y el índice de desplazamiento predefinido, un estado de probabilidad inicial del contexto para el segmento de los datos de video; y codificar por entropía, basándose en el estado de probabilidad inicial del contexto, un bin del valor para el elemento de sintaxis.

[0199] Ejemplo 2A. El método del ejemplo 1A, en donde el estado de probabilidad inicial representa una probabilidad inicial en un dominio lineal.

[0200] Ejemplo 3A. El método del ejemplo 1A, en donde el estado de probabilidad inicial representa una probabilidad inicial en un dominio cuadrático.

[0201] Ejemplo 4A. El método del ejemplo 3A, en donde la determinación del estado de probabilidad inicial se realiza sin usar una tabla de búsqueda (LUT) para mapear entre el índice de pendiente predefinido, el índice de desplazamiento predefinido y el estado de probabilidad inicial.

[0202] Ejemplo 5A. El método de cualquier combinación de los ejemplos 1A a 4A, en donde el estado de probabilidad inicial está representado por InitProbState, el índice de pendiente está representado por Slopeldx y el índice de desplazamiento está representado por OffsetIdx.

[0203] Ejemplo 6A. El método de cualquiera de los ejemplos 1A a 5A, en el que el proceso de codificación de entropía adaptable al contexto comprende un proceso de codificación aritmética binaria adaptativa al contexto (CABAC), o un proceso de codificación de longitud variable adaptativa al contexto (CAVLC).

[0204] Ejemplo 7A. El método de cualquiera de los ejemplos 1A a 6A, en donde la codificación comprende la decodificación.

[0205] Ejemplo 8A. El método de cualquiera de los ejemplos 1A a 7A, en donde la codificación comprende codificación.

[0206] Ejemplo 9A. Un dispositivo para codificar datos de video, que comprende uno o más medios para realizar el método de cualquiera de los ejemplos 1A-8A.

[0207] Ejemplo 10A. El dispositivo del ejemplo 9A, en donde el medio o los medios comprenden uno o más procesadores implementados en circuitos.

[0208] Ejemplo 11A. El dispositivo de cualquiera de los ejemplos 9A y 10A, que comprende además una memoria para almacenar los datos de video.

[0209] Ejemplo 12A. El dispositivo de cualquiera de los ejemplos 9A a 11A, que comprende además una pantalla configurada para visualizar datos de video decodificados.

[0210] Ejemplo 13A. El dispositivo de cualquiera de los ejemplos 9A a 12A, en donde el dispositivo comprende una o más de una cámara, una computadora, un dispositivo móvil, un dispositivo receptor de transmisión o un decodificador.

[0211] Ejemplo 14A. El dispositivo de cualquiera de los ejemplos 9A a 13A, en donde el dispositivo comprende un decodificador de video.

[0212] Ejemplo 15A. El dispositivo de cualquiera de los ejemplos 9A a 14A, en donde el dispositivo comprende un codificador de video.

[0213] Ejemplo 16A. 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 cualquiera de los ejemplos 1A a 8A.

[0214] Ejemplo 1B. Un método para la codificación de entropía de datos de video, que comprende: obtener un índice de desplazamiento predefinido para un contexto de una pluralidad de contextos utilizados en un proceso de codificación de entropía adaptable al contexto para codificar por entropía un valor para un elemento de sintaxis en un segmento de los datos de video; determinar, en base al índice de desplazamiento predefinido, un estado de probabilidad inicial del contexto para el segmento de los datos de video; y codificar por entropía, en base al estado de probabilidad inicial del contexto, un bin del valor para el elemento de sintaxis.

[0215] Ejemplo 2B. El método del ejemplo 1B, en donde el estado de probabilidad inicial representa una probabilidad inicial en un dominio lineal.

[0216] Ejemplo 3B. El método del ejemplo 1B, en donde el estado de probabilidad inicial representa una probabilidad inicial en un dominio cuadrático.

[0217] Ejemplo 4B. El método del ejemplo 3B, en donde la determinación del estado de probabilidad inicial se realiza sin usar una tabla de búsqueda (LUT) para mapear entre el índice de desplazamiento predefinido y el estado de probabilidad inicial.

[0218] Ejemplo 5B. El método de cualquier combinación de los ejemplos 1B a 4B, en donde el estado de probabilidad inicial está representado por InitProbState, y el índice de desplazamiento está representado por OffsetIdx.

[0219] Ejemplo 6B. El método de cualquiera de los ejemplos 1B a 5B, en donde el proceso de codificación de entropía adaptable al contexto comprende un proceso de codificación aritmética binaria adaptativa al contexto (CABAC), o un proceso de codificación de longitud variable adaptativa al contexto (CAVLC).

[0220] Ejemplo 7B. El método de cualquiera de los ejemplos 1B a 6B, en donde la codificación comprende la decodificación.

[0221] Ejemplo 8B. El método de cualquiera de los ejemplos 1B a 7B, en donde la codificación comprende la codificación.

[0222] Ejemplo 9B. Un dispositivo para codificar datos de video, que comprende uno o más medios para realizar el método de cualquiera de los ejemplos 1B-8B.

[0223] Ejemplo 10B. El dispositivo del ejemplo 9B, en el que el medio o los medios comprenden uno o más procesadores implementados en circuitos.

[0224] Ejemplo 11B. El dispositivo de cualquiera de los ejemplos 9B y 10B, que comprende además una memoria para almacenar los datos de video.

[0225] Ejemplo 12B. El dispositivo de cualquiera de los ejemplos 9B a 11B, que comprende además una pantalla configurada para visualizar datos de video decodificados.

[0226] Ejemplo 13B. El dispositivo de cualquiera de los ejemplos 9B a 12B, en donde el dispositivo comprende una o más de una cámara, una computadora, un dispositivo móvil, un dispositivo receptor de transmisión o un decodificador.

[0227] Ejemplo 14B. El dispositivo de cualquiera de los ejemplos 9B a 13B, en donde el dispositivo comprende un decodificador de video.

[0228] Ejemplo 15B. El dispositivo de cualquiera de los ejemplos 9B a 14B, en donde el dispositivo comprende un codificador de video.

[0229] Ejemplo 16B. Un medio de almacenamiento legible por computadora que tiene almacenadas las instrucciones que, cuando se ejecutan, hacen que uno o más procesadores realicen el método de cualquiera de los ejemplos 1B a 8B.

[0230] Ejemplo 1C. Un método para la codificación de entropía de datos de video, que comprende: determinar, en base al valor de una variable, un primer valor intermedio;

determinar, basándose en el primer valor intermedio, un valor de un estado de probabilidad de baja precisión de un modelo de contexto, en donde determinar el valor del estado de probabilidad de baja precisión comprende desplazar a la derecha el primer valor intermedio; y codificar, en base al valor del estado de probabilidad de baja precisión del modelo de contexto, un valor de al menos un bit de un elemento de sintaxis.

[0231] Ejemplo 2C. El método del ejemplo 1C, en donde determinar el primer valor intermedio comprende determinar el primer valor intermedio de acuerdo con la siguiente ecuación: $q = \text{Clip3}(18, 46, \text{SliceQP}_Y) - 16$ donde q es el primer valor intermedio, y SliceQP_Y es el valor de la variable.

[0232] Ejemplo 3C. El método del ejemplo 1C o del ejemplo 2C, en donde determinar el valor del estado de probabilidad de baja precisión del modelo de contexto comprende determinar el valor del estado de probabilidad de baja precisión del modelo de contexto de acuerdo con la siguiente ecuación: $\text{ProbabilityStateL} = 16 * a + ((b - a) * q >> 1) + 8$ donde q es el primer valor intermedio, ProbabilityStateL es el valor del estado de probabilidad de baja precisión del modelo de contexto, a es un segundo valor intermedio y b es un tercer valor intermedio.

[0233] Ejemplo 4C. El método del ejemplo 3C, que comprende además determinar valores para a y b de acuerdo con las siguientes ecuaciones: $a = (\text{initValue} >> 3) * 9$ $b = (\text{initValue} \& 7) * 9$, donde initValue es un valor de inicialización.

[0234] Ejemplo 5C. El método de cualquiera de los ejemplos 1C – 4C, que comprende además determinar, en base al valor del estado de probabilidad de baja precisión del modelo de contexto, un valor de un estado de probabilidad de alta precisión del modelo de contexto.

[0235] Ejemplo 6C. El método del ejemplo 5C, en el que determinar el valor del estado de probabilidad de alta precisión del modelo de contexto comprende determinar el valor del estado de probabilidad de alta precisión del modelo de contexto de acuerdo con la siguiente ecuación: $\text{ProbabilityStateH} = \text{ProbabilityStateL} << 4$, donde ProbabilityStateH es el valor del estado de probabilidad de alta precisión del modelo de contexto, y ProbabilityStateL es el valor del estado de probabilidad de baja precisión del modelo de contexto.

[0236] Ejemplo 7C. El método de cualquiera de los ejemplos 1C – 6C, en donde el proceso de codificación de entropía adaptable al contexto comprende un proceso de codificación aritmética binaria adaptativa al contexto (CABAC), o un proceso de codificación de longitud variable adaptativa al contexto (CAVLC).

[0237] Ejemplo 8C. El método de cualquiera de los ejemplos 1C-7C, en donde la

codificación comprende la decodificación.

[0238] Ejemplo 9C. El método de cualquiera de los ejemplos 1C-8C, en donde la codificación comprende la codificación.

[0239] Ejemplo 10C. Un dispositivo para codificar datos de video, que comprende uno o más medios para realizar el método de cualquiera de los ejemplos 1C-9C.

[0240] Ejemplo 11C. El dispositivo del ejemplo 10C, en donde el medio o los medios comprenden uno o más procesadores implementados en circuitos.

[0241] Ejemplo 12C. El dispositivo de cualquiera de los ejemplos 10C y 11C, que comprende además una memoria para almacenar los datos de video.

[0242] Ejemplo 13C. El dispositivo de cualquiera de los ejemplos 10C-12C, que comprende además una pantalla configurada para visualizar datos de video decodificados.

[0243] Ejemplo 14C. El dispositivo de cualquiera de los ejemplos 10C-13C, en donde el dispositivo comprende una o más de una cámara, una computadora, un dispositivo móvil, un dispositivo receptor de transmisión o un decodificador.

[0244] Ejemplo 15C. El dispositivo de cualquiera de los ejemplos 10C-14C, en donde el dispositivo comprende un decodificador de video.

[0245] Ejemplo 16C. El dispositivo de cualquiera de los ejemplos 10C-15C, en donde el dispositivo comprende un codificador de video.

[0246] Ejemplo 17C. Un medio de almacenamiento legible por computadora que ha almacenado instrucciones que, cuando se ejecutan, hacen que uno o más procesadores realicen el método de cualquiera de los ejemplos 1C-9C.

[0247] Debe reconocerse que, según el ejemplo, ciertos actos o eventos de cualquiera de las técnicas descritas en el presente documento pueden realizarse en una secuencia diferente, pueden agregarse, fusionarse u omitirse por completo (por ejemplo, no todos los actos o eventos descritos son necesarios para la práctica de las técnicas). Además, en ciertos ejemplos, los actos o eventos pueden realizarse simultáneamente, por ejemplo, a través del procesamiento de subprocesos múltiples, procesamiento de interrupción o procesadores múltiples, en lugar de secuencialmente.

[0248] En uno o más ejemplos, las funciones descritas pueden implementarse en *hardware*, *software*, *firmware* o cualquier combinación de estos. Si se implementa en *software*, las funciones pueden almacenarse o transmitirse como una o más instrucciones o código en un medio legible por computadora y ejecutarse por una unidad de procesamiento basada en *hardware*. Los medios legibles por computadora pueden incluir medios de almacenamiento legibles por computadora, que corresponden a un

medio tangible como los medios de almacenamiento de datos o medios de comunicación que incluyen 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 tangibles de almacenamiento legibles por computadora que no son transitorios o (2) un medio de comunicación tal como una señal o una onda portadora. Los medios de almacenamiento de datos pueden ser cualquier medio disponible al que puedan acceder una o más computadoras o uno o más procesadores para recuperar instrucciones, código 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.

[0249] A modo de ejemplo, y sin limitación, dichos medios de almacenamiento legibles por computadora pueden comprender RAM, ROM, EEPROM, CD-ROM u otro almacenamiento en disco óptico, almacenamiento en disco magnético u otros dispositivos de almacenamiento magnético, memoria flash o cualquier otro medio que puede usarse para almacenar el código de programa deseado en forma de instrucciones o estructuras de datos y al que puede acceder una computadora. Además, cualquier conexión se denomina correctamente un medio legible por computadora. Por ejemplo, si las instrucciones se transmiten desde un sitio web, un servidor u otra fuente remota utilizando un cable coaxial, un cable de fibra óptica, un par trenzado, una línea de abonado digital (DSL) o tecnologías inalámbricas como infrarrojos, radio y microondas, entonces el cable coaxial, el cable de fibra óptica, el par trenzado, la DSL o las tecnologías inalámbricas como infrarrojos, radio y microondas están incluidos en la definición de medio. Sin embargo, debe entenderse 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 se dirigen a medios de almacenamiento tangibles no transitorios. Disco óptico y disco magnético, como se usa en este documento, incluye disco compacto (CD), disco láser, disco óptico, disco versátil digital (DVD), disquete y disco Blu-ray, donde los discos magnéticos generalmente reproducen datos magnéticamente, mientras que los discos ópticos reproducen datos ópticamente con láser. Las combinaciones de lo anterior también deben incluirse dentro del alcance de los medios legibles por computadora.

[0250] Las instrucciones pueden ser ejecutadas por uno o más procesadores, tales como uno o más procesadores de señal digital (DSP), microprocesadores de propósito general, circuitos integrados de aplicación específica (ASIC), matrices de puertas

programables en campo (FPGA) u otro equivalente integrado o circuitos lógicos discretos. En consecuencia, los términos “procesador” y “circuitos de procesamiento”, como se usan en el presente documento, pueden referirse a cualquiera de las estructuras anteriores o cualquier otra estructura adecuada para la implementación de las técnicas descritas en el presente documento. Además, en algunos aspectos, la funcionalidad descrita en este documento puede proporcionarse dentro de módulos dedicados de *hardware* y/o *software* configurados para codificación y decodificación, o incorporados en un códec combinado. Además, las técnicas podrían implementarse completamente en uno o más circuitos o elementos lógicos.

[0251] Las técnicas de esta divulgación pueden implementarse en una amplia variedad de dispositivos o aparatos, incluyendo un teléfono inalámbrico, un circuito integrado (IC) o un conjunto de IC (por ejemplo, un conjunto de chips). En esta divulgación se describen diversos componentes, módulos o unidades para enfatizar los aspectos funcionales de los dispositivos configurados para realizar las técnicas divulgadas, pero no necesariamente requieren la realización mediante diferentes unidades de *hardware*. Por el contrario, como se describió anteriormente, varias unidades pueden combinarse en una unidad de *hardware* de códec o proporcionarse mediante una colección de unidades de *hardware* interoperativas, que incluyen uno o más procesadores como se describe anteriormente, junto con el *software* y/o *firmware* adecuado.

[0252] Se han descrito varios ejemplos. Estos y otros ejemplos están dentro del alcance de las siguientes reivindicaciones.