(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 636 967**

(51) Int. CI.:

H04N 19/597 (2014.01)
H04N 19/56 (2014.01)
H04N 13/00 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **06.03.2014 PCT/US2014/021343**

(87) Fecha y número de publicación internacional: **12.09.2014 WO14138450**

(96) Fecha de presentación y número de la solicitud europea: **06.03.2014 E 14712957 (1)**

(97) Fecha y número de publicación de la concesión europea: **10.05.2017 EP 2965522**

(54) Título: **Vectores de disparidad derivados en codificación de vídeo 3D**

(30) Prioridad:

06.03.2013 US 201361773680 P
05.04.2013 US 201361809174 P
05.03.2014 US 201414198442

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
10.10.2017

(73) Titular/es:

QUALCOMM INCORPORATED (100.0%)
International IP Administration,
5775 Morehouse Drive
San Diego, California 92121-1714, US

(72) Inventor/es:

CHEN, YING y
ZHANG, LI

(74) Agente/Representante:

FORTEA LAGUNA, Juan José

DESCRIPCIÓN

Vectores de disparidad derivados en codificación de vídeo 3D

[0001] Esta solicitud reivindica el beneficio de la solicitud provisional de Estados Unidos nº 61/773,680, presentada el 6 de marzo de 2013, y de la solicitud de patente provisional de Estados Unidos nº 61/809,174, presentada el 5 de abril de 2013.

Campo técnico

[0002] Esta divulgación se refiere a técnicas para la codificación de vídeo, y más específicamente a técnicas para codificación de vídeo de 3 dimensiones (3D).

Antecedentes

[0003] Las capacidades de vídeo digital pueden incorporarse a una amplia gama de dispositivos, incluidos televisores digitales, sistemas de difusión directa digital, sistemas de radiodifusión inalámbrica, asistentes digitales personales (PDA), ordenadores portátiles o de escritorio, cámaras digitales, dispositivos de grabación digitales, reproductores de medios digitales, dispositivos de videojuegos, consolas de videojuegos, teléfonos celulares o de radio por satélite, dispositivos de videoconferencia y similares. Los dispositivos de vídeo digital implementan técnicas de compresión de vídeo, tales como las descritas en las normas definidas por MPEG-2, MPEG-4, ITU-T H.263, ITU-T H.264/MPEG-4, Parte 10, Codificación Avanzada de Vídeo (AVC), la Norma de Codificación de Vídeo de Alta Eficacia (HEVC) y las ampliaciones de dichas normas, para transmitir, recibir y almacenar información de vídeo digital. Los dispositivos de vídeo pueden transmitir, recibir, codificar, decodificar y/o almacenar información de vídeo digital más eficazmente, implementando dichas técnicas de compresión de vídeo.

[0004] Las técnicas de compresión de vídeo llevan a cabo la predicción espacial (intraimagen) y/o la predicción temporal (interimagen) para reducir o eliminar la redundancia intrínseca en las secuencias de vídeo. Para la codificación de vídeo basada en bloques, un segmento de vídeo (es decir, una trama de vídeo o una parte de una trama de vídeo) puede dividirse en bloques de vídeo. Los bloques de vídeo de un segmento intracodificado (I) de una imagen se codifican mediante predicción espacial con respecto a unas muestras de referencia de bloques vecinos de la misma imagen. Los bloques de vídeo de un segmento intercodificado (P o B) de una imagen pueden usar predicción espacial con respecto a unas muestras de referencia de bloques vecinos de la misma imagen, o la predicción temporal con respecto a unas muestras de referencia de otras imágenes de referencia. Las imágenes pueden denominarse tramas, y las imágenes de referencia pueden denominarse tramas de referencia.

[0005] La predicción espacial o temporal dan como resultado un bloque predictivo para un bloque que se va a codificar. Los datos residuales representan diferencias de píxeles entre el bloque original que se va a codificar y el bloque predictivo. Un bloque intercodificado se codifica de acuerdo con un vector de movimiento que apunta a un bloque de muestras de referencia que forman el bloque predictivo, y los datos residuales que indican la diferencia entre el bloque codificado y el bloque predictivo. Un bloque intracodificado se codifica de acuerdo con un modo de codificación intra y los datos residuales. Para una mayor compresión, los datos residuales pueden transformarse desde el dominio del píxel a un dominio de transformada, dando como resultado coeficientes residuales, los cuales pueden cuantificarse posteriormente. Los coeficientes cuantificados, inicialmente dispuestos en una matriz bidimensional, pueden explorarse con el fin de producir un vector unidimensional de coeficientes, y puede aplicarse la codificación de entropía para lograr aún más compresión.

[0006] Las ampliaciones de algunas de las normas mencionadas anteriormente, incluida la H.264/AVC, proporcionan técnicas para la codificación de vídeo multivista con el fin de producir vídeo estéreo o tridimensional ("3D"). En particular, se han propuesto técnicas para la codificación multivista para su uso en AVC, incluida una norma de codificación de vídeo multivista (MVC) (que se ha convertido en la ampliación multivista de H.264/AVC). Se ha elaborado también una norma de codificación de vídeo escalable (SVC) como una ampliación de H.264/AVC.

[0007] Típicamente, el vídeo estéreo se obtiene utilizando dos vistas, por ejemplo, una vista izquierda y una vista derecha. Una imagen de la vista izquierda puede mostrarse de forma sustancialmente simultánea con una imagen de la vista derecha para conseguir un efecto de vídeo tridimensional. Por ejemplo, un usuario puede llevar gafas pasivas polarizadas que filtran la vista izquierda de la vista derecha. En otros ejemplos, las imágenes de las dos vistas se pueden mostrar en rápida sucesión, y el usuario puede usar gafas activas con obturación rápida de los ojos izquierdo y derecho a la misma frecuencia, pero con un desplazamiento de fase de 90 grados.

Resumen

[0008] En general, esta divulgación describe técnicas para codificación de vídeo 3D. En particular, esta divulgación está relacionada con la derivación de vectores de disparidad. La presente invención está definida por las reivindicaciones adjuntas.

[0009] Los detalles de uno o más ejemplos se exponen en los dibujos adjuntos y en la descripción siguiente. Otras características, objetivos y ventajas resultarán evidentes a partir de la descripción, los dibujos y las reivindicaciones.

Breve descripción de los dibujos

[0010]

La FIG. 1 es un diagrama de bloques que ilustra un ejemplo de sistema de codificación y decodificación de vídeo que puede utilizar las técnicas descritas en esta divulgación.
La FIG. 2 es un diagrama conceptual que ilustra un ejemplo de estructura de predicción para codificación multivista.
La FIG. 3 es un diagrama conceptual que ilustra un ejemplo de orden de decodificación multivista.
La FIG. 4 es un ejemplo de visualización conceptual de la predicción de síntesis de vistas basada en bloques VSP (B-VSP) basada en la distorsión regresiva.
La FIG. 5 es un diagrama conceptual que ilustra ejemplos de bloques vecinos en el espacio para un proceso de derivación de vectores de disparidad basado en bloques vecinos (NBDV).
La FIG. 6 es un diagrama conceptual que ilustra ejemplos de bloques vecinos en el tiempo para un proceso de derivación de NBDV.
La FIG. 7 es un diagrama de bloques que ilustra un ejemplo de codificador de vídeo que puede implementar una o más técnicas de esta divulgación.
La FIG. 8 es un diagrama de bloques que ilustra un ejemplo de decodificador de vídeo que puede implementar una o más técnicas de esta divulgación.
La FIG. 9A es un diagrama de flujo que ilustra un ejemplo de operación de un codificador de vídeo de acuerdo con un ejemplo de esta divulgación.
La FIG. 9B es un diagrama de flujo que ilustra un ejemplo de operación de un decodificador de vídeo de acuerdo con un ejemplo de esta divulgación.
La FIG. 10 es un diagrama de flujo que ilustra un ejemplo de operación de derivación de vector de disparidad de acuerdo con un ejemplo de esta divulgación.

Descripción detallada

[0011] En general, esta divulgación describe unas técnicas para codificación de vídeo multivista basado en códecs avanzados, que incluyen la codificación de dos o más vistas con el códec H.264/de codificación de vídeo avanzada (AVC) (por ejemplo, en una ampliación de codificación de vídeo multivista (MVC) de H.264 AVC). Esta divulgación propone técnicas relacionadas con la derivación de vectores de disparidad. La codificación de vídeo basada en 3D AVC (es decir, 3D-AVC) es una ampliación de vídeo más profundidad compatible con AVC de H.264/AVC para la codificación de vídeo de 3 dimensiones (3D). De forma similar a la ampliación MVC más profundidad (MVC+D) de H.264/AVC, la 3D-AVC admite la codificación de componentes de mapa de textura y profundidad. Sin embargo, las técnicas de esta divulgación pueden ser genéricamente aplicables a cualquier técnica de codificación de vídeo multivista, incluidas las ampliaciones multivista de la norma de codificación de vídeo de alta eficacia (HEVC).

[0012] En la codificación de vídeo multivista, el contenido de vídeo de diferentes vistas puede representar diferentes perspectivas. Por ejemplo, un bloque de vídeo de una imagen de una primera vista puede incluir contenido de vídeo que es similar a un bloque de vídeo de una imagen de una segunda vista. En este ejemplo, la ubicación del bloque de vídeo en la imagen de la primera vista y la ubicación del bloque de vídeo en la imagen de la segunda vista pueden ser diferentes. Por ejemplo, puede haber algún desplazamiento (es decir, disparidad) entre las ubicaciones de los bloques de vídeo de las diferentes vistas. En la codificación de vídeo multivista, la predicción entre vistas basada en las componentes de vista reconstruidos a partir de diferentes vistas puede estar habilitada. La predicción entre vistas puede lograr ganancias de codificación explotando el hecho de que las imágenes de cada vista que representan la misma instancia de vídeo de tiempo pueden incluir contenido de vídeo similar.

[0013] Cuando un bloque de vídeo de una imagen actual se codifica usando predicción entre vistas, el bloque puede tener un vector de movimiento que indica una ubicación en una imagen de referencia entre vistas. Una imagen de referencia entre vistas puede ser una imagen de referencia que está en (es decir, asociada con) la misma instancia de tiempo que una imagen actual, pero está en (es decir, asociada con) una vista diferente a la imagen actual. Si un vector de movimiento de un bloque indica una ubicación en una imagen de referencia entre vistas, el vector de movimiento puede denominarse vector de movimiento de disparidad. Un codificador de vídeo (por ejemplo, un codificador de vídeo o un decodificador de vídeo) puede usar un vector de movimiento de disparidad de un bloque actual para determinar un bloque predictivo para el bloque actual. Si el codificador de vídeo es un codificador de vídeo, el codificador de vídeo puede usar el bloque predictivo para el bloque actual a fin de generar datos residuales para el bloque actual. Si el codificador de vídeo es un decodificador de vídeo, el codificador de vídeo puede usar el bloque predictivo para el bloque actual y los datos residuales para el bloque actual para reconstruir los valores de muestra para el bloque de vídeo actual.

[0014] Por otra parte, un bloque de una imagen particular puede tener información de movimiento o datos residuales que son similares a la información de movimiento o los datos residuales de un bloque correspondiente de una

imagen de referencia entre vistas. Por consiguiente, un codificador de vídeo puede predecir la información de movimiento o los datos residuales de un bloque actual en una imagen actual basándose en información de movimiento o datos residuales de un bloque correspondiente de una imagen de referencia entre vistas. El codificador de vídeo puede determinar un vector de disparidad para el bloque actual con el fin de determinar una ubicación del bloque correspondiente dentro de la imagen de referencia entre vistas. El codificador de vídeo puede predecir la información de movimiento o los datos residuales del bloque actual basándose en la información de movimiento o los datos residuales del bloque correspondiente de la imagen de referencia entre vistas, independientemente de si el bloque actual tiene un vector de movimiento de disparidad. Por lo tanto, si se predice la información de movimiento o los datos residuales de un bloque actual basándose en la información de movimiento o en los datos residuales de un bloque correspondiente de una imagen de referencia entre vistas, se dice que el bloque actual tiene un vector de disparidad. El vector de disparidad puede denominarse vector de disparidad implícita (IDV) cuando el vector de disparidad se utiliza para el proceso de derivación de vectores de disparidad de bloques codificados posteriormente. El vector de disparidad para el bloque actual puede ser igual al vector de disparidad para uno de los bloques anteriores.

[0015] El codificador de vídeo puede utilizar un proceso de derivación de vectores de paridad basado en bloques vecinos (NBDV) para derivar un vector de disparidad para un bloque actual. En el proceso de derivación de NBDV, el codificador de vídeo puede comprobar bloques vecinos del bloque actual. Los bloques vecinos pueden incluir bloques vecinos en el espacio y bloques vecinos en el tiempo. Los bloques vecinos en el espacio están en la misma imagen que el bloque actual (es decir, la imagen actual). Los bloques vecinos en el tiempo están en una o más imágenes distintas a la imagen actual. Cuando el codificador de vídeo comprueba un bloque vecino, el codificador de vídeo puede determinar si el bloque vecino tiene un vector de movimiento de disparidad. Cuando el codificador de vídeo determina que uno de los bloques vecinos tiene un vector de movimiento de disparidad, el codificador de vídeo puede dejar de comprobar bloques vecinos y puede convertir el vector de movimiento de disparidad del bloque vecino en el vector de disparidad para el bloque actual. Además, si ninguno de los bloques vecinos tiene un vector de movimiento de disparidad, el codificador de vídeo puede determinar si alguno de los bloques vecinos en el espacio tiene un IDV. Cuando el codificador de vídeo determina que uno de los bloques vecinos en el espacio tiene un IDV, el codificador de vídeo puede dejar de comprobar los bloques vecinos y puede convertir el IDV del bloque vecino en el vector de disparidad para el bloque actual.

[0016] Existen varios problemas con respecto a los procesos de derivación de NBDV existentes. Por ejemplo, el uso de IDV en el proceso de derivación de NBDV puede requerir un aumento significativo de los requisitos de almacenamiento y el número de accesos a la memoria. Las técnicas de esta divulgación pueden abordar dichos problemas con respecto a los procesos de derivación de NBDV. Por ejemplo, un codificador de vídeo puede derivar, de acuerdo con una técnica de esta divulgación, basándose al menos en parte en un vector de disparidad derivado (DDV) para un segmento de una imagen actual de los datos de vídeo, un NBDV para un primer bloque del segmento. El segmento puede incluir uno o más bloques de datos de vídeo. El codificador de vídeo puede codificar el primer bloque basándose al menos en parte en el NBDV para el primer bloque. Además, el codificador de vídeo puede almacenar el NBDV para el primer bloque como un DDV actualizado para el segmento. Después de almacenar el NBDV para el primer bloque como el DDV actualizado, el codificador de vídeo puede derivar, basándose al menos en parte en el DDV actualizado, un NBDV para un segundo bloque del segmento. Además, el codificador de vídeo puede codificar el segundo bloque basándose al menos en parte en el NBDV para el segundo bloque. Este proceso puede continuar para cada bloque del segmento. En particular, el DDV para el segmento puede actualizarse basándose en el NBDV del bloque codificado anterior y, a continuación, el DDV actualizado se utiliza para derivar el NBDV para el bloque siguiente. Usando un DDV de esta manera, el codificador de vídeo puede ser capaz de determinar vectores de disparidad más precisos para bloques del segmento. El aumento de la precisión de los vectores de disparidad puede disminuir el tamaño del flujo de bits.

[0017] La FIG. 1 es un diagrama de bloques que ilustra un ejemplo de sistema de codificación y decodificación de vídeo 10 que puede utilizar las técnicas descritas en esta divulgación. Tal como se utiliza en el presente documento, el término "codificador de vídeo" se refiere genéricamente tanto a codificadores de vídeo como a decodificadores de vídeo. En esta divulgación, los términos "codificación de vídeo" o "codificación" pueden referirse genéricamente a la codificación de vídeo o la decodificación de vídeo.

[0018] Como se muestra en la FIG. 1, el sistema 10 incluye un dispositivo de origen 12 que genera datos de vídeo codificado, que un dispositivo de destino 14 va a decodificar en un momento posterior. En consecuencia, el dispositivo de origen 12 puede denominarse dispositivo de codificación de vídeo o aparato de codificación de vídeo. El dispositivo de destino 14 puede decodificar los datos de vídeo codificado generados por el dispositivo de origen 12. En consecuencia, el dispositivo de destino 14 puede denominarse dispositivo de decodificación de vídeo o aparato de decodificación de vídeo. El dispositivo de origen 12 y el dispositivo de destino 14 pueden ser ejemplos de dispositivos de codificación de vídeo o aparatos de codificación de vídeo.

[0019] El dispositivo de origen 12 y el dispositivo de destino 14 pueden comprender cualquiera entre una amplia gama de dispositivos, que incluyen ordenadores de sobremesa, ordenadores plegables (es decir, portátiles), ordenadores de tableta, decodificadores, equipos telefónicos de mano tales como los denominados teléfonos “inteligentes”, televisores, cámaras, dispositivos de visualización, reproductores de medios digitales, consolas de

videojuegos, dispositivos de transmisión de vídeo o similares. En algunos casos, el dispositivo de origen 12 y el dispositivo de destino 14 pueden estar equipados para la comunicación inalámbrica.

[0020] En el ejemplo de la FIG. 1, el dispositivo de origen 12 incluye una fuente de vídeo 18, un codificador de vídeo 20 y una interfaz de salida 22. En algunos ejemplos, la interfaz de salida 22 incluye un modulador/desmodulador (módem) y/o un transmisor. La fuente de vídeo 18 puede incluir una fuente tal como un dispositivo de captura de vídeo, (por ejemplo, una videocámara), un archivo de vídeo que contiene vídeo previamente capturado, una interfaz de alimentación de vídeo para recibir vídeo desde un proveedor de contenido de vídeo y/o un sistema de gráficos de ordenador para generar datos de gráficos de ordenador como vídeo de origen, o una combinación de dichas fuentes. En un ejemplo, si la fuente de vídeo 18 es una videocámara, el dispositivo de origen 12 y el dispositivo de destino 14 pueden formar los denominados teléfonos con cámara o videoteléfonos. Sin embargo, las técnicas descritas en esta divulgación pueden ser aplicables a la codificación de vídeo en general, y pueden aplicarse a aplicaciones inalámbricas y/o alámbricas.

[0021] El vídeo capturado, precapturado o generado por ordenador puede ser codificado por el codificador de vídeo 20. La interfaz de salida 22 del dispositivo de origen 12 puede transmitir los datos de vídeo codificado directamente al dispositivo de destino 14. El dispositivo de almacenamiento 34 puede almacenar los datos de vídeo codificado para un acceso posterior por el dispositivo de destino 14 u otros dispositivos, para decodificación y/o reproducción.

[0022] El dispositivo de destino 14 incluye una interfaz de entrada 28, un decodificador de vídeo 30 y un dispositivo de visualización 32. En algunos casos, la interfaz de entrada 28 puede incluir un receptor y/o un módem. La interfaz de entrada 28 del dispositivo de destino 14 puede recibir los datos de vídeo codificado por un enlace 16. Los datos de vídeo codificado, comunicados por el enlace 16, o proporcionados en el dispositivo de almacenamiento 34, pueden incluir una diversidad de elementos sintácticos generados por el codificador de vídeo 20, para su uso por un decodificador de vídeo, tal como el decodificador de vídeo 30, en la decodificación de los datos de vídeo. Tales elementos sintácticos pueden incluirse con los datos de vídeo codificado, transmitidos en un medio de comunicación, almacenarse en un medio de almacenamiento o almacenarse en un servidor de ficheros.

[0023] El dispositivo de visualización 32 puede estar integrado con, o ser externo a, el dispositivo de destino 14. En algunos ejemplos, el dispositivo de destino 14 puede incluir un dispositivo de visualización integrado y también estar configurado para interconectarse con un dispositivo de visualización externo. En otros ejemplos, el dispositivo de destino 14 puede ser un dispositivo de visualización. En general, el dispositivo de visualización 32 visualiza los datos de vídeo decodificado ante un usuario, y puede comprender cualquiera entre una variedad de dispositivos de visualización, tales como una pantalla de cristal líquido (LCD), una pantalla de plasma, una pantalla de diodos orgánicos emisores de luz (OLED) u otro tipo de dispositivo de visualización.

[0024] Como se ha indicado anteriormente, el dispositivo de destino 14 puede recibir datos de vídeo codificado que se van a decodificar mediante el enlace 16. El enlace 16 puede comprender un tipo de medio o dispositivo capaz de desplazar los datos de vídeo codificado desde el dispositivo de origen 12 al dispositivo de destino 14. En un ejemplo, el enlace 16 puede comprender un medio de comunicación para permitir al dispositivo de origen 12 transmitir datos de vídeo codificado directamente al dispositivo de destino 14 en tiempo real. Los datos de vídeo codificado pueden modularse de acuerdo con una norma de comunicación, tal como un protocolo de comunicación inalámbrica, y transmitidos al dispositivo de destino 14. El medio de comunicación puede comprender cualquier medio de comunicación inalámbrico o alámbrico, tal 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, tal como una red de área local, una red de área extensa o una red global tal como Internet. El medio de comunicación puede incluir encaminadores, conmutadores, estaciones base o cualquier otro equipo que pueda ser útil para facilitar la comunicación desde el dispositivo de origen 12 al dispositivo de destino 14.

[0025] De forma alternativa, los datos codificados pueden transmitirse desde la interfaz de salida 22 a un dispositivo de almacenamiento 34. De forma similar, se puede acceder a los datos codificados desde el dispositivo de almacenamiento 34 mediante una interfaz de entrada 28. El dispositivo de almacenamiento 34 puede incluir cualquiera entre una diversidad de medios de almacenamiento de datos, de acceso distribuido o local, tales como una unidad de disco duro, unos discos Blu-ray, discos DVD, discos CD-ROM, una memoria flash, memoria volátil o no volátil o cualquier otro medio de almacenamiento digital adecuado, para almacenar datos de vídeo codificado. En un ejemplo adicional, el dispositivo de almacenamiento 34 puede corresponder a un servidor de archivos o a otro dispositivo de almacenamiento intermedio que pueda retener el vídeo codificado generado por el dispositivo de origen 12. El dispositivo de destino 14 puede acceder a los datos de vídeo almacenados del dispositivo de almacenamiento 34 a través de transmisión en continuo o descarga. El servidor de archivos puede ser cualquier tipo de servidor capaz de almacenar datos de vídeo codificado y transmitir esos datos de vídeo codificado al dispositivo de destino 14. Entre los ejemplos de servidores de archivos se incluyen servidores web (por ejemplo, para sitios web), servidores de protocolo de transferencia de archivos (FTP), unos dispositivos de almacenamiento conectado en red (NAS), unas unidades de disco local u otros tipos de dispositivos o sistemas que proporcionan datos a otros dispositivos informáticos. El dispositivo de destino 14 puede acceder a los datos de vídeo codificado a través de cualquier conexión de datos estándar, incluyendo una conexión a Internet. Esto puede incluir un canal inalámbrico (por ejemplo, una conexión de wifi), una conexión alámbrica (por ejemplo, DSL, módem de cable, etc.), o una

combinación de ambos que sea adecuada para acceder a datos de vídeo codificado, almacenados en un servidor de archivos. La transmisión de datos de vídeo codificado desde el dispositivo de almacenamiento 34 puede ser una transmisión en continuo, una transmisión de descarga o una combinación de ambas.

[0026] Las técnicas de esta divulgación no están limitadas necesariamente a aplicaciones o configuraciones inalámbricas. Las técnicas pueden aplicarse a la codificación de vídeo, en soporte de cualquiera entre una diversidad de aplicaciones de multimedios, tales como difusiones de televisión por el aire, transmisiones de televisión por cable, transmisiones de televisión por satélite, transmisiones de vídeo en continuo, por ejemplo, mediante Internet, codificación de vídeo digital para su almacenamiento en un medio de almacenamiento de datos, decodificación de vídeo digital almacenado en un medio de almacenamiento de datos, u otras aplicaciones. En algunos ejemplos, el sistema 10 puede configurarse para admitir la transmisión de vídeo unidireccional o bidireccional a fin de admitir aplicaciones tales como la transmisión de vídeo, la reproducción de vídeo, la radiodifusión de vídeo y/o la videotelefonía.

[0027] Aunque no se muestra en la FIG. 1, en algunos aspectos, el codificador de vídeo 20 y el decodificador de vídeo 30 pueden estar integrados, cada uno de ellos, con un codificador y un decodificador de audio, y pueden incluir unidades adecuadas MUX-DEMUX, u otro tipo de hardware y software, para gestionar la codificación, tanto de audio como de vídeo, en un flujo de datos común o en flujos de datos diferentes. Si procede, en algunos ejemplos, las unidades MUX-DEMUX pueden ajustarse al protocolo de multiplexado ITU H.223 o a otros protocolos, tales como el protocolo de datagramas de usuario (UDP).

[0028] El codificador de vídeo 20 y el decodificador de vídeo 30 pueden implementarse cada uno como cualquiera de entre una variedad de circuitos de codificadores adecuados, tales como uno o más microprocesadores, procesadores de señales digitales (DSP), circuitos integrados específicos de la aplicación (ASIC), matrices de puertas programables *in situ* (FPGA), lógica discreta, software, hardware, firmware o cualquier combinación de estos. Cuando las técnicas se implementan parcialmente en software, un dispositivo puede almacenar instrucciones para el software en unos medios legibles por ordenador no transitorios adecuados, y ejecutar las instrucciones en hardware, mediante uno o más procesadores, para realizar las técnicas de esta divulgación. Tanto el codificador de vídeo 20 como el decodificador de vídeo 30 pueden estar incluidos en uno o más codificadores o decodificadores, donde cualquiera de ambos puede estar integrado como parte de un codificador/decodificador (CODEC) combinado en un dispositivo respectivo.

[0029] Esta divulgación puede referirse en general al codificador de vídeo 20 que "señaliza" cierta información a otro dispositivo, tal como el decodificador de vídeo 30. El término "señalizar" puede referirse en general a la comunicación de elementos sintácticos y/u otros datos usadospara decodificar los datos de vídeo comprimidos. Dicha comunicación puede producirse en tiempo real o casi real. De forma alternativa, dicha comunicación puede producirse durante un lapso de tiempo, tal como podría ocurrir cuando se almacenan elementos sintácticos en un medio de almacenamiento legible por ordenador en un flujo de bits codificado en el momento de la codificación, que a continuación un dispositivo de decodificación puede recuperar en cualquier momento tras haber sido almacenado en este medio. Por consiguiente, la señalización puede referirse en general a la provisión de información en un flujo de bits codificado para uso en el procesamiento y/o la decodificación del flujo de bits codificado.

[0030] En algunos ejemplos, el codificador de vídeo 20 y el decodificador de vídeo 30 funcionan de acuerdo con una norma de compresión de vídeo, tal como la ISO/IEC MPEG-4 Visual e ITU-T H.264 (también conocida como ISO/IEC MPEG-4 AVC), incluida su ampliación de codificación de vídeo escalable (SVC), ampliación de de codificación de vídeo multivista (MVC) y ampliación 3DV basada en MVC. Además, actualmente se están emprendiendo unas iniciativas para generar una ampliación de codificación de vídeo tridimensional (3DV) para H.264/AVC, en concreto, 3DV basado en AVC o "3D-AVC". Un proyecto conjunto de la ampliación MVC de H.264 se describe en "Advanced video coding for generic audiovisual services", Recomendación ITU-T H.264, marzo de 2010 (en adelante, la especificación H.264/AVC). En otros ejemplos, el codificador de vídeo 20 y el decodificador de vídeo 30 pueden funcionar de acuerdo con la Recomendación ITU-T H.261, ISO/IEC MPEG-1 Visual, ITU-T H.262 o ISO/IEC MPEG-2 Visual y ITU-T H.263, ISO/IEC-4 Visual. Sin embargo, las técnicas de esta divulgación no están limitadas a ninguna norma de codificación particular.

[0031] Como se ha indicado anteriormente, un Equipo de Colaboración Conjunta en Codificación de Vídeo 3D (JCT-3V) de VCEG y MPEG está elaborando actualmente una norma 3DV basada en H.264/AVC, es decir, 3D-AVC. Para 3D-AVC, se han incluido y admitido nuevas herramientas de codificación además de la predicción entre vistas en MVC. Desde el 3 de marzo de 2014, el software para 3D-AVC (es decir, 3D-ATM) se puede descargar en el siguiente enlace: [3D-ATM versión 6.2]: http://mpeg3dv.research.nokia.com/svn/mpeg3dv/tags/3DV-ATMv6.2/. Una versión preliminar de 3D-AVC está disponible al público: Hannuksela *et al*., "3D-AVC draft text 5", JCT3V-C1002, Ginebra, CH, enero de 2013 (en adelante, "3D-AVC draft text 5"). Desde el 3 de marzo de 2014, el proyecto de texto 3D-AVC 5 está disponible en el siguiente enlace: http://phenix.it-sudparis.eu/jct2/doc_end_user/documents/3_Geneva/wg11/JCT3V-C1002-v3.zip.

[0032] En otros ejemplos, el codificador de vídeo 20 y el decodificador de vídeo 30 pueden funcionar de acuerdo con la norma de Codificación de Vídeo de Alta Eficacia (HEVC) elaborada por el Equipo de Colaboración Conjunta en

Codificación de Vídeo (JCT-VC) del Grupo de Expertos en Codificación de Vídeo (VCEG) de ITU-T y el Grupo de Expertos en Imágenes en Movimiento (MPEG) de ISO/IEC o una ampliación de HEVC. En Bross *et al*., "High Efficiency Video Coding (HEVC) text specification draft 9", 11th Meeting: Shanghai, CN, del 10 al 19 de octubre de 2012, documento JCTVC-K1003, se proporciona un borrador de trabajo de la norma HEVC, denominado "HEVC Working Draft 9". Desde el 3 de marzo de 2014, el documento HEVC Working Draft 9 se puede descargar en http://phenix.int-evry.fr/jct/doc_end_user/documents/11_Shanghai/wg11/JCTVC-K1003-v12.zip.

[0033] Además, actualmente se están dedicando esfuerzos a la elaboración de ampliaciones de codificación multivista y 3DV para HEVC. Dicho de otro modo, un Equipo de Colaboración Conjunta en Codificación de Vídeo 3D (JCT-3V) de VCEG y MPEG está elaborando una norma 3DV basada en HEVC, para la cual una parte de los trabajos de normalización incluye la normalización del códec de vídeo multivista basado en HEVC (MV-HEVC) y otra parte, la codificación de vídeo 3D basada en HEVC (3D-HEVC). El codificador de vídeo 20 y el decodificador de vídeo 30 pueden funcionar de acuerdo con dichas ampliaciones a la norma HEVC. La ampliación de codificación multivista de HEVC puede denominarse MV-HEVC. El documento de Tech *et al*., "MV-HEVC Working Draft 1", JCT3V-A1004, Equipo de Colaboración Conjunta sobre Elaboración de Ampliación de Codificación de Vídeo 3D de la ITU-T SG 16 WP 3 e ISO/IEC JTC 1/SC 29/WG 11, 1st Meeting: Estocolmo, SE, 16-20 de julio de 2012 (en adelante, "JCT3V-A1004" o "MV-HEVC Working Draft 1"), proporciona un borrador de trabajo para MV-HEVC. El documento de Tech *et al*., "MV-HEVC Working Draft 2", JCT3V-B1004, Equipo de Colaboración Conjunta sobre Elaboración de Ampliación de Codificación de Vídeo 3D de la ITU-T SG 16 WP 3 e ISO/IEC JTC 1/SC 29/WG 11, 2nd Meeting: Shanghai, CN, 13-19 de octubre de 2012 (en adelante, "MV-HEVC Working Draft 2"), proporciona otro borrador de trabajo para MV-HEVC.

[0034] La ampliación 3DV de HEVC puede denominarse 3D-HEVC. El documento de Tech *et al*., "Draft of 3D-HEVC Test Model Description Draft", JCT3V-B1005, Equipo de Colaboración Conjunta sobre Elaboración de Ampliación de Codificación de Vídeo 3D de la ITU-T SG 16 WP 3 e ISO/IEC JTC 1/SC 29/WG 11, 1st Meeting: Estocolmo, SE, 16-20 de julio de 2012 (en adelante, "3D-HEVC Test Model 1") describe el software de referencia, así como un borrador de trabajo de 3D-HEVC. El documento de Tech *et al*., "3D-HEVC Test Model Description draft 2", JCT3V-B1005, Equipo de Colaboración Conjunta sobre Elaboración de Ampliación de Codificación de Vídeo 3D de la ITU-T SG 16 WP 3 e ISO/IEC JTC 1/SC 29/WG 11, 2n Meeting: Shanghai, CN, Oct. 2012 (en adelante, "3D-HEVC Test Model 2") incluye una descripción de software de referencia, así como otro borrador de trabajo de 3D-HEVC.

[0035] Una secuencia de vídeo incluye típicamente una serie de tramas de vídeo. Un codificador de vídeo 20 actúa típicamente sobre bloques de vídeo de tramas de vídeo individuales con el fin de codificar los datos de vídeo. Esta divulgación puede usar el término "bloque de vídeo" o "bloque" para referirse a uno o más bloques de muestras y estructuras sintácticas usadas para codificar muestras del uno o más bloques de muestras (es decir, "bloques de muestras"). Los ejemplos de tipos de bloques pueden incluir macrobloques, particiones de macrobloques, submacrobloques, unidades de codificación, unidades de predicción, bloques de árbol de codificación, y así sucesivamente.

[0036] Los bloques de muestras de vídeo pueden presentar tamaños fijos o variables y pueden diferir en tamaño de acuerdo con una norma de codificación especificada. Una muestra (es decir, un "valor de muestra") puede indicar una componente de color (por ejemplo, una componente luma o croma) de un píxel. En algunos casos, una muestra puede indicar un valor de profundidad.

[0037] En H.264/AVC, un grupo de imágenes (GOP) comprende en general una serie de una o más tramas de vídeo. Un GOP puede incluir datos sintácticos en una cabecera del GOP, una cabecera de una o más tramas del GOP o en otras ubicaciones, que indican el número de tramas incluidas en el GOP. Cada trama puede incluir datos sintácticos de trama que describen una modalidad de codificación para la respectiva trama.

[0038] Además, en H.264/AVC, un bloque de vídeo puede corresponder a un macrobloque (MB) o una partición de un macrobloque (es decir, una partición de macrobloque). Un MB es un bloque 16×16 de muestras de luma y dos bloques correspondientes de muestras de croma de una imagen que tiene tres matrices de muestras, o un bloque 16×16 de muestras de una imagen monocromática o una imagen que se codifica mediante tres planos de color separados. Una partición de MB es un bloque de muestras de luma y dos bloques correspondientes de muestras de croma resultantes de una partición de un macrobloque para la predicción inter para una imagen que tiene tres matrices de muestras o un bloque de muestras de luma resultante de una partición de un macrobloque para predicción inter de una imagen monocromática o una imagen que se codifica mediante tres planos de color separados. En H.264/AVC, cada macrobloque sometido a predicción inter puede dividirse de cuatro formas diferentes: una partición 16×16 de macrobloque, dos particiones 16×8 de macrobloque, dos particiones 8×16 de macrobloque y cuatro particiones 8×8 de macrobloque. Las diferentes particiones de MB de un MB pueden tener diferentes valores de índice de referencia para cada dirección de predicción (es decir, RefPicList0 o RefPicList1).

[0039] El concepto de macrobloque de H.264/AVC no existe en HEVC. Por el contrario, los macrobloques se reemplazan por una estructura jerárquica sumamente flexible basada en un esquema de árbol cuaternario genérico. Dentro de este sistema, se definen tres tipos de bloques, es decir, unidades de codificación (CU), unidades de predicción (PU) y unidades de transformada (TU). Una CU es una unidad básica de división de zonas. El concepto

de CU es análogo al concepto de macrobloque, pero una CU no está restringida a un tamaño máximo y una CU permite la división recursiva en cuatro CU de igual tamaño para mejorar la adaptabilidad del contenido. Una PU es una unidad básica de predicción inter/intra. En algunos ejemplos, una PU puede contener varias particiones de forma arbitraria en una sola PU para codificar eficazmente patrones de imagen irregulares. Una TU es una unidad básica de transformada. Las TU de una CU pueden definirse independientemente de las PU de la CU. Sin embargo, el tamaño de una TU está limitado a la CU a la que pertenece la TU. Esta separación de la estructura de bloques en tres conceptos diferentes puede permitir optimizar cada uno de ellos de acuerdo con su función, lo que puede resultar en una mejora de la eficacia de codificación.

[0040] Para generar una representación codificada de una imagen en HEVC, el codificador de vídeo 20 puede generar un conjunto de unidades de árbol de codificación (CTU). Una CTU también puede denominarse "bloque de árbol" o una "unidad de codificación más grande" (LCU). Cada una de las CTU puede comprender un bloque de árbol de codificación de muestras de luma, dos bloques de árbol de codificación correspondientes de muestras de croma y estructuras sintácticas usadas para codificar las muestras de los bloques de árbol de codificación. En imágenes monocromáticas o imágenes que tienen tres planos de color separados, una CTU puede comprender un solo bloque de árbol de codificación y estructuras sintácticas usadas para codificar las muestras del bloque de árbol de codificación. Un bloque de árbol de codificación puede ser un bloque N×N de muestras. Por lo tanto, un modelo de prueba HEVC indica que una trama o imagen de vídeo puede dividirse en una secuencia de bloques de árbol o LCU que incluyen muestras tanto de luma como de croma. Las CTU de HEVC pueden ser análogas en gran medida a los macrobloques de otras normas, tales como la H.264/AVC. Sin embargo, una CTU no está necesariamente limitada a un tamaño particular y puede incluir una o más CU.

[0041] Para generar una CTU codificada en HEVC, el codificador de vídeo 20 puede llevar a cabo de forma recursiva una partición de árbol cuaternario en los bloques de árbol de codificación de una CTU para dividir los bloques de árbol de codificación en bloques de codificación, de ahí el nombre "unidades de árbol de codificación." En otras palabras, cada bloque de árbol de codificación puede dividirse en unas CU de acuerdo con un árbol cuaternario. Por ejemplo, un bloque de árbol de codificación, como nodo raíz del árbol cuaternario, puede separarse en cuatro nodos hijo, y cada nodo hijo puede a su vez ser un nodo padre y dividirse en otros cuatro nodos hijo. Un nodo hijo final, no separado, como nodo hoja del árbol cuaternario, comprende un nodo de codificación, es decir, un bloque de vídeo codificado. Los datos sintácticos asociados a un flujo de bits codificado pueden definir un número máximo de veces que puede separarse un bloque de árbol, y también pueden definir un tamaño mínimo de los nodos de codificación.

[0042] Un bloque de codificación es un bloque N×N de muestras. Una CU puede comprender un bloque de codificación de muestras de luma y dos bloques de codificación correspondientes de muestras de croma de una imagen que tiene una matriz de muestras de luma, una matriz de muestras de Cb y una matriz de muestras de Cr y estructuras sintácticas usadas para codificar las muestras de los bloques de codificación. En imágenes monocromáticas o imágenes que tienen tres planos de color separados, una CU puede comprender un único bloque de codificación y estructuras sintácticas usadas para codificar las muestras del bloque de codificación. Un tamaño de la CU corresponde, en general, a un tamaño de un bloque de codificación de la CU y típicamente es de forma cuadrada. El tamaño de la CU puede variar desde 8×8 píxeles hasta el tamaño de una CTU con un tamaño máximo de 64×64 píxeles o más.

[0043] El codificador de vídeo 20 puede dividir un bloque de codificación de una CU en uno o más bloques de predicción. En general, una PU incluye datos relacionados con el proceso de predicción. Una PU de una CU puede comprender un bloque de predicción de muestras de luma, dos bloques de predicción correspondientes de muestras de croma y estructuras sintácticas usadas para predecir los bloques de predicción. En imágenes monocromáticas o imágenes que tienen tres planos de color separados, una PU puede comprender un solo bloque de predicción y estructuras sintácticas usadas para predecir el bloque de predicción. Un bloque de predicción puede ser un bloque rectangular (por ejemplo M×N, donde M puede ser o no igual a N) de muestras a las que se aplica la misma predicción. Las PU pueden dividirse para no tener forma cuadrada. En el contexto de HEVC y otras normas de codificación de vídeo, los términos "bloque de vídeo" o "bloque" pueden aplicarse a una LCU, una CU o una PU.

[0044] En un ejemplo, los modelos de prueba para HEVC admiten la predicción en diversos tamaños de PU. Suponiendo que el tamaño de una CU particular sea 2N×2N, los modelos de prueba para HEVC admiten la predicción intra en tamaños de PU de 2N×2N o N×N, y la predicción inter en tamaños de PU simétricos de 2N×2N, 2N×N, N×2N o N×N. Los modelos de prueba para HEVC también admiten la división asimétrica para la predicción inter en tamaños de PU de 2N×nU, 2N×nD, nL×2N y nR×2N. En la división asimétrica, una dirección de una CU no está dividida, mientras que la otra dirección está dividida en 25% y 75%. La parte de la CU correspondiente a la división de 25% está indicada por una “n” seguida de una indicación “arriba”, “abajo”, “izquierda” o “derecha”. Así, por ejemplo, “2N×nU” se refiere a una CU de tamaño 2N×2N que está dividida horizontalmente con una PU 2N×0,5N encima y una PU 2N×1,5N debajo.

[0045] En esta divulgación, "N×N" y "N por N" pueden usarse de manera intercambiable para hacer referencia a las dimensiones de píxeles de un bloque de vídeo, en términos de dimensiones verticales y horizontales, por ejemplo,

16×16 píxeles o 16 por 16 píxeles. En general, un bloque 16×16 tendrá 16 píxeles en una dirección vertical (y = 16) y 16 píxeles en una dirección horizontal (x = 16). Asimismo, un bloque N×N tiene en general N píxeles en una dirección vertical y N píxeles en una dirección horizontal, donde N representa un valor entero no negativo. Los píxeles de un bloque pueden estar ordenados en filas y columnas. Además, no es necesario que los bloques presenten necesariamente el mismo número de píxeles en la dirección horizontal y en la dirección vertical. Por ejemplo, los bloques pueden comprender N×M píxeles, donde M no es necesariamente igual a N.

[0046] El codificador de vídeo 20 puede generar bloques predictivos (por ejemplo, bloques de luma, Cb y Cr) para bloques de predicción (por ejemplo, bloques de predicción de luma, Cb y Cr) de cada PU de la CU. Por lo tanto, en esta divulgación, se puede decir que una CU se divide en una o más PU. Para facilitar la explicación, esta divulgación puede referirse al tamaño de un bloque de predicción de una PU simplemente como el tamaño de la PU. Los datos sintácticos asociados a una CU pueden describir, por ejemplo, la división de la CU en una o más PU. Los modos de división pueden diferir dependiendo de si la CU está codificada en modo de salto o directo, codificada en modo de predicción intra o codificada en modo de predicción inter.

[0047] En algunos ejemplos, cuando un bloque de vídeo, tal como un MB, una partición MB, una CU, una PU, etc., se codifica en modo de salto, no se señaliza ningún dato residual para el bloque de vídeo. Por ejemplo, en H.264/AVC, un MB saltado es un MB para el que no se codifica ningún dato que no sea una indicación de que el MB debe decodificarse como "saltado". En algunos ejemplos, cuando se codifica un bloque de vídeo mediante el modo directo, no se codifica ningún vector de movimiento para el bloque de vídeo. Por ejemplo, en H.264/AVC, la predicción directa es una predicción inter para un bloque (es decir, una matriz M×N de muestras) de las que no se decodifica ningún vector de movimiento.

[0048] Una trama o imagen de vídeo puede dividirse en uno o más segmentos. Un segmento puede incluir un número entero de bloques de vídeo ordenados consecutivamente por orden de codificación, tal como un orden de exploración por trama. En H.264/AVC, un segmento puede incluir un número entero de macrobloques ordenados consecutivamente en un orden de codificación. En HEVC, un segmento puede incluir un número entero de CTU ordenadas consecutivamente por orden de codificación. Cada segmento de una imagen puede incluir datos sintácticos de segmento que describen un modo de codificación para el segmento respectivo. Un segmento codificado puede comprender una cabecera de segmento y datos de segmento. La cabecera de segmento de un segmento puede ser una estructura sintáctica que incluye elementos sintácticos que proporcionan información sobre el segmento. Los datos del segmento pueden incluir bloques de vídeo codificado del segmento. Un codificador de vídeo 20 actúa típicamente sobre bloques de vídeo de segmentos de individuales (es decir, "segmentos de vídeo" con el fin de codificar los datos de vídeo.

[0049] Las normas de codificación de vídeo definen varios tipos de segmentos. Por ejemplo, H.264/AVC y HEVC definen segmentos I, segmentos P y segmentos B. En general, un segmento I es un segmento que se decodifica mediante predicción intra solamente. En general, un segmento P es un segmento que puede decodificarse mediante predicción intra o predicción inter usando como máximo un vector de movimiento y un índice de referencia para predecir los valores de muestra de cada bloque. En general, un segmento B (es decir, un segmento predictivo b) es un segmento que se puede decodificar mediante predicción intra o predicción inter usando como máximo dos vectores de movimiento e índices de referencia para predecir los valores de muestra de cada bloque. Además, la norma H.264/AVC define tipos de segmentos adicionales, tales como segmentos SI y segmentos SP. Un segmento SI es un segmento que se codifica mediante predicción intra solo y mediante cuantificación de las muestras de predicción. Por lo tanto, en H.264/AVC, un segmento I es un segmento que no es un segmento SI que se decodifica mediante predicción intra solamente. Un segmento SP es un segmento que puede codificarse mediante predicción intra o predicción inter con cuantificación de las muestras de predicción usando como máximo un vector de movimiento e índice de referencia para predecir los valores de muestra de cada bloque. Por lo tanto, en H.264/AVC, un segmento P es un segmento que no es un segmento SP que puede decodificarse mediante predicción intra o predicción inter usando como máximo un vector de movimiento y un índice de referencia para predecir los valores de muestra de cada bloque.

[0050] Cuando el codificador de vídeo 20 codifica un bloque de vídeo actual, el codificador de vídeo 20 puede generar bloques predictivos que se corresponden con el bloque de vídeo actual. El codificador de vídeo 20 puede realizar predicción intra o predicción inter para generar los bloques predictivos. Por ejemplo, el codificador de vídeo 20 puede codificar macrobloques mediante predicción inter o predicción intra. Cuando el codificador de vídeo 20 realiza la predicción intra para un bloque de vídeo actual, el codificador de vídeo 20 puede generar, basándose en muestras de la misma imagen que el bloque de vídeo actual, bloques predictivos para el bloque de vídeo actual. Por ejemplo, cuando el codificador de vídeo 20 codifica un macrobloque mediante predicción intra, el codificador de vídeo 20 puede generar uno o más bloques predictivos para el macrobloque basándose en muestras de la imagen actual. Un macrobloque codificado mediante predicción intra puede denominarse macrobloque intra. Mientras que la norma H.264 proporciona nueve modos de codificación de predicción intra, la norma HEVC puede proporcionar hasta treinta y tres modos de codificación de predicción intra. Cuando la PU está codificada en modalidad intra, la PU puede incluir datos que describen un modo de predicción intra para la PU.

[0051] Cuando el codificador de vídeo 20 realiza predicción inter para generar bloques predictivos para un bloque de

vídeo actual, el codificador de vídeo 20 puede generar los bloques predictivos basándose en muestras de una o más imágenes de referencia. Las imágenes de referencia pueden ser imágenes distintas de la imagen que contiene el bloque de vídeo actual.

[0052] En el contexto de H.264/AVC, cuando el codificador de vídeo 20 codifica un macrobloque mediante predicción inter, el codificador de vídeo 20 genera uno o más bloques predictivos para el macrobloque basándose en muestras de una o más imágenes distintas de la imagen actual (es decir, la imagen que contiene el macrobloque). Un macrobloque codificado mediante predicción inter puede denominarse macrobloque inter.

[0053] En el contexto de HEVC, cuando la PU está codificada en modo inter, la PU puede incluir datos que definen un vector de movimiento para la PU. Los datos que definen el vector de movimiento para una PU pueden describir, por ejemplo, una componente horizontal del vector de movimiento, una componente vertical del vector de movimiento, una resolución para el vector de movimiento (por ejemplo, precisión de un cuarto de píxel o precisión de un octavo de píxel), una imagen de referencia a la que apunta el vector de movimiento y/o una lista de imágenes de referencia (por ejemplo, la Lista 0, la Lista 1 o la Lista C) para el vector de movimiento, que puede indicarse mediante una dirección de predicción.

[0054] Una vez que el codificador de vídeo 20 ha generado bloques predictivos para un bloque de vídeo actual, el codificador de vídeo 20 puede generar bloques residuales para el bloque de vídeo actual. Cada muestra de un bloque residual puede estar basada en una diferencia entre unas muestras correspondientes de un bloque de luma o croma del bloque de vídeo actual y un bloque predictivo para el bloque de vídeo actual. Por ejemplo, en el contexto de HEVC, los datos residuales pueden corresponder a diferencias de píxeles entre píxeles de la imagen no codificada y valores de predicción correspondientes a las CU. El codificador de vídeo 20 puede aplicar una transformada a unas muestras de un bloque residual para generar un bloque de coeficientes de transformada. El codificador de vídeo 20 puede aplicar varias transformadas al bloque residual. Por ejemplo, en el contexto de HEVC, tras la codificación intrapredictiva o interpredictiva mediante las PU de una CU, el codificador de vídeo 20 puede calcular datos residuales a los que se aplican las transformadas especificadas por las TU de la CU. Dicho de otro modo, el codificador de vídeo 20 puede formar los datos residuales para la CU, y a continuación transformar los datos residuales para generar coeficientes de transformada. En algunos ejemplos, el codificador de vídeo 20 puede aplicar una transformada tal como una transformada de coseno discreta (DCT), una transformada de número entero, una transformada de wavelet o una transformada conceptualmente similar al bloque residual.

[0055] El codificador de vídeo 20 puede cuantificar los bloques de coeficientes de transformada para reducir aún más el número de bits utilizados para representar el bloque de vídeo actual. Tras cualquier transformada para generar coeficientes de transformada, el codificador de vídeo 20 puede realizar la cuantificación de los coeficientes de transformada. Cuantificación se refiere en general a un proceso en el que los coeficientes de transformada se cuantifican para reducir posiblemente la cantidad de datos usados para representar los coeficientes y proporcionar compresión adicional. El proceso de cuantificación puede reducir la profundidad de bits asociada a algunos o la totalidad de los coeficientes. Por ejemplo, un valor de n bits puede redondearse a la baja hasta un valor de m bits durante la cuantificación, donde n es mayor que m.

[0056] En algunos ejemplos, el codificador de vídeo 20 puede, tras cuantificar un bloque de coeficientes de transformada, explorar los coeficientes de transformada cuantificados para formar un vector de 1 dimensión (es decir, un vector en serie) de las componentes de transformada cuantificados. En algunos ejemplos, el codificador de vídeo 20 puede usar un orden de exploración predefinido para explorar los coeficientes de transformada cuantificados y generar un vector en serie. En otros ejemplos, el codificador de vídeo 20 puede realizar una exploración adaptativa. Los elementos sintácticos que representan los coeficientes de transformada cuantificados pueden estar sometidos a codificados de entropía. En otras palabras, después de explorar los coeficientes de transformada cuantificados para formar un vector unidimensional, el codificador de vídeo 20 puede realizar la codificación de entropía del vector unidimensional.

[0057] La norma HEVC admite unas transformadas de acuerdo con las TU, que pueden ser diferentes para diferentes CU. En general, se usa una TU para los procesos de transformada y cuantificación. Una CU dada que tiene una o más PU también puede incluir una o más TU. Después de la predicción, el codificador de vídeo 20 puede calcular unos valores residuales a partir del bloque de vídeo identificado por el nodo de codificación de acuerdo con la PU. El nodo de codificación se actualiza a continuación para hacer referencia a los valores residuales, en lugar del bloque de vídeo original. Los valores residuales pueden comprender valores de diferencias de píxeles que pueden transformarse en coeficientes de transformada, cuantificarse y explorarse usando las transformadas y otra información de transformada especificada en las TU, para generar coeficientes de transformada en serie para la codificación de entropía. El nodo de codificación puede, una vez más, actualizarse para referirse a estos coeficientes de transformada en serie.

[0058] El codificador de vídeo 20 puede realizar la codificación de entropía de elementos sintácticos que representan coeficientes de transformada en el bloque de coeficientes de transformada y otros elementos sintácticos asociados con el bloque de vídeo actual. Por ejemplo, el codificador de vídeo 20 puede realizar la codificación aritmética binaria adaptativa al contexto (CABAC), codificación de longitud variable adaptativa al contexto (CAVLC),

codificación exponencial de Golomb, codificación aritmética binaria adaptativa al contexto y basada en sintaxis (SBAC), entropía de división de intervalo de probabilidad (PIPE) u otro tipo de codificación de entropía de los elementos sintácticos. El codificador de vídeo 20 puede generar un flujo de bits que incluye los elementos sintácticos sometidos a codificación de entropía y otros elementos sintácticos asociados con el bloque de vídeo actual.

[0059] En HEVC, el codificador de vídeo 20 puede utilizar la división en árbol cuaternario para descomponer el uno o más bloques residuales de una CU (por ejemplo, bloques residuales de luma, Cb y Cr de la CU) en uno o más bloques de transformada (por ejemplo, bloques de transformada de luma, Cb y Cr). Un bloque de transformada es un bloque rectangular (por ejemplo, cuadrado o no cuadrado) de muestras a las que se aplica la misma transformada. Una TU de una CU puede comprender un bloque de transformada de muestras de luma, dos bloques de transformada correspondientes de muestras de croma y estructuras sintácticas utilizadas para transformar las muestras de bloques de transformada. De este modo, cada TU de una CU puede estar asociada con un bloque de transformada de luma, un bloque de transformada de Cb y un bloque de transformada de Cr. El bloque de transformada de luma asociado con la TU puede ser un subbloque del bloque residual luma de la CU. El bloque de transformada de Cb puede ser un subbloque del bloque residual de Cb de la CU. El bloque de transformada de Cr puede ser un subbloque del bloque residual de Cr de la CU. En imágenes monocromáticas o imágenes que tienen tres planos de color separados, una TU puede comprender un solo bloque de transformada y unas estructuras sintácticas usadas para transformar las muestras del bloque de transformada. De este modo, las muestras residuales correspondientes a una CU pueden subdividirse en unidades más pequeñas mediante una estructura de árbol cuaternario conocida como "árbol cuaternario residual" (RQT). Los nodos hoja del RQT pueden denominarse TU. Los datos sintácticos asociados a una CU también pueden describir, por ejemplo, la división de la CU en una o más TU de acuerdo con un árbol cuaternario.

[0060] En algunos ejemplos, las TU de una CU definida para una LCU dividida se forman con un tamaño basado en el tamaño de las PU de la CU, aunque esto puede no ser siempre así. En algunos ejemplos, las TU son típicamente del mismo tamaño que las PU o más pequeñas.

[0061] Además, en HEVC, el codificador de vídeo 20 puede aplicar una o más transformadas a un bloque de transformada de una TU para generar un bloque de coeficientes para la TU. Un bloque de coeficientes puede ser una matriz bidimensional de coeficientes de transformada. Por ejemplo, el codificador de vídeo 20 puede aplicar una o más transformadas a un bloque de transformada de luma de una TU a fin de generar un bloque de coeficientes de luma para la TU. Un coeficiente de transformada puede ser una cantidad escalar. El codificador de vídeo 20 puede aplicar una o más transformadas a un bloque de transformada de Cb de una TU para generar un bloque de coeficientes para la TU. El codificador de vídeo 20 puede aplicar una o más transformadas a un bloque de transformada de Cr de una TU para generar un bloque de coeficientes de Cr para la TU. De esta forma, los valores de diferencias de píxeles asociados a las TU pueden transformarse para generar coeficientes de transformada, que pueden cuantificarse.

[0062] El flujo de bits puede incluir una secuencia de bits que forma una representación de imágenes codificadas y datos asociados. El flujo de bits puede comprender una secuencia de unidades de capa de abstracción de red (NAL). Cada una de las unidades NAL incluye una cabecera de unidad NAL y encapsula una carga útil de secuencia de bytes sin procesar (RBSP). La cabecera de la unidad NAL puede incluir un elemento sintáctico que indica un código de tipo de unidad NAL. El código de tipo de unidad NAL especificado por la cabecera de unidad NAL de una unidad NAL indica el tipo de la unidad NAL. Una RBSP puede ser una estructura sintáctica que contiene un número entero de bytes que se encapsula dentro de una unidad NAL. En algunos casos, una RBSP incluye cero bits.

[0063] Diferentes tipos de unidades NAL pueden encapsular diferentes tipos de RBSP. Por ejemplo, un primer tipo de unidad NAL puede encapsular una RBSP para un conjunto de parámetros de imagen (PPS), un segundo tipo de unidad NAL puede encapsular una RBSP para un segmento codificado, un tercer tipo de unidad NAL puede encapsular una RBSP para información de realce complementaria (SEI), y así sucesivamente. Las unidades NAL que encapsulan las RBSP para datos de codificación de vídeo (a diferencia de las RBSP para conjuntos de parámetros y mensajes SEI) pueden denominarse unidades NAL de capa de codificación de vídeo (VCL). Una unidad NAL que encapsula un segmento codificado puede denominarse unidad NAL de segmento codificado.

[0064] El decodificador de vídeo 30 puede recibir un flujo de bits que incluye una representación codificada de datos de vídeo. Por ejemplo, el decodificador de vídeo 30 puede analizar el flujo de bits para extraer elementos sintácticos del flujo de bits. Como parte de la extracción de los elementos sintácticos del flujo de bits, el decodificador de vídeo 30 puede realizar la decodificación de entropía de partes del flujo de bits. El decodificador de vídeo 30 puede realizar, basándose al menos en parte en los elementos sintácticos asociados con un bloque de vídeo actual (por ejemplo, un MB o una partición de MB, etc.), la predicción inter o intra para generar bloques predictivos. Además, el decodificador de vídeo 30 puede realizar la cuantificación inversa de los coeficientes de transformada de los bloques de coeficientes de transformada y puede aplicar una o más transformadas inversas a los bloques de coeficientes de transformada para generar bloques residuales. El decodificador de vídeo 30 puede entonces reconstruir los bloques de luma y croma del bloque de vídeo actual basándose, al menos en parte, en los bloques residuales y los bloques predictivos. De este modo, mediante la reconstrucción de los bloques de luma y croma de cada bloque de vídeo de una imagen, el decodificador de vídeo 30 puede reconstruir la imagen.

[0065] Como se ha mencionado anteriormente, el codificador de vídeo 20 puede llevar a cabo la predicción inter para generar bloques predictivos para un bloque de vídeo particular. Más específicamente, el codificador de vídeo 20 puede realizar una predicción inter unidireccional o una predicción inter bidireccional para generar los bloques predictivos.

[0066] Cuando el codificador de vídeo 20 realiza la predicción inter unidireccional para un bloque de vídeo actual (por ejemplo, un MB, una una partición de MB, una PU, etc.), el codificador de vídeo 20 puede buscar un bloque de referencia en las imágenes de referencia de una sola lista de imágenes de referencia (por ejemplo, "Lista 0" o "RefPicList0"). El bloque de referencia puede ser un bloque de muestras de luma y unos bloques correspondientes de muestras de croma que son similares a los bloques de luma y croma del bloque de vídeo actual. Además, cuando el codificador de vídeo 20 realiza una predicción inter unidireccional, el codificador de vídeo 20 puede generar información de movimiento para el bloque de vídeo particular. La información de movimiento para el bloque de vídeo particular puede incluir un vector de movimiento y un índice de referencia. El vector de movimiento puede indicar un desplazamiento espacial entre una posición de la imagen actual de los bloques de muestras del bloque de vídeo actual y una posición de la imagen de referencia del bloque de referencia. El índice de referencia indica una posición de la lista de imágenes de referencia de la imagen de referencia que contiene el bloque de referencia. Las muestras de los bloques predictivos para el bloque de vídeo actual pueden ser iguales a unas muestras correspondientes del bloque de referencia.

[0067] Cuando el codificador de vídeo 20 realiza la predicción inter bidireccional para un bloque de vídeo actual (por ejemplo, un MB, una partición de MB, una PU, etc.), el codificador de vídeo 20 puede buscar un primer bloque de referencia en las imágenes de referencia de una primera lista de imágenes de referencia ("lista 0" o "RefPicList0") y puede buscar un segundo bloque de referencia en las imágenes de referencia de una segunda lista de imágenes de referencia ("lista 1" o "RefPicList1"). El codificador de vídeo 20 puede generar, basándose al menos en parte en el primer y el segundo bloques de referencia, los bloques predictivos para el bloque de vídeo actual. Además, el codificador de vídeo 20 puede generar un primer vector de movimiento que indica un desplazamiento espacial entre un bloque de muestras del bloque de vídeo actual y el primer bloque de referencia. El codificador de vídeo 20 también puede generar un primer índice de referencia que identifica una ubicación de la primera lista de imágenes de referencia de la imagen de referencia que contiene el primer bloque de referencia. Además, el codificador de vídeo 20 puede generar un segundo vector de movimiento que indica un desplazamiento espacial entre los bloques del bloque de vídeo actual y el segundo bloque de referencia. El codificador de vídeo 20 también puede generar un segundo índice de referencia que identifica una ubicación de la segunda lista de imágenes de referencia de la imagen de referencia que incluye el segundo bloque de referencia.

[0068] Cuando el codificador de vídeo 20 realiza la predicción inter unidireccional para un bloque de vídeo actual (por ejemplo, un MB, una partición de MB, una PU,etc.), el decodificador de vídeo 30 puede utilizar la información de movimiento del bloque de vídeo actual para identificar el bloque la referencia del bloque de vídeo actual. El decodificador de vídeo 30 puede generar entonces los bloques predictivos para el bloque de vídeo actual basándose en el bloque de referencia del bloque de vídeo actual. Cuando el codificador de vídeo 20 realiza la predicción inter bidireccional para el bloque de vídeo actual, el decodificador de vídeo 30 puede utilizar la información de movimiento para el bloque de vídeo actual para identificar los dos bloques de referencia del bloque de vídeo actual. El decodificador de vídeo 30 puede generar los bloques predictivos del bloque de vídeo actual basándose en los dos bloques de referencia del bloque de vídeo actual.

[0069] En H.264/AVC, cuando un MB no está dividido en cuatro particiones MB 8x8, el MB puede tener solo un vector de movimiento para toda la partición de MB en cada dirección de predicción (es decir, RefPicList0 o RefPicList1). Cuando un MB se divide en cuatro particiones de MB 8×8, cada partición de MB 8×8 puede dividirse además en subbloques. Hay cuatro formas diferentes de dividir subbloques a partir de una partición de MB 8×8: un subbloque 8×8, dos subbloques 8×4, dos subbloques 4×8 o cuatro subbloques 4×4. Cada subbloque puede tener un vector de movimiento diferente en cada dirección de predicción. La manera en que una partición de MB 8×8 MB se divide en subbloques puede llamarse partición en subbloques.

[0070] La codificación de vídeo multivista (MVC) es una ampliación de la norma H.264/AVC. En la codificación multivista, puede haber varias vistas de la misma escena desde diferentes puntos de vista. En el contexto de la codificación multivista, el término "unidad de acceso" puede referirse a un conjunto de imágenes que corresponden a la misma instancia de tiempo. Así, los datos de vídeo pueden conceptualizarse como una serie de unidades de acceso que se producen a lo largo del tiempo.

[0071] En la codificación multivista, un flujo de bits puede tener una pluralidad de capas. Cada una de las capas puede corresponder a una vista diferente. En la codificación multivista, una vista puede denominarse "vista básica" si un decodificador de vídeo (por ejemplo, decodificador de vídeo 30) puede decodificar imágenes de la vista sin referencia a imágenes de ninguna otra vista. Una vista puede denominarse vista no básica si la decodificación de la vista depende de la decodificación de las imágenes de una o más vistas diferentes.

[0072] La FIG. 2 es un diagrama conceptual que ilustra un ejemplo de estructura de predicción para codificación

multivista. Una estructura de predicción MVC típica (que incluye tanto la predicción entre imágenes dentro de cada vista como la predicción entre vistas) para la codificación de vídeo multivista se muestra en la FIG. 2, donde las predicciones se indican mediante flechas, donde el objeto al que se apunta utiliza el objeto desde el que se apunta como referencia de predicción.

[0073] La estructura de predicción multivista de la FIG. 2 incluye predicción temporal y entre vistas. En el ejemplo de la FIG. 2 cada cuadrado corresponde a una componente de vista. En el ejemplo de la FIG. 2, las unidades de acceso están etiquetadas como T0 ... T11 y las vistas están etiquetadas como S0 ... S7. Los cuadrados etiquetados como "I" son componentes de vista sometidas a predicción intra. Los cuadrados etiquetados como "P" son componentes de vista sometidas a predicción intra unidireccional. Los cuadrados etiquetados "B" y "b" son componentes de vista sometidas a predicción intra bidireccional. Los cuadrados etiquetados como "b" pueden usar cuadrados etiquetados como "B" como imágenes de referencia. Una flecha que apunta desde un primer cuadrado a un segundo cuadrado indica que el primer cuadrado está disponible en la predicción inter como imagen de referencia para el segundo cuadrado. Como se indica mediante las flechas verticales en la FIG. 2, las componentes de vista de diferentes vistas de la misma unidad de acceso pueden estar disponibles como imágenes de referencia. El uso de una componente de vista de una unidad de acceso como imagen de referencia para otra componente de vista de la misma unidad de acceso puede denominarse predicción entre vistas.

[0074] En MVC, la predicción entre vistas se realiza entre las imágenes captadas en las diferentes vistas de la misma unidad de acceso (es decir, en la misma instancia de tiempo) para eliminar la correlación entre vistas. Una imagen codificada con predicción entre vistas se puede añadir a una lista de imágenes de referencia para la predicción entre vistas de las otras vistas no básicas. Una imagen de referencia de predicción entre vistas puede colocarse en cualquier posición de una lista de imágenes de referencia, de la misma forma que con una imagen de referencia de predicción inter.

[0075] En el contexto de la codificación de vídeo multivista, hay dos tipos de vectores de movimiento. Un tipo de vector de movimiento es un vector de movimiento normal que apunta a una imagen de referencia temporal (es decir, una imagen en una instancia de tiempo diferente de una imagen actual). El tipo de predicción inter correspondiente a un vector de movimiento temporal normal puede denominarse "predicción con compensación de movimiento" o "MCP". Cuando se utiliza una imagen de referencia de predicción entre vistas para la compensación de movimiento, el vector de movimiento correspondiente puede denominarse "vector de movimiento de disparidad". En otras palabras, un vector de movimiento de disparidad apunta a una imagen de una vista diferente (es decir, una imagen de referencia de disparidad o una imagen de referencia entre vistas). El tipo de predicción inter correspondiente a un vector de movimiento de disparidad puede denominarse "predicción con compensación de disparidad" o "DCP".

[0076] La FIG. 3 es un diagrama conceptual que ilustra un ejemplo de orden de decodificación multivista. Dicho de otro modo, se muestra un orden de decodificación de MVC típico (es decir, un orden de flujo de bits) en la FIG. 3. La disposición de orden de decodificación se denomina codificación de tiempo primero. Debe tenerse en cuenta que el orden de decodificación de las unidades de acceso puede no ser idéntico al orden de salida o de visualización. En la FIG. 3, cada una de S0-S7 se refiere a una vista diferente del vídeo multivista. Cada una de T1-T9 representa una instancia de tiempo de salida. Una unidad de acceso puede incluir las imágenes codificadas de todas las vistas para una instancia de tiempo de salida. Por ejemplo, una primera unidad de acceso puede incluir todas las vistas S0-S7 para la instancia de tiempo T1, una segunda unidad de acceso puede incluir todas las vistas S0-S7 para la instancia de tiempo T2, etcétera.

[0077] En la siguiente sección se analiza la codificación de vídeo multivista y 3D en general. En el contexto de la codificación de vídeo multivista y 3D, una "componente de vista" puede ser una representación codificada de una vista en una unidad de acceso único. Cuando una vista incluye tanto representaciones de textura como de profundidad codificadas, una componente de vista puede comprender (por ejemplo, consistir en) una componente de vista de textura y una componente de vista de profundidad. Por lo tanto, cada componente de vista de textura puede tener una componente de vista de profundidad correspondiente. En general, las componentes de vista de textura incluyen contenido de vídeo (por ejemplo, componentes de luma y croma de valores de píxeles), y las componentes de vista de profundidad pueden indicar profundidades relativas de los píxeles en las componentes de vista de textura. De este modo, la pluralidad de imágenes de vídeo para cada vista pueden denominarse componentes de vista de textura. En esta divulgación, una "vista" puede referirse a una secuencia de componentes de vista asociadas con el mismo identificador de vista.

[0078] Más específicamente, una componente de vista de la textura (es decir, una imagen de textura) puede ser una representación codificada de la textura de una vista en una única unidad de acceso. La componente de vista de textura incluye el contenido de imagen real que se muestra. Por ejemplo, la componente de vista de textura puede incluir las componentes de luma (Y) y croma (Cb y Cr). Una vista de textura puede ser una secuencia de componentes de vista de textura asociadas con un valor idéntico de índice de orden de vista. Un índice de orden de vista de una vista puede indicar una posición de cámara de la vista con respecto a otras vistas.

[0079] Las técnicas de la presente divulgación se refieren a la codificación de datos de vídeo 3D mediante la codificación de datos de textura y profundidad. En general, el término "textura" se usa para describir valores de

luminancia (es decir, brillo o "luma") de una imagen y valores de crominancia (es decir, color o "croma") de la imagen. En algunos ejemplos, una imagen de textura puede incluir un conjunto de datos de luminancia y dos conjuntos de datos de crominancia, para matices azules (Cb) y matices rojos (Cr). En ciertos formatos de muestreo de croma, tales como 4:2:2 o 4:2:0, los datos de croma se someten a muestreo de frecuencia reducida con respecto a los datos de luma. Es decir, la resolución espacial de los píxeles de crominancia puede ser menor que la resolución espacial de los píxeles de luminancia correspondientes, por ejemplo, un medio o un cuarto de la resolución de luminancia.

[0080] Una componente vista de la profundidad (es decir, una imagen de profundidad) puede ser una representación codificada de la profundidad de una vista en una única unidad de acceso. Una vista de profundidad puede ser una secuencia de componentes de vista de profundidad asociadas con un valor idéntico del índice de orden de vistas. El componente de vista de profundidad puede indicar profundidades relativas de los píxeles en su componente de vista de textura correspondiente. Como un ejemplo, la componente de vista de profundidad es una imagen en escala de grises que incluye únicamente valores de luma. En otras palabras, la componente de vista de profundidad tal vez no transmita ningún contenido de imagen, pero en lugar de eso proporcione una medida de las profundidades relativas de los píxeles en la componente de vista de textura.

[0081] En algunos ejemplos, un píxel blanco puro en la componente de vista de profundidad indica que su correspondiente píxel o píxeles de la componente de vista de textura correspondiente está más cerca de la perspectiva del observador, y un píxel negro puro en la componente de vista de profundidad indica que su correspondiente píxel o píxeles de la componente de vista de textura correspondiente está más alejado de la perspectiva del observador. Los diversos tonos de gris entre negro y blanco indican diferentes niveles de profundidad. Por ejemplo, un píxel gris oscuro en la componente de vista de profundidad puede indicar que su píxel correspondiente de la componente de vista de textura está más alejado que un píxel gris claro de la componente de vista de profundidad. Dado que únicamente es necesaria la escala de grises para identificar la profundidad de los píxeles, la componente de vista de profundidad no necesita incluir componentes de croma, ya que los valores de color para la componente de vista de profundidad pueden no servir para ningún fin. La componente de vista de profundidad que utiliza solo valores de luma (por ejemplo, valores de intensidad) para identificar la profundidad se proporciona con fines ilustrativos y no debe considerarse limitativa. En otros ejemplos, puede utilizarse cualquier técnica para indicar las profundidades relativas de los píxeles en la componente de vista de textura.

[0082] Los datos de profundidad en general describen valores de profundidad para los datos de textura correspondientes. Por ejemplo, una imagen de profundidad puede incluir un conjunto de píxeles de profundidad, cada uno de los cuales describe una profundidad para unos datos de textura correspondientes. Los datos de profundidad pueden usarse para determinar la disparidad horizontal para los datos de textura correspondientes. Por lo tanto, un dispositivo que recibe los datos de textura y profundidad puede visualizar una primera imagen de textura para una vista (por ejemplo, una vista de ojo izquierdo) y puede usar los datos de profundidad para modificar la primera imagen de textura para generar una segunda imagen de textura para la otra vista (por ejemplo, una vista de ojo derecho) compensando valores de píxeles de la primera imagen mediante los valores de disparidad horizontal determinados basándose en los valores de profundidad. En general, la disparidad horizontal (o simplemente "disparidad") describe el desplazamiento espacial horizontal de un píxel en una primera vista hacia un píxel correspondiente de una segunda vista, donde los dos píxeles corresponden a la misma parte del mismo objeto que se representa en la dos vistas.

[0083] En otros ejemplos adicionales, pueden definirse datos de profundidad para los píxeles en una dimensión z perpendicular al plano de imagen, de tal manera que una profundidad asociada a un píxel dado se define en relación con un plano de disparidad cero definido para la imagen. Dicha profundidad puede usarse para crear disparidad horizontal para visualizar el píxel, de manera que el píxel se visualiza de manera diferente para los ojos izquierdo y derecho, dependiendo del valor de profundidad de la dimensión z del píxel con respecto al plano de disparidad cero. El plano de disparidad cero puede cambiar para diferentes partes de una secuencia de vídeo, y la cantidad de profundidad relativa al plano de disparidad cero también puede cambiar. Los píxeles situados en el plano de disparidad cero pueden definirse de forma similar para el ojo izquierdo y el derecho. Los píxeles situados delante del plano de disparidad cero pueden mostrarse en diferentes ubicaciones para el ojo izquierdo y el derecho (por ejemplo, con disparidad horizontal) a fin de crear una percepción de que el píxel parece salir de la imagen en la dirección z perpendicular al plano de imagen. Los píxeles situados detrás el plano de disparidad nula pueden mostrarse ligeramente borrosos, para presentar una ligera percepción de profundidad, o pueden mostrarse en ubicaciones diferentes para el ojo izquierdo y derecho (por ejemplo, con disparidad horizontal que es opuesta a la de los píxeles situados delante del plano de disparidad cero). También pueden usarse muchas otras técnicas para expresar o definir datos de profundidad para una imagen.

[0084] Para cada píxel de la componente de vista de profundidad, puede haber uno o más píxeles correspondiente en la componente de vista de textura. Por ejemplo, si las resoluciones espaciales de la componente de vista de profundidad y la componente de vista de textura son iguales, cada píxel de la componente de vista de profundidad corresponde a un píxel de la componente de vista de textura. Si la resolución espacial de la componente de vista de profundidad es menor que la de la componente de vista de textura, entonces cada píxel en la componente de vista de profundidad corresponde a varios píxeles en la componente de vista de textura. El valor del píxel de la

componente de vista de profundidad puede indicar la profundidad relativa del uno o más píxeles correspondientes en la vista de textura.

[0085] En algunos ejemplos, un codificador de vídeo de señala datos de vídeo para las componentes de vista de textura y los correspondientes componentes de vista de profundidad para cada una de las vistas. Un decodificador de vídeo 30 puede utilizar tanto los datos de vídeo de componentes de vista de textura como las componentes de vista de profundidad para decodificar el contenido de vídeo de las vistas para la visualización. Una pantalla visualiza a continuación el vídeo multivista para generar vídeo 3D.

[0086] Con referencia de nuevo a la FIG. 3, cada una de las vistas incluye conjuntos de imágenes. Por ejemplo, la vista S0 incluye un conjunto de imágenes 0, 8, 16, 24, 32, 40, 48, 56 y 64, la vista S1 incluye el conjunto de imágenes 1, 9, 17, 25, 33, 41, 49, 57 y 65, etc. Cada conjunto incluye dos imágenes: una imagen se denomina componente de vista de textura, y la otra imagen se denomina componente de vista de profundidad. La componente de vista de textura y la componente de vista de profundidad de un conjunto de imágenes de una vista pueden considerarse como correspondientes entre sí. Por ejemplo, la componente de vista de textura de un conjunto de imágenes de una vista se considera correspondiente al componente de vista de profundidad del conjunto de las imágenes de la vista, y viceversa (es decir, la componente de vista de profundidad corresponde a su componente de vista de textura en el conjunto, y viceversa). Como se usa en esta divulgación, una componente de vista de textura que corresponde a una componente de vista de profundidad puede considerarse como la componente de vista de textura y la componente de vista de profundidad que son parte de una misma vista de una única unidad de acceso.

[0087] En la siguiente sección de esta divulgación se analiza la norma de codificación de vídeo 3D basada en AVC (es decir, 3D-AVC). A continuación se analiza un orden de codificación de las componentes de vista en 3D-AVC. La 3D-AVC es compatible con la norma H.264/AVC de tal forma que la parte de textura de la vista básica es completamente decodificable para el decodificador H.264/AVC. Para las componentes de vista mejoradas en 3D-AVC, la profundidad puede codificarse antes de la textura, y una componente de vista de textura puede codificarse basándose en la información de la componente de vista de profundidad, lo que también se conoce como codificación de profundidad primero. Por el contrario, cada componente de vista de textura se codifica antes de las respectivas componentes de vista de profundidad en órdenes de codificación de textura primero. Por ejemplo, los órdenes de codificación de las componentes de vista de textura y profundidad en 3D-AVC se pueden ejemplificar tal como se indica a continuación: en los que T0 y D0, respectivamente, se refieren a las componentes de vista de textura y profundidad de la vista básica, y Ti y Di, respectivamente, se refieren a las componentes de vista de textura y profundidad de la i-ésima vista dependiente. En los siguientes ejemplos hay tres vistas:

- T0 D0 D1 D2 T1 T2: Las vistas básicas (T0 y D0) se codifican con el orden de codificación de textura primero, mientras que la vista dependiente se codifica con el orden de codificación de profundidad primero. Actualmente se utiliza un orden de codificación híbrido en condiciones de prueba comunes de 3D-AVC.
- T0 D0 T1 D1 T2 D2: Todos las componentes de vista se codifican con el orden de codificación de textura primero.

[0088] Si la predicción entre vistas está habilitada para Ti, la vista de la textura de referencia se define como la vista que incluye la imagen de referencia entre vistas, y la vista de profundidad correspondiente se define como la vista de profundidad de referencia que tiene el mismo índice de orden de vista que la vista de textura de referencia.

[0089] Un codificador de vídeo puede utilizar un vector de disparidad (DV) como estimador de la disparidad entre los dos vistas. Debido a que los bloques vecinos comparten casi la misma información de movimiento/disparidad en la codificación de vídeo, el bloque actual puede usar la información de vector de movimiento de bloques vecinos como un buen indicador. A continuación se analiza la derivación del vector de disparidad 3D-AVC a través del mapa de profundidad. Las técnicas para derivar el vector de disparidad pueden variar con cada herramienta de codificación de bajo nivel, pero, comúnmente, los datos de profundidad de las vistas dependientes se emplean para la codificación de la componente de vista de textura debido al orden de codificación de profundidad primero.

[0090] Una predicción entre vistas de síntesis de vistas basada en bloques en bucle (BVSP) y una predicción de vectores de movimiento basada en profundidad (D-MVP) en la 3D-AVC son las herramientas de codificación de bajo nivel, principalmente, mediante el vector de disparidad convertido a partir de los valores de profundidad del mapa de profundidad en la trama dependiente. Normalmente, en el software 3D-AVC, los resultados del proceso de conversión desde el valor del mapa de profundidad real hasta una disparidad con una vista particular se almacenan en tablas de consulta con los parámetros de cámara.

[0091] La BVSP puede habilitarse en 3D-AVC. La BVSP fue propuesta originalmente en el documento de Su *et al.*, "3DV-CE1.a: Block-based View Synthesis Prediction for 3DV-ATM", Equipo de Colaboración Conjunta sobre Elaboración de Ampliación de Codificación de Vídeo 3D de ITU-T SG 16 WP 3 e ISO/IEC JTC 1/SC 29/WG 11, 1st Meeting, Estocolmo, SE, 16-20 de julio de 2012, documento JCT3V-A0107 (en adelante, "JCT3V-A0107") que desde el 3 de marzo de 2014 se puede descargar en el siguiente enlace: http://phenix.it-sudparis.eu/jct2/doc_end_user/documents/1_Stockholm/wg11/JCT3V-A0107-v1.zip.

[0092] La FIG. 4 es un ejemplo de visualización conceptual de BVSP basada en la distorsión regresiva. Con referencia a la FIG. 4, se va a suponer que se utiliza el siguiente orden de codificación: (T0, D0, D1, T1). El componente de textura T0 es una vista básica, y T1 es una vista dependiente codificada con la VSP. Los componentes de mapa de profundidad D0 y D1 son mapas de profundidad respectivos asociados con T0 y T1.

[0093] En la vista dependiente T1, los valores de muestra de Cb del bloque actual se predicen a partir de la zona de referencia R(Cb) que consiste en unos valores de muestra de la vista básica T0. El vector de desplazamiento entre muestras codificadas y de referencia se denomina vector de disparidad derivado entre T1 y T0 a partir de un valor de mapa de profundidad asociado con una muestra de textura codificada actualmente.

[0094] En algunos ejemplos, un codificador de vídeo puede utilizar las siguientes ecuaciones para llevar a cabo un proceso de conversión de un valor de profundidad a un vector de disparidad:

$$Z(Cb(j,i)) = \frac{1}{\frac{d(Cb(j,i))}{255}\cdot\left(\frac{1}{Znear}-\frac{1}{Zfar}\right)+\frac{1}{Zfar}}; \qquad (1)$$

$$D(Cb(j,i)) = \frac{f\cdot b}{Z(Cb(j,i))}; \qquad (2)$$

donde *j* e *i* son coordenadas de espacio locales dentro de Cb, *d*(*Cb*(*j,i*)) es un valor de mapa de profundidad de una imagen de mapa de profundidad de una vista nº 1, Z es el valor de profundidad real de *d*(*Cb*(*j,i*)) y *D* es la componente horizontal de un vector de disparidad derivado con respecto a una vista particular nº 0. Los parámetros *f*, *b, Znear* y *Zfar* son parámetros que especifican la configuración de la cámara, es decir, la distancia focal utilizada (*f*), la separación de la cámara (*b*) entre la vista nº 1 y la vista nº 0, y el rango de profundidad (*Znear*, *Zfar*) representan parámetros de conversión del mapa de profundidad.

[0095] En algunos ejemplos, la componente vertical del vector de disparidad derivado se iguala siempre a 0. En una implementación actual de 3D-AVC (es decir, una implementación 3DV-ATM), las ecuaciones (1) y (2) están precalculadas para cada valor del mapa de profundidad (0...255) y almacenadas como una tabla de consulta. De este modo, un codificador de vídeo puede utilizar la tabla de consulta para convertir valores de profundidad en vectores de disparidad sin calcular las ecuaciones (1) y (2) proporcionadas anteriormente.

[0096] Una cuestión de implementación relacionada con la BVSP implica la indicación de bloques BVSP. En algunos ejemplos, los bloques BVSP se indican como sigue. Un indicador en el nivel de MB indica si un MB actual se ha codificado con el modo de salto/directo convencional o si el MB actual se ha codificado con el modo de salto/directo pero se ha predicho a partir de una componente de referencia sintetizada. Para cada partición de MB (de 16×16 a 8×8), un índice de referencia correspondiente a una lista de imágenes de referencia señala una imagen de referencia de la lista de imágenes de referencia. Cuando un codificador de vídeo utiliza el modo BVSP para codificar una partición de MB, el codificador de vídeo no señala las diferencias de vectores de movimiento (MVD) para la partición de MB, porque no hay ningún vector de movimiento para los bloques sometidos a codificación de BVSP. Cuando el indicador o un índice de referencia indican que una partición de MB se codifica mediante una componente de referencia sintetizada, un codificador de vídeo puede invocar la predicción de una partición como se describe a continuación.

[0097] Otra cuestión de implementación relacionada con la BVSP implica el proceso de derivación de predicción. N×M puede indicar el tamaño de una partición de MB, donde N o M son iguales a 8 ó 16. Si la partición de MB se ha codificado con el modo BVSP, la partición de MB se subdivide además en varias subregiones con el tamaño igual a K×K, donde K puede ser 4×4, 2×2 o 1×1. Para cada subregión de la partición de MB, un codificador de vídeo deriva un vector de disparidad derivado separado. Además, para cada subregión respectiva de la partición de MB, el codificador de vídeo utiliza el vector de disparidad derivado para localizar un bloque correspondiente en la imagen de referencia entre vistas, es decir, R(cb) en la FIG. 4. El codificador de vídeo puede predecir la subregión respectiva a partir del bloque correspondiente para la subregión respectiva. Un ejemplo de BVSP se basa en la distorsión regresiva para bloques de un tamaño de 4×4 (que significa que K es igual a 4.) Los vectores de disparidad derivados no se almacenan para los bloques sometidos a codificación de BVSP, porque no hay herramientas de codificación que utilicen dichos vectores.

[0098] Otra cuestión de implementación implica el proceso de derivación de vectores de disparidad. Cuando se aplica un orden de codificación de profundidad primero, un codificador de vídeo puede obtener el vector de disparidad derivado convirtiendo un valor de profundidad del correspondiente bloque de profundidad en la vista de profundidad no básica correspondiente, como se muestra en la FIG. 4. Se pueden aplicar varias técnicas para seleccionar el valor de profundidad de un bloque de profundidad, tal como el valor de profundidad de la posición central del bloque de profundidad, el valor máximo de todos los valores de profundidad de un bloque de profundidad,

el valor máximo de cuatro píxeles de esquina de un bloque de profundidad y el valor de profundidad del píxel inferior derecho del bloque de profundidad/MB de profundidad. Cuando se aplica el orden de codificación de textura primero, el codificador de vídeo puede inhabilitar los modos de BVSP, porque la vista de profundidad no básica correspondiente no está disponible cuando se decodifica la vista de textura no básica.

[0099] A continuación se analiza la predicción del vector de movimiento basada en la profundidad (D-MVP) en 3D-AVC para los modos inter normales. La D-MVP es un procedimiento de predicción de vectores de movimiento que incorpora datos de mapa de profundidad asociados a la vista actual, que están disponibles gracias al orden de codificación de profundidad primero. Un codificador de vídeo puede aplicar la D-MVP con las componentes de vista de textura en las vistas dependientes (es decir, no básicas).

[0100] En 3D-AVC, el procedimiento D-MVP se incorpora a la predicción de vectores de movimiento basada en la función mediana convencional de H.264/AVC. Específicamente, el tipo de vector de movimiento que se va a predecir (es decir, tanto si es un vector de movimiento temporal o un vector de movimiento de disparidad) se identifica primero de una manera tal que los índices de referencia de los vectores de movimiento de los bloques vecinos se comprueban para conocer el tipo de predicción de movimiento.

[0101] Los bloques vecinos incluyen, por orden, un bloque izquierdo, un bloque superior, un bloque superior derecho y un bloque superior izquierdo con respecto al bloque actual. En algunos ejemplos, un codificador de vídeo solo puede utilizar el vector de movimiento en el bloque superior izquierdo cuando uno de los otros tres bloques vecinos (es decir, el bloque izquierdo, el bloque superior o el bloque superior derecho) no contienen ningún vector de movimiento y, por lo tanto, se consideran no disponibles.

[0102] Posteriormente, si tres bloques vecinos están disponibles, el codificador de vídeo puede emplear los vectores de movimiento en los tres bloques vecinos para la predicción de vectores de movimiento de los vectores de movimiento para el bloque actual. En la predicción temporal, si los vectores de movimiento de los tres bloques vecinos son todos del mismo tipo y tienen todos los mismos índices de referencia, el codificador de vídeo puede usar un filtro de mediana directamente como se describe en H.264/AVC. De lo contrario (si los vectores de movimiento de los tres bloques vecinos pertenecen a tipos diferentes y los tres bloques vecinos tienen diferentes índices de referencia), el codificador de vídeo puede derivar además un vector de movimiento para el bloque actual. Cuando la imagen de referencia actual es una imagen de referencia entre vistas, el codificador de vídeo puede comprobar los tipos de vectores de movimiento y sus índices de referencia en posiciones de bloques vecinos. Si los vectores de movimiento son todos del mismo tipo y tienen los mismos índices de referencia, el codificador de vídeo puede aplicar el filtro de mediana. En ambos casos, si están disponibles menos de tres bloques vecinos, el codificador de vídeo puede derivar además vectores de movimiento para los bloques no disponibles, de modo que tres bloques vecinos pasen a estar disponibles.

[0103] Un vector de movimiento derivado para un bloque vecino puede denominarse vector de movimiento derivado. Para derivar un vector de movimiento de un bloque actual, un codificador de vídeo puede determinar si un vector de movimiento actual (es decir, un vector de movimiento de un bloque vecino) es un vector de movimiento de disparidad, si el vector de movimiento del bloque vecino es de un tipo diferente al tipo del vector de movimiento actual o si el vector de movimiento del bloque vecino no está disponible. Si se cumple cualquiera de estas condiciones, el codificador de vídeo puede establecer que el vector de movimiento derivado del bloque actual es un vector de movimiento de disparidad, que el codificador de vídeo puede convertir a partir del correspondiente componente de vista de profundidad. El codificador de vídeo puede convertir el valor máximo de los valores de profundidad de las cuatro esquinas del bloque correspondiente de la componente de vista de profundidad de la misma vista en un valor de disparidad. El codificador de vídeo puede establecer el valor de disparidad como la componente horizontal del vector de movimiento derivado. El codificador de vídeo puede establecer que la componente vertical del vector de movimiento derivado sea cero.

[0104] Si el vector de movimiento actual es un vector de movimiento temporal, el codificador de vídeo puede utilizar el valor de disparidad (derivado de manera similar a la mencionada anteriormente) para determinar un vector de movimiento temporal del bloque de referencia en la vista de referencia (básica). El codificador de vídeo puede establecer que el vector de movimiento derivado sea el vector de movimiento temporal. Si se considera que el vector de movimiento temporal no está disponible (por ejemplo, el bloque vecino en el tiempo es un bloque intra o un vector de movimiento del bloque vecino en el tiempo no apunta a una imagen de referencia en la vista de referencia alineada con la imagen de referencia actual), el codificador de vídeo puede establecer el vector de movimiento derivado en cero.

[0105] A continuación se analiza la predicción de movimiento entre visitas en 3D-AVC para los modos de salto y directo. Como se describe en las secciones 7.3.5 y 7.4.5 de la especificación H.264/AVC, una estructura sintáctica macroblock_layer para un macrobloque puede incluir un elemento sintáctico mb_type que especifica un tipo de macrobloque para el macrobloque. La semántica del elemento sintáctico mb_type depende del tipo de segmento del segmento que contiene el macrobloque. Si el segmento es un segmento P, los tipos de macrobloques incluyen un tipo P_Skip. Cuando el tipo de macrobloque de un macrobloque es P_Skip, no hay más datos presentes para el macrobloque en el flujo de bits. Si el segmento es un segmento B, los tipos de macrobloques incluyen un modo

B_Skip y un B_Direct_16×16 (es decir, un modo directo B-16×16). Cuando el tipo de macrobloque de un macrobloque es B_Skip, no hay más datos presentes para el macrobloque en el flujo de bits. Cuando el tipo de macrobloque de un macrobloque es B_Direct_16×16, no hay diferencias de vector de movimiento o índices de referencia presentes para el macrobloque en el flujo de bits. Además, cuando el tipo de macrobloque de un macrobloque es B_Direct_16×16, las funciones MbPartWidth (B_Direct_16×16), y MbPartHeight (B_Direct_16×16) se utilizan en el proceso de derivación para vectores de movimiento e índices de trama de referencia según la subcláusula 8.4.1 de la especificación H.264/AVC para la predicción de modo directo.

[0106] Además, una estructura sintáctica macroblock_layer puede incluir una o más estructuras sintácticas sub_mb_pred. Una estructura sintáctica sub_mb_pred puede incluir cuatro elementos sintácticos sub_mb_type que especifican tipos de submacrobloques. Los tipos de submacrobloques incluyen un modo B_Direct_8×8 (es decir, un modo directo B-8×8). Cuando el tipo submacrobloque de un submacrobloque es B_Direct_8×8, no hay diferencias de vector de movimiento o índices de referencia presentes para el submacrobloque en el flujo de bits. Las funciones SubMbPartWidth (B_Direct_8×8) y SubMbPartHeight (B_Direct_8×8) se utilizan en el proceso de derivación para vectores de movimiento e índices de trama de referencia según la subcláusula 8.4.1 de la especificación H.264/AVC para la predicción de modo directo.

[0107] Un codificador de vídeo puede llevar a cabo la predicción de movimiento entre vistas en 3D-AVC en salto P, salto B, modo directo B-16×16 y modo directo B-8×8. Para realizar una predicción de movimiento entre vistas, el codificador de vídeo puede derivar primero un vector de disparidad para un bloque actual a partir de los bloques vecinos, así como el vector de disparidad convertido a partir de los valores de profundidad de la componente de vista de profundidad de la misma vista. Si un bloque vecino en el espacio disponible contiene un vector de movimiento de disparidad, el codificador de vídeo puede determinar que este vector de movimiento de disparidad es el vector de disparidad para el bloque actual. De lo contrario, cuando ninguno de los bloques vecinos tiene un vector de movimiento de disparidad, el codificador de vídeo puede convertir un vector de movimiento de disparidad de un bloque a partir de los valores de profundidad (de forma similar a la conversión en D-MVP). Posteriormente, el codificador de vídeo puede aplicar un filtro de mediana a tres bloques vecinos para determinar el vector de disparidad para el bloque actual.

[0108] El codificador de vídeo puede utilizar el vector de disparidad para el bloque actual para determinar un vector de movimiento temporal del bloque de referencia en la vista de referencia (por ejemplo, básica). Si el vector de movimiento temporal no está disponible, el codificador de vídeo puede derivar en primer lugar el índice de referencia, y el codificador de vídeo puede aplicar D-MVP, como se ha analizado anteriormente, para generar un indicador de vector de movimiento.

[0109] Las ampliaciones 3D-AVC y multivista de la HEVC pueden utilizar procesos de derivación de Vectores de Disparidad Basados en Bloques Vecinos (NBDV) para derivar vectores de disparidad para bloques. En general, un proceso de derivación de NBDV utiliza información de disparidad vecina para estimar un vector de disparidad para un bloque actual. Inicialmente se definen varios bloques candidatos vecinos en el espacio y en el tiempo. El codificador de vídeo puede comprobar cada uno de los bloques vecinos en un orden predefinido. El orden predefinido puede determinarse mediante una prioridad de la correlación entre el bloque actual y los bloques vecinos. Una vez que el codificador de vídeo determina que un bloque vecino tiene un vector de movimiento de disparidad (es decir, un vector de movimiento del bloque vecino apunta a una imagen de referencia entre vistas), el codificador de vídeo puede convertir el vector de movimiento de disparidad en un vector de disparidad para el bloque actual.

[0110] En general, el codificador de vídeo puede utilizar dos conjuntos de bloques vecinos al llevar a cabo un proceso de derivación de NBDV. Un conjunto de bloques vecinos pertenece a los bloques vecinos en el espacio y el otro conjunto de bloques vecinos pertenece a los bloques vecinos en el tiempo. En algunos ejemplos, el término NBDV se utiliza para referirse a un vector de disparidad de un bloque si se ha utilizado un proceso de derivación de NBDV para derivar el vector de disparidad del bloque.

[0111] Un codificador de vídeo puede utilizar el proceso de derivación de NBDV como procedimiento de derivación de vectores de disparidad en 3D-HEVC, que utiliza el orden de codificación de textura primero para todas las vistas. En el diseño 3D-HEVC actual, el codificador de vídeo también puede utilizar el proceso de derivación de NBDV para recuperar datos de profundidad de un mapa de profundidad de una vista de referencia. La 3D-HEVC adoptó primero el procedimiento NBDV propuesto en el documento de Zhang *et al.* "Disparity vector generation results", Equipo de Colaboración Conjunta sobre Elaboración de Ampliación de Codificación de Vídeo de la ITU-T SG 16 WP 3 e ISO/IEC JTC 1/SC 29/WG 11, 1st Meeting: Estocolmo, SE, 16-20 de julio de 2012, documento JCT3V-A0097 (en adelante, JCT3V-A0097). Los vectores de disparidad implícita se incluyen con un NBDV simplificado en JCT3V-A0126: Sung et *al*., "3D-CE5.h: Simplification of disparity vector derivation for HEVC-based 3D video coding", Equipo de Colaboración Conjunta sobre Elaboración de Ampliación de Codificación de Vídeo de la ITU-T SG 16 WP 3 e ISO/IEC JTC 1/SC 29/WG 11, 1st Meeting: Estocolmo, Suecia, 16-20 de julio de 2012, documento JCT3V-A0126 (en adelante, JCT3V-A0126). Además, en Kang *et al*., "3D-CE5.h related: Improvements for disparity vector derivation", Equipo de Colaboración Conjunta sobre Elaboración de Ampliación de Codificación de Vídeo de la ITU-T SG 16 WP

3 e ISO/IEC JTC 1/SC 29/WG 11, 2nd Meeting: Shanghai, CN, 13-19 Oct. de 2012, documento JCT3V-B0047 (en adelante, JCT3V-B0047), el proceso de derivación de NBDV se simplifica además eliminando los vectores de disparidad implícita almacenados en el la memoria intermedia de imágenes decodificadas, al tiempo que se consigue una ganancia de codificación mejorada con la selección de imagen de acceso aleatorio (RAP).

[0112] En algún proceso de derivación de NBDV, un codificador de vídeo utiliza cinco bloques vecinos en el espacio para la derivación del vector de disparidad. Los cinco bloques vecinos en el espacio son los bloques inferior izquierdo, izquierdo, superior derecho, superior y superior izquierdo de una PU actual, designados por A_0, A_1, B_0, B_1 y B_2. Los cinco bloques vecinos en el espacio utilizados en el proceso de derivación de NBDV propuesto pueden ser los mismos cinco bloques vecinos en el espacio utilizados en los modos de fusión en HEVC. Por lo tanto, en algunos ejemplos, no se requiere ningún acceso adicional a la memoria para acceder a los cinco bloques vecinos en el espacio.

[0113] Para la comprobación de bloques vecinos en el tiempo, un codificador de vídeo puede realizar primero un proceso de confección de una lista de imágenes candidatas. El codificador de vídeo puede tratar todas las imágenes de referencia de la vista actual como imágenes candidatas. El codificador de vídeo puede insertar primero una denominada imagen de referencia coubicada en la lista de imágenes candidatas, seguida por el resto de imágenes candidatas (es decir, imágenes de referencia de la vista actual) por orden ascendente del índice de referencia. Es decir, el codificador de vídeo puede insertar las imágenes candidatas restantes en la lista de imágenes candidatas de acuerdo con el orden en que las imágenes candidatas restantes aparecen en las listas de imágenes de referencia (por ejemplo, RefPicList0 y RefPicList1) de la imagen actual. Uno o más elementos sintácticos de una cabecera de segmento de un segmento que contiene el bloque actual puede indicar la imagen coubicada. En algunos ejemplos, cuando las imágenes de referencia con el mismo índice de referencia en ambas listas de imágenes de referencia (por ejemplo, RefPicList0 y RefPicList1) están disponibles para su uso en el proceso de derivación de NBDV, la imagen de referencia de la misma lista de imágenes de referencia que la imagen coubicada precede, en la lista de imágenes candidatas, a la otra imagen de referencia.

[0114] Cuando un codificador de vídeo codifica un bloque con predicción de movimiento entre vistas, el codificador de vídeo puede derivar un vector disparidad para seleccionar un bloque correspondiente en una vista diferente. El término "vector de disparidad implícita" o "IDV" (o en algunas circunstancias "vector de disparidad derivado") puede referirse a un vector de disparidad derivado en la predicción de movimiento entre vistas. Por ejemplo, aunque un codificador de vídeo puede codificar un bloque con predicción de movimiento (es decir, predicción de movimiento temporal), el codificador de vídeo no descarta el vector de disparidad derivado. Por el contrario, el codificador de vídeo puede utilizar el vector de disparidad con el propósito de codificar un bloque siguiente. Específicamente, el codificador de vídeo puede tratar el vector de disparidad como un vector de disparidad implícita y puede usar el vector de disparidad implícita en un proceso de derivación de NBDV para determinar un vector de disparidad para uno o más bloques adicionales.

[0115] Típicamente, cuando el codificador de vídeo realiza el proceso de derivación de NBDV, el codificador de vídeo comprueba unos vectores de movimiento de disparidad en los bloques vecinos en el tiempo, los vectores de movimiento de disparidad en los bloques vecinos en el espacio y finalmente el vector de disparidad implícita, por orden. Una vez que el codificador de vídeo encuentra el vector de disparidad, el codificador de puede terminar el proceso de derivación de NBDV.

[0116] La VSP regresiva puede estar habilitada en 3D-HEVC. En 3D-HEVC, cuando un codificador de vídeo aplica un orden de codificación de textura primero, el codificador de vídeo puede derivar, para cada PU, un vector de disparidad a partir del proceso de derivación de NBDV con o sin consideración a los valores de profundidad de una vista de profundidad de referencia. Una vez que el codificador de vídeo ha obtenido un vector de disparidad, el codificador de vídeo puede perfeccionar aún más el vector de disparidad para cada subregión 4×4 de una PU si la subregión 4×4 de la PU se codifica con el modo BVSP.

[0117] El proceso de perfeccionamiento puede incluir dos etapas. En la primera etapa, el codificador de vídeo puede seleccionar un valor de profundidad máxima de un bloque de profundidad 4×4 en la vista de profundidad de referencia. El codificador de vídeo puede utilizar el vector de disparidad derivado para localizar el bloque de profundidad 4×4. En la segunda etapa, el codificador de vídeo puede convertir el valor de profundidad en una componente horizontal del vector de disparidad perfeccionado, mientras mantiene la componente vertical del vector de disparidad perfeccionado en 0. Una vez que se ha perfeccionado el vector de disparidad para una subregión 4×4 de una PU, el codificador de vídeo puede utilizar el vector de disparidad perfeccionado para localizar un bloque en la vista de textura de referencia para la compensación de movimiento.

[0118] Como se ha indicado anteriormente, la 3D-HEVC puede usar un proceso de derivación de NBDV para derivar vectores de disparidad. En la solicitud de patente provisional de Estados Unidos nº 61/769,716, presentada el 26 de febrero de 2013, en lo sucesivo la solicitud ‘716, se describe un procedimiento de derivación de NBDV para 3D-AVC. Como se describe en la solicitud '716, un codificador de vídeo puede usar un proceso de derivación de NBDV de nivel MB para derivar un vector de disparidad para un MB actual. El codificador de vídeo puede utilizar además el

vector de disparidad para el MB actual para la predicción del vector de movimiento. Como se describe en la solicitud '716, una vez que el codificador de vídeo identifica un vector de movimiento de disparidad (es decir, una vez que el codificador de vídeo identifica un bloque vecino en el tiempo o en el espacio que usa una imagen de referencia entre vistas), el codificador de vídeo puede presentar el vector de movimiento de disparidad como vector de disparidad para el MB actual.

[0119] La FIG. 5 es un diagrama conceptual que ilustra unos ejemplos de bloques vecinos en el espacio para un proceso de derivación de NBDV. En la solicitud '716, cuando un codificador de vídeo comprueba bloques vecinos en el espacio en el proceso de derivación de NBDV, el codificador de vídeo puede comprobar los bloques vecinos en el espacio que el codificador de vídeo comprueba en el proceso de predicción de movimiento AVC. Cuando el codificador de vídeo comprueba los bloques vecinos en el espacio en el proceso de derivación de NBDV propuesto en la solicitud '716, el codificador de vídeo puede comprobar los bloques vecinos en el espacio en el orden de A (izquierdo), B (superior), C (superior derecho) y D (superior izquierdo), como se muestra en el ejemplo de la FIG. 5. En otras palabras, los bloques vecinos en el espacio que se comprobarán en el proceso de predicción de movimiento AVC se comprueban en el orden de A (izquierdo), B (superior), C (superior derecho) y D (superior izquierdo) en el proceso NBDV propuesto.

[0120] Además, en el proceso de derivación de NBDV propuesto en la solicitud '716, el codificador de vídeo puede comprobar bloques vecinos en el tiempo. La FIG. 6 es un diagrama conceptual que ilustra ejemplos de bloques vecinos en el tiempo para un proceso de derivación de NBDV. El codificador de vídeo puede comprobar bloques de hasta dos imágenes de referencia en la misma vista que la imagen actual: (RefPicList1 [0] y RefPicList0[0] para segmentos B y RefPicList0[0] para segmentos P). Actualmente, el codificador de vídeo puede comprobar tres bloques temporales imagen por imagen. Para cada imagen, el codificador de vídeo puede comprobar los bloques coubicados en relación con el MB coubicado como se indica a continuación en el orden BR (inferior derecho), CT3 (centro 3) y CO2 (esquina 2), como se indica en el ejemplo de la FIG. 6.

[0121] Además, en la solicitud '716, la terminación del proceso de derivación de NBDV se describe como sigue. El codificador de vídeo puede comprobar los bloques vecinos mencionados anteriormente por orden. De forma similar a la 3D-HEVC, el codificador de vídeo puede comprobar bloques vecinos en el tiempo primero y puede comprobar los bloques vecinos en el espacio después. Una vez que el codificador de vídeo determina que un bloque contiene un vector de movimiento de disparidad disponible, el codificador de vídeo puede terminar el proceso de derivación.

[0122] En la tabla siguiente se muestra la ganancia de codificación del procedimiento propuesto en la solicitud '716, comparada con la MVC+D.

	Codificación de textura		Codificación de profundidad		Total (PSNR codificada)		Total (PSNR sintetizada)	
	dBR,%	dPSNR,dB	dBR,%	dPSNR,dB	dBR,%	dPSNR,dB	dBR,%	dPSNR,dB
S01	27,67	-0,83	-3,83	0,20	33,18	-0,98	32,82	-0,99
S02	7,95	-0,23	-15,22	0,68	12,20	-0,35	11,49	-0,34
S03	16,99	-0,55	-20,91	1,85	15,99	-0,53	20,24	-0,60
S04	19,99	-0,67	-21,22	1,43	25,03	-0,83	20,34	-0,64
S05	22,77	-1,03	-22,40	1,30	44,49	-1,78	39,03	-1,45
S06	29,93	-1,36	-15,42	0,75	43,01	-1,80	36,11	-1,44
S08	13,15	-0,54	-11,16	0,49	19,79	-0,77	18,82	-0,63
Promedio	19,78	-0,74	-15,74	0,96	27,67	-1,01	25,55	-0,87

El procedimiento propuesto de la solicitud '716 permite la codificación solo de textura, que no se admite eficazmente en la 3D-AVC. Cuando se habilita la misma configuración de solo textura, la ganancia de codificación de la 3D-AVC actual es solo del 1%.

[0123] En algunos ejemplos, se utiliza un proceso derivación de NBDV que accede a una componente de vista de profundidad de una vista de referencia en 3D-AVC. Como se describe en la solicitud de patente provisional de Estados Unidos nº 61/770,268 (la solicitud '268), presentada el 27 de febrero de 2013, el proceso de derivación de NBDV puede mejorarse más accediendo al componente de vista de profundidad de una vista básica/de referencia. Tal como se describe en la solicitud '268, un codificador de vídeo puede utilizar el vector de disparidad derivado de los bloques vecinos para localizar píxeles de profundidad en la componente de vista de profundidad, de manera que el codificador de vídeo pueda perfeccionar aún más el vector de disparidad. La siguiente tabla muestra una ganancia de codificación del procedimiento propuesto de la solicitud '268 comparada con la MVC+D.

	Codificación de textura		Codificación de profundidad h		Total (PSNR codificada)		Total (PSNR sintetizada)	
	dBR,%	dPSNR,dB	dBR,%	dPSNR,dB	dBR,%	dPSNR,dB	dBR,%	dPSNR,dB
S01	40,61	-1,09	-4,43	0,23	45,32	-1,22	43,63	-1,20
S02	12,89	-0,36	-18,08	0,83	16,61	-0,46	14,90	-0,43
S03	22,07	-0,69	-25,71	2,40	20,22	-0,65	23,97	-0,69
S04	28,87	-0,93	-27,45	1,90	33,30	-1,06	26,72	-0,81
S05	28,47	-1,22	-22,69	1,33	49,34	-1,89	44,42	-1,58
S06	35,69	-1,56	-16,56	0,81	48,32	-1,97	40,81	-1,58
S08	16,48	-0,66	-11,57	0,51	22,51	-0,85	21,52	-0,70
Promedio	26,44	-0,93	-18,07	1,14	33,66	-1,16	30,85	-1,00

Como se ha mostrado anteriormente, el procedimiento propuesto de la solicitud ‘268 proporciona un 5% más de ganancia de codificación, aunque se sigue necesitando acceder al componente de vista de profundidad.

[0124] Los procedimientos de derivación de NBDV diseñados para la 3D-AVC pueden experimentar los siguientes problemas. En primer lugar, cuando se utiliza el procedimiento de derivación de NBDV propuesto en la solicitud '716, las herramientas de codificación que incluyen D-MVP y BVSP pueden llegar a ser menos eficientes, principalmente debido a que los vectores de disparidad no son lo suficientemente precisos. En segundo lugar, cuando se utiliza el procedimiento de derivación de NBDV propuesto en la solicitud '268, siempre es necesario acceder a una componente de vista de profundidad. Incluso en este caso, el vector de disparidad inicial obtenido a partir de los bloques vecinos también podría impedir que el codificador de vídeo identificara un área más precisa en la componente de vista de profundidad. En tercer lugar, un vector de disparidad podría mejorarse accediendo a los IDV como en la 3D-HEVC. Sin embargo, el uso de dichos IDV puede requerir almacenamiento adicional tanto para los bloques vecinos en el espacio como para los bloques vecinos en el tiempo. El almacenamiento de IDV para bloques vecinos en el espacio y en el tiempo puede requerir una cantidad de almacenamiento adicional significativa en un nivel similar al tamaño del campo de movimiento y más accesos a la memoria.

[0125] Esta divulgación propone un proceso NBDV mejorado. Específicamente, las técnicas de esta divulgación pueden proporcionar soluciones para mejorar la generación de NBDV (es decir, el proceso de derivación de NBDV) almacenando uno o más vectores de disparidad derivados (DDV) para un segmento entero. Las técnicas de esta divulgación se pueden diseñar principalmente para procedimientos NBDV como se describe en la solicitud '716 y la solicitud' 268.

[0126] De acuerdo con una o más técnicas de la presente divulgación, un codificador de vídeo (por ejemplo, el codificador de vídeo 20 o el decodificador de vídeo 30) puede almacenar uno o más DDV. El codificador de vídeo puede usar los DDV para predecir vectores de disparidad de bloques de vídeo codificados subsiguientemente. En otras palabras, se almacenan uno o más DDV de los bloques anteriores para una mejor predicción del vector de disparidad de bloques futuros.

[0127] En algunos ejemplos, el codificador de vídeo no utiliza un DDV para predecir un vector de disparidad de un bloque actual si el bloque actual es una componente de vista de profundidad y el bloque a partir del cual el codificador de vídeo ha derivado el DDV es una componente de vista de textura. Además, el codificador de vídeo no utiliza un DDV para predecir un vector de disparidad de un bloque actual si el bloque actual es una componente de vista de textura y el bloque a partir del cual el codificador de vídeo ha derivado el DDV es una componente de vista de profundidad. Por ejemplo, un DDV no se utiliza para predecir un vector de disparidad de ningún bloque de ningún componente de vista que es diferente de la componente de vista que no contiene el MB actual. En otras palabras, un DDV no se utiliza para predecir un vector de disparidad de ningún bloque de ningún componente de vista que no contiene el MB actual.

[0128] Además, en algunos ejemplos, el codificador de vídeo no utiliza un DDV para predecir un vector de disparidad de un bloque actual si el bloque actual y el bloque a partir del cual el codificador de vídeo ha derivado el DDV están en segmentos diferentes. Por ejemplo, un DDV no se utiliza para predecir un vector de disparidad de ningún bloque de ningún segmento que es diferente del segmento que no contiene el MB actual.

[0129] Como se ha indicado anteriormente, el codificador de vídeo puede mantener uno o más DDV para un segmento. En algunos ejemplos, el número de DDV mantenidos para el segmento por el codificador de vídeo no es proporcional a la anchura de una imagen actual, la altura de la imagen actual o el número de bloques de un segmento actual. En otras palabras, el número de DDV que se va a mantener no es proporcional a la anchura de la imagen, la altura de imagen o el número de bloques que tiene un segmento. Más bien, en algunos ejemplos, el

número de DDV mantenidos por el codificador de vídeo es proporcional al número de bloques de un MB, una CU o una LCU.

[0130] En algunos ejemplos de esta divulgación, el codificador de vídeo solo almacena un DDV. Por ejemplo, una vez que el codificador de vídeo codifica el primer (por orden de codificación) MB de un segmento, el codificador de vídeo puede almacenar el vector de disparidad del MB para el DDV. Al codificar el MB actual, el codificador de vídeo puede usar el DDV almacenado en un proceso de derivación de NBDV para derivar un vector de disparidad para un MB subsiguiente. En otras palabras, cuando se codifica el MB actual, el DDV que se almacena previamente, generado al codificar el MB anterior (que típicamente es el MB izquierdo), se utiliza para el NBDV.

[0131] De esta manera, un codificador de vídeo puede, en algunos ejemplos, almacenar un DDV para un segmento de una imagen actual de los datos de vídeo. Además, el codificador de vídeo puede derivar, basándose al menos en parte en un DDV para un segmento de una imagen actual de los datos de vídeo, un NBDV para un primer bloque del segmento. El codificador de vídeo puede codificar el primer bloque basándose al menos en parte en el NBDV para el primer bloque. Además, el codificador de vídeo puede almacenar el NBDV para el primer bloque como DDV. Por ejemplo, una vez que (es decir, después de que) el codificador de vídeo ha codificado el primer MB del segmento, el codificador de vídeo puede almacenar el vector de disparidad del MB como DDV. Por lo tanto, en algunos ejemplos, el codificador de vídeo puede almacenar solo un DDV para un segmento de una imagen actual de los datos de vídeo, en el que el segmento incluye varios bloques. Además, en dichos ejemplos, el codificador de vídeo puede utilizar el DDV para el segmento en un proceso de derivación de NBDV a fin de determinar un vector de disparidad para un bloque particular, en el que el bloque particular es uno de los bloques del segmento. En dichos ejemplos, el codificador de vídeo puede almacenar, como DDV para el segmento, el vector de disparidad para el bloque particular

[0132] Después de almacenar el NBDV para el primer bloque como DDV, el codificador de vídeo puede derivar, basándose al menos en parte en el DDV, un NBDV para un segundo bloque del segmento. En algunos ejemplos, el primer bloque está justo a la izquierda del segundo bloque. Además, el codificador de vídeo puede codificar el segundo bloque basándose al menos en parte en el NBDV para el segundo bloque. En dichos ejemplos, el primer bloque puede ser un macrobloque y el segundo bloque también puede ser un macrobloque. En otros casos, el primer bloque y el segundo bloque pueden ser CU, el primer bloque y el segundo bloque pueden ser PU o el primer bloque y el segundo bloque pueden ser LCU.

[0133] En algunos ejemplos, el codificador de vídeo puede actualizar el DDV para especificar el vector de disparidad del MB actual. Por ejemplo, el codificador de vídeo puede actualizar el DDV para especificar el vector de disparidad del MB actual solamente cuando el codificador de vídeo codifica el MB actual mediante predicción de movimiento entre vistas. En otras palabras, el DDV solo se actualiza cuando el MB actual se codifica utilizando el resultado NBDV, por ejemplo, utilizando la predicción de movimiento entre vistas. En otro ejemplo, el codificador de vídeo solo actualiza el DDV para especificar el vector de disparidad del MB actual cuando el MB actual está intercodificado. En otras palabras, el DDV solo se actualiza cuando el MB actual está intercodificado. En otro ejemplo, el codificador de vídeo solo actualiza el DDV para especificar el vector de disparidad del MB actual cuando el MB actual está en un segmento intercodificado (por ejemplo, un segmento P o un segmento B). Por lo tanto, en este ejemplo, el DDV solo se actualiza cuando el segmento actual está intercodificado. En otro ejemplo, el codificador de vídeo solo actualiza el DDV para especificar el vector de disparidad del MB actual cuando el MB actual está codificado con el modo de salto o el modo directo. En otras palabras, el DDV solo se actualiza cuando el MB actual se codifica con el modo de salto o el modo directo.

[0134] Además, en algunos ejemplos, el codificador de vídeo puede almacenar como DDV un vector de disparidad para un bloque, como respuesta a la determinación de que una magnitud del vector de disparidad para el bloque es mayor que una magnitud del DDV. Por ejemplo, el codificador de vídeo puede actualizar el DDV para especificar el vector de disparidad del MB actual solamente si una componente horizontal del vector de disparidad del MB actual es mayor que una componente horizontal del DDV. En otras palabras, el DDV solo se almacena/actualiza más si el vector de disparidad derivado del MB actual es mayor que el DDV almacenado previamente. Se comparará aquí el valor absoluto de la componente horizontal del vector de disparidad.

[0135] En algunos ejemplos, el codificador de vídeo actualiza el DDV para especificar el vector de disparidad de un bloque actual solo si el DDV es cero. En algunos ejemplos, el codificador de vídeo puede determinar que el DDV es cero cuando ambas componentes del DDV son iguales a cero. En otros ejemplos, el codificador de vídeo puede determinar que el DDV es cero cuando la componente horizontal del DDV es igual a cero. Por lo tanto, en dichos ejemplos, el DDV solo se almacena o actualiza más si el DDV es cero (por ser ambas componentes cero o por ser simplemente la componente horizontal cero).

[0136] En algunos ejemplos, un codificador de vídeo puede derivar un vector disparidad para un bloque actual a partir de un DDV almacenado solo cuando el DDV almacenado especifica un vector de disparidad para un bloque situado a la izquierda del bloque actual. Por ejemplo, si un MB a partir del cual el codificador de vídeo ha derivado un DDV no es adyacente al lado izquierdo de un MB actual, el codificador de vídeo no utiliza el DDV para derivar un vector de disparidad para el MB actual. En otras palabras, si el MB anterior utilizado para generar el DDV no es el

MB situado en el lado izquierdo del MB actual, el DDV no se utiliza para derivar el NBDV del MB actual.

[0137] Por lo tanto, en los diversos ejemplos de esta divulgación proporcionados anteriormente y en otras partes de esta divulgación, el codificador de vídeo puede actualizar un DDV para especificar el NBDV para un bloque como respuesta a una condición que se cumple, siendo la condición una de entre: el primer bloque se codifica mediante predicción de movimiento entre vistas, el primer bloque se intercodifica, el segmento se intercodifica, el primer bloque se codifica con el modo de salto o el modo directo, una magnitud del NBDV para el primer bloque es mayor que una magnitud del DDV y el DDV no es cero.

[0138] Un codificador de vídeo puede derivar el NBDV para el bloque actual basándose al menos en parte en un DDV almacenado, unos vectores de movimiento de disparidad para unos bloques vecinos en el tiempo (TDV) y unos vectores de movimiento disparidad para unos bloques vecinos en el espacio (SDV). Es decir, el DDV se puede utilizar junto con los vectores de movimiento de disparidad posibles de los bloques vecinos en el tiempo (designados por TDV) y vectores de movimiento de disparidad de los bloques vecinos en el espacio (designados por SDV) de varias maneras. Por lo tanto, derivar el NBDV para un primer bloque puede comprender usar el DDV, un vector de movimiento de disparidad para un bloque vecino en el tiempo, y un vector de movimiento de disparidad para un bloque vecino en el espacio para derivar el NBDV para el bloque.

[0139] Por ejemplo, el codificador de vídeo puede establecer el vector disparidad del bloque actual como DDV cuando el codificador de vídeo es incapaz de identificar cualquier TDV o SDV. En otras palabras, si no se identifica ningún TDV ni SDV disponible, el vector de disparidad se establece como DDV. Por lo tanto, como respuesta a la determinación de que no hay ningún TDV o SDV disponible para su uso en la determinación de un NBDV para un bloque actual, el codificador de vídeo puede determinar que el NBDV para el bloque actual es igual al DDV almacenado.

[0140] En algunos ejemplos, el codificador de vídeo solo establece el vector de disparidad para el bloque actual como DDV si no hay TDV o SDV disponibles y el DDV no es cero. El codificador de vídeo puede determinar que el DDV no es cero cuando cualquiera de las componentes del DDV es distinto de cero. De forma alternativa, el codificador de vídeo puede determinar que el DDV no es cero cuando la componente horizontal del DDV es distinta de cero. En este ejemplo, el codificador de vídeo puede establecer el vector de disparidad para el bloque actual en cero si el DDV es cero. En otras palabras, cuando los vecinos en el tiempo y en el espacio no proporcionan ningún vector de movimiento de disparidad disponible, si el DDV no es cero (por ser ambas componentes cero o por ser simplemente la componente horizontal cero), el DDV que no es cero se presenta como NBDV; en caso contrario, se presenta un vector de disparidad cero.

[0141] Además, en algunos ejemplos en los que el codificador de vídeo lleva a cabo un proceso de derivación de NBDV en el que se comprueban unos TDV y SDV además de comprobarse un DDV, el codificador de vídeo puede comprobar los TDV, SDV, y el DDV en diversos órdenes. En otras palabras, los vectores de disparidad se comprueban en un orden predefinido dado. Una vez que el codificador de vídeo identifica un vector de movimiento de disparidad que está disponible y que no es cero (por ser ambas componentes no cero o por ser simplemente la componente horizontal no cero), el codificador de vídeo puede derivar el vector de disparidad del bloque actual a partir del vector de movimiento de disparidad identificado. Por ejemplo, cuando el codificador de vídeo identifica un vector de movimiento de disparidad disponible no cero entre los DDV almacenados, los TDV y los SDV mientras comprueba el DDV almacenado, los TDV y los SDV en un orden predefinido, un codificador de vídeo puede determinar que un vector de disparidad para un bloque actual es igual al vector de disparidad distinto de cero disponible. Los posibles órdenes de comprobación predefinidos pueden incluir:

- Los TDV, los SDV y el DDV;
- Los SDV, los TDV y el DDV;
- El DDV, los SDV y los TDV;
- El DDV, los TDV y los SDV;
- Los TDV, el DDV y los SDV; y
- Los SDV, el DDV y los TDV.

Por ejemplo, si el codificador de vídeo utiliza el orden de los TDV, los SDV y el DDV o los SDV, los TDV y el DDV, derivar el NBDV para un bloque puede comprender establecer el NBDV para el bloque como DDV después de determinar que el bloque vecino en el tiempo no tiene un vector de movimiento de disparidad y después de determinar que el bloque vecino en el espacio no tiene un vector de movimiento de disparidad.

[0142] Más específicamente, si el codificador de vídeo utiliza el orden de los TDV, los SDV y el DDV, el codificador de vídeo puede determinar si un bloque vecino en el tiene un vector de movimiento de disparidad. Además, como respuesta a la determinación de que el bloque vecino en el tiempo tiene un vector de movimiento de disparidad, el codificador de vídeo puede derivar el NBDV para el bloque basándose en el vector de movimiento de disparidad del bloque vecino en el tiempo. Como respuesta a la determinación de que el bloque vecino en el tiempo no tiene un vector de movimiento de disparidad, el codificador de vídeo puede determinar si cualquier bloque vecino en el espacio de una pluralidad de bloques vecinos en el espacio tiene un vector de movimiento de disparidad. Como

respuesta a la determinación de que un bloque vecino en el espacio particular de la pluralidad de bloques vecinos en el espacio tiene un vector de movimiento de disparidad, el codificador de vídeo puede derivar el NBDV para el bloque basándose en el vector de movimiento de disparidad del bloque vecino en el espacio particular. Como respuesta a la determinación de que ningún bloque vecino en el espacio de la pluralidad de bloques vecinos en el espacio tiene un vector de movimiento de disparidad, el codificador de vídeo puede derivar el NBDV para el bloque como DDV.

[0143] En algunos ejemplos, el codificador de vídeo deriva dos o más DDV. En dichos ejemplos, el codificador de vídeo puede formar los DDV en un cierto orden. En algunos de dichos ejemplos, el codificador de vídeo puede considerar conjuntamente los DDV con los TDV y los SDV en el mismo orden predefinido que el descrito en los ejemplos anteriores donde el codificador de vídeo realiza un proceso de derivación de NBDV que comprueba los TDV y SDV además de comprobar un DDV, reproduciendo el único DDV con varios DDV. Por ejemplo, cuando se identifica un vector de disparidad distinto de cero disponible entre los TDV, los SDV y los dos o más DDV almacenados mientras se comprueban los TDV, los SDV y los dos o más DDV almacenados en un orden de comprobación predefinido, un codificador de vídeo puede determinar que un vector de disparidad para un bloque actual es igual al vector de disparidad distinto de cero disponible, en el que se comprueban en el orden de comprobación predefinido los dos o más DDV almacenados de acuerdo con el orden particular.

[0144] En algunos ejemplos en los que el codificador de vídeo deriva dos o más DDV, si un DDV no se genera a partir de un MB vecino del bloque actual, el DDV se excluye de los DDV utilizados para predecir el vector de disparidad del MB actual.

[0145] Además, en algunos ejemplos, el codificador de vídeo deriva dos DDV: DDV 1 y DDV 2. Puede establecerse inicialmente que DDV 1 y DDV 2 sean iguales. Sin embargo, el DDV 1 se actualiza una vez que el MB actual, que es el MB vecino de la derecha del MB actual, se codifica y el DDV 2 se actualiza una vez que el MB actual, que es el MB vecino superior del MB actual, se codifica. Es decir, el DDV 1 se actualiza una vez que el MB actual se codifica como MB vecino de la derecha del MB codificado previamente de forma consecutiva, y el DDV 2 se actualiza una vez que el MB actual se codifica como MB vecino inferior del MB codificado previamente de forma consecutiva. Cuando el codificador de vídeo codifica un MB actual, el codificador de vídeo puede codificar DDV 1 y DDV 2, teniendo DDV 1 una prioridad superior a DDV 2. En otras palabras, el codificador de vídeo comprueba DDV 1 antes que DDV 2. En otro ejemplo, el codificador de vídeo solo utiliza DDV 1 para derivar el vector de disparidad del bloque actual si DDV 1 se deriva del bloque situado a la izquierda del bloque actual. Por ejemplo, solo si DDV 1 se deriva a partir del MB vecino de la izquierda, se utiliza DDV 1 para derivar el vector de disparidad. Además, el codificador de vídeo puede usar DDV 2 para derivar el vector de disparidad para un bloque actual si DDV 2 se deriva a partir del bloque que está encima del bloque actual. Por ejemplo, si DDV 2 se deriva a partir del MB vecino superior, DDV 2 se utiliza para derivar el vector de disparidad.

[0146] En algunos ejemplos, cuando el acceso a una componente de vista de profundidad de una vista básica/de referencia está permitido, son posibles dos o más sistemas alternativos en términos de uso del DDV. En un sistema, el codificador de vídeo solo utiliza el DDV cuando el acceso a la componente de vista de profundidad no está permitido. Por ejemplo, como respuesta a la determinación de que el acceso a una componente de vista de profundidad no está permitido, el codificador de vídeo puede derivar, basándose en uno o más DDV almacenados, un vector de disparidad para un bloque actual.

[0147] En otro ejemplo en el que el acceso a una componente de vista de profundidad de una vista básica/de referencia está permitido, el codificador de vídeo utiliza el DDV para determinar un vector de disparidad inicial como se ha descrito anteriormente. Posteriormente, el codificador de vídeo utiliza además el vector de disparidad inicial para identificar un bloque de profundidad potencialmente más preciso para perfeccionar el vector de movimiento de disparidad. Por ejemplo, cuando el acceso a una componente de vista de profundidad de una vista básica/de referencia está permitido, el codificador de vídeo puede determinar, basándose en uno o más DDV almacenados, un NBDV inicial para el bloque actual y puede utilizar el vector de disparidad inicial para el bloque actual para identificar un bloque de profundidad potencialmente más preciso a fin de perfeccionar el vector de disparidad inicial para el bloque actual.

[0148] Asimismo, cuando en algunos ejemplos en los que el acceso a una componente de vista de profundidad de una vista básica/de referencia está permitido, un codificador de vídeo puede mejorar un DDV mediante el vector de disparidad perfeccionado como se describe en la solicitud '268. Por ejemplo, cuando el acceso a una componente de vista de profundidad de una vista básica/de referencia está permitido, un codificador de vídeo puede perfeccionar uno o más DDV almacenados. En algunos ejemplos, el codificador de vídeo almacena el vector de disparidad perfeccionado en el nivel de MB como DDV. En otro ejemplo, el codificador de vídeo almacena el vector de disparidad perfeccionado en un nivel de bloque menor que el nivel de MB, como DDV. Por ejemplo, el codificador de vídeo puede almacenar el vector de disparidad perfeccionado de la esquina inferior derecha dentro del MB como DDV. De esta manera, el codificador de vídeo puede perfeccionar, basándose en píxeles de profundidad de una componente de vista de profundidad, el NBDV para un bloque y cuando el codificador de vídeo almacena el NBDV para el bloque como DDV, el codificador de vídeo puede actualizar el DDV para especificar el NBDV perfeccionado para el bloque.

[0149] La siguiente es una generalización de las técnicas propuestas (por ejemplo, procedimientos) de la presente divulgación para la 3D-HEVC. En un primer punto de una generalización de las técnicas propuestas para la 3D-HEVC, las técnicas anteriormente descritas de esta divulgación también se pueden aplicar a la 3D-HEVC reemplazando los MB por PU, CU o LCU. En un segundo punto de una generalización de las técnicas propuestas de esta divulgación para la 3D-HEVC, cuando las técnicas propuestas de esta divulgación se usan en 3D-HEVC, los codificadores de vídeo pueden no necesitar almacenar IDV. En otras palabras, en 3D-AVC, 3D-HEVC y potencialmente otras normas y especificaciones de codificación de vídeo, el codificador de vídeo no puede, de acuerdo con algunos ejemplos de esta divulgación, almacenar unos IDV para cualquier bloque del segmento.

[0150] En un tercer punto de una generalización de las técnicas propuestas para la 3D-HEVC, para cada LCU o CU, un codificador de vídeo puede almacenar unos DDV de los bloques del extremo derecho en una unidad más grande o igual a la unidad de codificación más pequeña (por ejemplo, 4×4 u 8×8). Cuando se codifica una PU actual, el codificador de vídeo puede usar los DDV que el codificador de vídeo ha derivado a partir de los bloques de la izquierda de la PU actual de la LCU izquierda para predecir el vector de disparidad de la PU actual. Por ejemplo, cuando la LCU es de 64×64 y la unidad de codificación más pequeña es de 4×4, el codificador de vídeo puede almacenar hasta 16 DDV. Algunos de los DDV almacenados pueden ser iguales, ya que los DDV almacenados pertenecen a los bloques de la misma PU.

[0151] Además, para una LCU o CU, el codificador de vídeo puede almacenar solamente un DDV derivado a partir de una de las PU de la LCU o CU. Al codificar una PU actual, el codificador de vídeo puede usar el DDV de la LCU/CU situada a la izquierda de la PU actual para predecir el vector de disparidad de la PU actual. En un ejemplo, el codificador de vídeo puede seleccionar el DDV de una LCU o CU que se va a almacenar a través de una función de todos los DDV disponibles de las PU de la LCU o CU.

[0152] De esta manera, el codificador de vídeo 20 puede almacenar uno o más DDV de uno o más bloques codificados previamente de un segmento. Además, el codificador de vídeo 20 puede derivar, basándose en el uno o más DDV almacenados, un NBDV para un bloque actual del segmento. El codificador de vídeo 20 puede codificar el bloque actual basándose al menos en parte en el NBDV para el bloque actual. De manera similar, el decodificador de vídeo 20 puede almacenar uno o más DDV de uno o más bloques previamente codificados de un segmento. Además, el decodificador de vídeo 20 puede derivar, basándose al menos en parte en el uno o más DDV almacenados, un NBDV para un bloque actual del segmento. El decodificador de vídeo 30 puede decodificar el bloque actual basándose al menos en parte en el NBDV para el bloque actual.

[0153] En otro punto de una generalización de las técnicas propuestas de la presente divulgación para la 3D-HEVC, el codificador de vídeo puede mantener solo una DDV para cada segmento. El codificador de vídeo puede actualizar el DDV una vez que se codifica una CU. En diferentes ejemplos, el codificador de vídeo puede actualizar el DDV de diferentes maneras.

[0154] Por ejemplo, cuando un segmento no se intracodifica, el codificador de vídeo puede invocar el proceso de derivación de NBDV para cada CU, independientemente de si la CU utiliza un vector disparidad resultante del proceso de derivación de NBDV. En otras palabras, cuando un segmento no está intracodificado, se invoca un NBDV para cada CU independientemente de si la CU usa o no el resultado de NBDV. El codificador de vídeo puede usar el resultado NBDV para actualizar el DDV, lo que significa que el codificador de vídeo puede establecer que el DDV sea igual al vector de disparidad derivado mediante el proceso de derivación de NBDV.

[0155] En otro ejemplo de cómo un codificador de vídeo puede actualizar un DDV cuando las técnicas de la presente divulgación se generalizan para la 3D-HEVC, el codificador de vídeo puede invocar un proceso de derivación de NBDV después de que el codificador de vídeo haya utilizado la predicción inter para codificar una CU. En otras palabras, una vez que se ha intercodificado una CU, se invoca el NBDV para la CU. El codificador de vídeo puede utilizar el resultado de NBDV (es decir, el vector de disparidad derivado mediante el proceso de derivación de NBDV) para actualizar el DDV. De lo contrario, si el codificador de vídeo no utiliza la predicción inter para codificar la CU, el codificador de vídeo no actualiza el DDV.

[0156] En otro ejemplo de cómo un codificador de vídeo puede actualizar un DDV cuando las técnicas de la presente divulgación se generalizan para la 3D-HEVC, el codificador de vídeo puede actualizar el DDV usando un resultado de NBDV (es decir, un vector de disparidad derivado mediante un proceso de derivación de NBDV) cuando el codificador de vídeo codifica una PU de una UC usando el resultado de NBDV. En este ejemplo, el codificador de vídeo no actualiza el DDV cuando el codificador de vídeo codifica una PU sin usar el resultado de NBDV. En otras palabras, una vez que se codifica una PU de una CU usando el resultado de NBDV, el resultado de NBDV se utiliza para actualizar el DDV, de lo contrario, el DDV no se actualiza.

[0157] En otro ejemplo de cómo un codificador de vídeo puede actualizar un DDV cuando las técnicas de la presente divulgación se generalizan para la 3D-HEVC, el codificador de vídeo puede actualizar el DDV para especificar un resultado de NBDV (es decir, un vector de disparidad derivado mediante un proceso de derivación de NBDV) cuando el codificador de vídeo codifica al menos un PU de una CU con el modo de salto. En otras palabras,

en este ejemplo, una vez que al menos una PU de una CU se codifica con el modo de salto, el resultado de NBDV se utiliza para actualizar el DDV.

[0158] En otro ejemplo de punto de una generalización de las técnicas propuestas de la presente divulgación para la 3D-HEVC, el codificador de vídeo puede, en algunos ejemplos, mantener solo un DDV para cada segmento, y el codificador de vídeo puede solo actualizar el DDV una vez que la PU se ha codificado. Por lo tanto, en varios ejemplos de esta divulgación, solo se mantiene un DDV para el segmento, y el DDV se actualiza una vez para cada bloque (macrobloque, CU, PU,etc.) del segmento que se codifica después de que el primer bloque de segmento del segmento se ha codificado. En algunos ejemplos, el codificador de vídeo puede actualizar el DDV de varias maneras. Por ejemplo, el codificador de vídeo puede invocar un proceso de derivación de NBDV para cada PU respectiva de un segmento cuando el segmento no está intracodificado, independientemente de si el codificador de vídeo utiliza un resultado de NBDV para codificar la PU respectiva. En este ejemplo, el codificador de vídeo puede utilizar el resultado de NBDV para actualizar el DDV. En otras palabras, cuando un segmento no está intracodificado, se invoca el NBDV para cada PU independientemente de si la PU usa o no el resultado de NBDV.

[0159] En otro ejemplo de cómo un codificador de vídeo puede actualizar un DDV cuando el codificador de vídeo mantiene solo un DDV para cada segmento, el codificador de vídeo puede invocar un proceso de derivación de NBDV si el codificador de vídeo ha utilizado la predicción inter para codificar una PU. En este ejemplo, el codificador de vídeo puede utilizar el resultado de NBDV para actualizar el DDV. De lo contrario, si el codificador de vídeo no ha utilizado la predicción inter para codificar la PU, el codificador de vídeo no utiliza el resultado de NBDV para actualizar el DDV. En otras palabras, una vez que se intercodifica una PU, se invoca el NBDV para la PU. El resultado de NBDV se utiliza para actualizar el DDV, de lo contrario, el DDV no se actualiza.

[0160] En otro ejemplo de cómo un codificador de vídeo puede actualizar un DDV cuando el codificador de vídeo mantiene solo un DDV para cada segmento, el codificador de vídeo puede utilizar un resultado de NBDV para actualizar el DDV, cuando el codificador de vídeo codifica una PU usando el resultado de NBDV. En este ejemplo, el codificador de vídeo no utiliza el resultado de NBDV para actualizar el DDV cuando el codificador de vídeo no codifica la PU usando el resultado de NBDV. En otras palabras, una vez que se codifica una PU con el resultado de NBDV, el resultado de NBDV se utiliza para actualizar el DDV, de lo contrario, el DDV no se actualiza.

[0161] En otro ejemplo de cómo un codificador de vídeo puede actualizar un DDV cuando el codificador de vídeo mantiene solo un DDV para cada segmento, el codificador de vídeo puede utilizar un resultado de NBDV para actualizar el DDV cuando el codificador de vídeo utiliza el modo de salto para codificar una PU. En otras palabras, una vez que se codifica una PU con el modo de salto, el resultado de NBDV se utiliza para actualizar el DDV.

[0162] En algunos ejemplos en los que las técnicas de la presente divulgación se generalizan para la 3D-HEVC, un codificador de vídeo mantiene un DDV para cada segmento, y el codificador de vídeo actualiza el DDV una vez que el codificador de vídeo codifica una CU o una vez que el codificador de vídeo codifica una PU, el codificador de vídeo puede utilizar el DDV para derivar el vector de disparidad de un bloque (es decir, el NBDV) cuando los bloques vecinos en el espacio y en el tiempo del bloque no tienen vectores de movimiento de disparidad. En otras palabras, el DDV se usa para derivar el NBDV cuando los bloques vecinos en el espacio y en el tiempo no contienen un vector de movimiento de disparidad. En este caso, el codificador de vídeo puede establecer que el NBDV sea igual al DDV. Posteriormente, el codificador de vídeo puede perfeccionar aún más el NBDV, que se establece como DDV, accediendo a la profundidad de la vista de referencia. En algunos ejemplos, el codificador de vídeo puede establecer que el DDV sea igual al vector de disparidad perfeccionado que resulta del acceso a una componente de vista de profundidad de una vista de referencia usando el NBDV original (es decir, el vector de disparidad no perfeccionado del bloque).

[0163] Además, en algunos ejemplos en los que las técnicas de la presente divulgación se aplican a la 3D-AVC o la 3D-HEVC, el codificador de vídeo puede solo almacenar y actualizar la componente horizontal de los DDV, mientras que el codificador de vídeo puede siempre establecer la componente vertical de los DDV a 0.

[0164] Uno o más ejemplos de técnicas de esta divulgación pueden generalizarse a cualquier códec de vídeo. Por ejemplo, se puede derivar una variable para cada bloque. La variable se utiliza y es necesaria para decodificar el bloque actual. La variable se puede actualizar después de decodificar cada bloque. La variable que puede ser actualizada por el bloque anterior se tiene en cuenta para predecir el bloque actual. La variable solo se predice a partir de los bloques vecinos y no se señala directa o indirectamente (por ejemplo, utilizando codificación diferencial).

[0165] La FIG. 7 es un diagrama de bloques que ilustra un ejemplo de codificador de vídeo 20 que puede implementar una o más técnicas de esta divulgación. En el ejemplo de la FIG. 7, el decodificador de vídeo 20 incluye una unidad de procesamiento de predicción 48, un sumador 50, una unidad de procesamiento de transformada 52, una unidad de cuantificación 54, una unidad de codificación de entropía 56, una unidad de cuantificación inversa 58, una unidad de procesamiento de transformada inversa 60, un sumador 62, una unidad de filtro 64 y una memoria de imágenes de referencia 66. La unidad de procesamiento de predicción 48 incluye una unidad de estimación de movimiento 68 (es decir, una unidad de compensación de movimiento y disparidad), una unidad de compensación de movimiento 70 (es decir, una unidad de compensación de movimiento y disparidad) y una unidad de

procesamiento de predicción intra 72.

[0166] Como se muestra en la FIG. 7, el codificador de vídeo 20 recibe datos de vídeo y divide los datos en bloques de vídeo. Esta división también puede incluir la división de imágenes en segmentos, mosaicos u otras unidades mayores, así como la división en bloques de vídeo, por ejemplo, de acuerdo con una estructura de árbol cuaternario de LCU y CU o mediante división de macrobloques en el caso de la HEVC. El codificador de vídeo 20 ilustra en general los componentes que codifican bloques de vídeo de un segmento que se va a codificar. El segmento puede dividirse en varios bloques de vídeo (y, posiblemente, en conjuntos de bloques de vídeo denominados mosaicos).

[0167] La unidad de procesamiento de predicción 48 puede seleccionar uno de una pluralidad de posibles modos de codificación, tal como uno de una pluralidad de modos de codificación intra o uno de una pluralidad de modos de codificación inter o de modos de codificación entre vistas, para el bloque de vídeo actual, basándose en resultados de errores (por ejemplo, la velocidad de codificación y el nivel de distorsión). La unidad de procesamiento de predicción 48 puede realizar la codificación intra e inter de bloques de vídeo de segmentos. La codificación inter (es decir, la predicción inter temporal) puede basarse en la predicción temporal para reducir o eliminar la redundancia temporal en el vídeo de tramas o imágenes adyacentes de una secuencia de vídeo. La codificación intra (es decir, el "modo intra" o el "modo I") puede referirse a cualquiera de varios modos de compresión espacial. Los modos inter, tales como la predicción unidireccional (es decir, "unipredicción" o "modo P") o la predicción bidireccional (es decir, "bipredicción" o "modo B"), pueden referirse a cualquiera de varios modos de compresión temporal. Además, la unidad de procesamiento de predicción 48 puede realizar predicción entre vistas entre imágenes de diferentes vistas, como se ha descrito anteriormente. La unidad de procesamiento de predicción 48 puede proporcionar el bloque intracodificado o intercodificado resultante al sumador 50 para generar datos de bloques residuales, y al sumador 62 para reconstruir el bloque original para su uso como imagen de referencia.

[0168] La unidad de procesamiento de predicción intra 72, de la unidad de procesamiento de predicción 48 puede realizar la codificación de predicción intra (es decir, la codificación intra) del bloque de vídeo actual con respecto a uno o más bloques vecinos de la misma trama o segmento que el bloque actual que se va a codificar, para proporcionar compresión espacial. Dicho de otro modo, la codificación intra se basa en la predicción espacial para reducir o eliminar redundancia espacial en el vídeo de una trama o imagen de vídeo dada. La unidad de estimación de movimiento 68 y la unidad de compensación de movimiento 70 de la unidad de procesamiento de predicción 48 realizan la codificación de predicción inter y/o la codificación entre vistas del bloque de vídeo actual en relación con uno o más bloques predictivos de una o más imágenes de referencia y/o vistas de referencia, para proporcionar compresión temporal y entre vistas.

[0169] La unidad de procesamiento de predicción 48 puede determinar el modo de predicción inter y/o el modo de predicción entre vistas para un segmento de acuerdo con un patrón predeterminado para una secuencia de vídeo. El patrón predeterminado puede designar segmentos de la secuencia como segmentos P o segmentos B. La unidad de estimación de movimiento 68 y la unidad de compensación de movimiento 70 pueden estar sumamente integradas, pero se ilustran por separado con fines conceptuales. La unidad de estimación de movimiento 68 puede realizar estimación de movimiento y/o estimación de disparidad. La estimación de movimiento comprende el proceso de generación de vectores de movimiento, que estiman el movimiento para bloques de vídeo. Un vector de movimiento, por ejemplo, puede indicar el desplazamiento de un bloque actual (por ejemplo, una PU, un MB, una partición de MB, etc.) de una trama o imagen de vídeo actual con respecto a un bloque predictivo de una imagen de referencia. La estimación de disparidad es el proceso de generación de vectores de disparidad, que puede usarse para predecir un bloque codificado actualmente a partir de un bloque de una vista diferente.

[0170] Un bloque predictivo puede ser un bloque que se comprueba que guarda una estrecha coincidencia con un bloque actual (por ejemplo, una PU, un MB, una partición de MB, etc.) que se va a codificar en términos de diferencias de píxeles, que pueden determinarse mediante una suma de diferencias absolutas (SAD), una suma de diferencias al cuadrado (SSD) u otras métricas de diferencias. En algunos ejemplos, el codificador de vídeo 20 puede calcular valores para posiciones de píxel de subentero de imágenes de referencia almacenadas en la memoria de imágenes de referencia 66. La memoria de imágenes de referencia 66 también se puede denominar memoria intermedia de imágenes decodificadas. Por ejemplo, el codificador de vídeo 20 puede interpolar valores de posiciones de un cuarto de píxel, posiciones de un octavo de píxel u otras posiciones de píxel fraccionario de la imagen de referencia. Por lo tanto, la unidad de estimación de movimiento 68 puede realizar una búsqueda de movimiento en relación con las posiciones de píxel completo y las posiciones de píxel fraccionario, y facilitar un vector de movimiento con una precisión de píxel fraccionario.

[0171] La unidad de procesamiento de predicción 48 puede calcular un vector de movimiento (para predicción con compensación de movimiento) y/o un vector de disparidad (para predicción con compensación de disparidad) para un bloque (por ejemplo, una PU, un MB, una partición de MB, etc.) de un segmento sometido a codificación inter o predicción entre vistas, comparando la posición del bloque con la posición de un bloque predictivo de una imagen de referencia. La imagen de referencia puede seleccionarse entre una primera lista de imágenes de referencia (Lista 0) o una segunda lista de imágenes de referencia (Lista 1), cada una de las cuales identifica una o más imágenes de referencia almacenadas en la memoria de imágenes de referencia 66. Para la predicción entre vistas, la imagen de referencia está en una vista diferente. La unidad de estimación de movimiento 68 puede enviar el vector de

movimiento y/o el vector de disparidad calculados a la unidad de codificación de entropía 56 y la unidad de compensación de movimiento 70.

[0172] La unidad de compensación de movimiento 70 puede llevar a cabo la compensación de movimiento y/o la compensación de disparidad. La compensación de movimiento y/o la compensación de disparidad pueden implicar obtener o generar el bloque predictivo basándose en el vector de movimiento determinado mediante estimación de movimiento y/o estimación de paridad, realizando posiblemente interpolaciones hasta la precisión de subpíxel. Tras recibir el vector de movimiento y/o la disparidad para un bloque (por ejemplo, una PU, un MB, una partición de MB, etc.), la unidad de compensación de movimiento 70 puede localizar el bloque predictivo al que apunta el vector de movimiento y/o el vector de disparidad en una de las listas de imágenes de referencia. La unidad de procesamiento de predicción 48 también puede generar elementos sintácticos asociados a los bloques de vídeo y el segmento, para su uso por el decodificador de vídeo 30 en la decodificación de los bloques de vídeo del segmento.

[0173] El sumador 50 puede formar un bloque de vídeo residual restando valores de píxeles del bloque predictivo a los valores de píxeles correspondientes del bloque de vídeo actual que se está codificando, generando de ese modo valores de diferencias de píxeles. Los valores de diferencias de píxeles forman datos residuales para el bloque de vídeo actual. Los valores de diferencias de píxeles pueden incluir tanto valores de diferencia de luma como valores de diferencia de croma. El sumador 50 puede representar el componente o los componentes que realizan esta operación de sustracción.

[0174] La unidad de procesamiento de predicción intra 72 puede realizar la predicción intra para generar uno o más bloques predictivos para un bloque de vídeo actual, como alternativa a la predicción inter llevada a cabo por la unidad de estimación de movimiento 68 y la unidad de compensación de movimiento 70, como se ha descrito anteriormente. Por ejemplo, la unidad de procesamiento de predicción intra 72 puede determinar un modo de predicción intra para generar el uno o más bloques predictivos para el bloque de vídeo actual. Después de determinar (es decir, seleccionar) un modo de predicción intra para un bloque de vídeo actual, la unidad de procesamiento de predicción intra 72 puede proporcionar información (por ejemplo, un elemento sintáctico) indicativa del modo de predicción intra seleccionado para el bloque de vídeo actual a la unidad de codificación de entropía 56. La unidad de codificación de entropía 56 puede codificar la información que indica el modo de predicción intra seleccionada.

[0175] En algunos ejemplos, la unidad de procesamiento de predicción intra 72 puede codificar un bloque de vídeo actual (es decir, la unidad de procesamiento de predicción intra 72 puede generar uno o más bloques predictivos para el bloque de vídeo actual) usando varios modos de predicción intra. Por ejemplo, la unidad de procesamiento de predicción intra 72 puede generar bloques predictivos para el bloque de vídeo actual durante pasadas de codificación separadas. La unidad de procesamiento de predicción intra 72 puede seleccionar entre los modos probados un modo de predicción intra apropiado para usar. Por ejemplo, la unidad de procesamiento de predicción intra 72 puede calcular valores de velocidad-distorsión usando un análisis de velocidad-distorsión para los diversos modos de predicción intra probados, y seleccionar el modo de predicción intra que tiene las mejores características de velocidad-distorsión entre los modos de predicción intra probados. En general, el análisis de velocidad-distorsión determina una cantidad de distorsión (o error) entre un bloque codificado y un bloque original no codificado que se ha codificado para generar el bloque codificado, así como una velocidad de bits (es decir, un número de bits) utilizada para generar el bloque codificado. La unidad de procesamiento de predicción intra 72 puede calcular proporciones a partir de las distorsiones y velocidades para los diversos bloques codificados, a fin de determinar qué modo de predicción intra presenta el mejor valor de velocidad-distorsión para el bloque predictivo.

[0176] En algunos ejemplos, la unidad de estimación de movimiento 68 puede determinar un vector de disparidad para el bloque de vídeo actual. La unidad de estimación de movimiento 68 puede utilizar el vector de disparidad para el bloque actual para realizar la predicción del vector de movimiento para el bloque de vídeo actual. Por ejemplo, la unidad de estimación de movimiento 68 puede utilizar el vector de disparidad para el bloque actual para determinar un bloque correspondiente en una imagen de referencia entre vistas. Además, la unidad de estimación de movimiento 68 puede generar una lista de candidatos de vectores de movimiento (por ejemplo, una lista de candidatos de fusión o una lista de candidatos de AMVP) para el bloque de vídeo actual. La lista de candidatos de vectores de movimiento puede incluir uno o más candidatos que especifican uno o más vectores de movimiento del correspondiente bloque de referencia entre vistas. La unidad de compensación de movimiento 70 puede determinar bloques predictivos basándose en los candidatos y puede seleccionar uno de los candidatos basándose en un análisis de velocidad-distorsión de los bloques predictivos y otros datos. La unidad de compensación de movimiento 70 puede generar uno o más elementos sintácticos que indican el candidato seleccionado. Además, en algunos ejemplos, la unidad de compensación de movimiento 70 puede usar el vector de disparidad para el bloque de vídeo actual a fin de realizar una predicción residual (por ejemplo, una predicción residual avanzada) para el bloque de vídeo actual. Por lo tanto, en dichos ejemplos, el bloque predictivo generado en última instancia por la unidad de compensación de movimiento 70 puede estar basado en una diferencia entre un bloque predictivo para el bloque de vídeo actual y un indicador residual para el bloque de vídeo actual.

[0177] La unidad de estimación de movimiento 68 puede usar un proceso de derivación de NBDV para determinar el vector de disparidad para el bloque de vídeo actual. De acuerdo con una o más técnicas de esta divulgación, la

unidad de estimación de movimiento 68 puede inicializar uno o más DDV para el segmento actual. La unidad de estimación de movimiento 68 puede usar uno o más DDV para el segmento actual en el proceso de derivación de NBDV a fin de determinar el vector de disparidad para el bloque de vídeo actual. En algunos ejemplos, la unidad de estimación de movimiento 68 inicializa un único DDV para el segmento actual y actualiza el DDV de tal forma que, para cada bloque de vídeo respectivo del segmento actual después del primer bloque de vídeo del segmento actual, el DDV del segmento actual especifica el vector de disparidad para el bloque de vídeo que se halla justo antes del bloque de vídeo respectivo por orden de codificación.

[0178] De esta manera, la unidad de estimación de movimiento 68 puede almacenar un DDV por un segmento actual de una imagen actual. Además, la unidad de estimación de movimiento 68 puede derivar, basándose al menos en parte en el DDV, un NBDV para un primer bloque de vídeo del segmento actual. La unidad de compensación de movimiento 70 y otras unidades del codificador de vídeo 20 pueden codificar el primer bloque de vídeo basándose al menos en parte en el NBDV para el primer bloque de vídeo. Además, la unidad de estimación de movimiento 68 puede almacenar el NBDV para el primer bloque de vídeo como DDV, actualizando así el DDV. Después de almacenar el NBDV para el primer bloque de vídeo como DDV, la unidad de estimación de movimiento 68 puede derivar, basándose al menos en parte en el DDV actualizado, un NBDV para un segundo bloque de vídeo del segmento actual. La unidad de compensación de movimiento 70 y otras unidades del codificador de vídeo 20 pueden codificar el segundo bloque de vídeo basándose al menos en parte en el NBDV para el segundo bloque de vídeo. La unidad de estimación de movimiento 68, la unidad de compensación de movimiento 70 y otras unidades del codificador de vídeo 20 pueden continuar realizando estas acciones para bloques de vídeo adicionales del segmento actual.

[0179] Después de que la unidad de procesamiento de predicción 48 genere un bloque predictivo para un bloque de vídeo actual, ya sea mediante predicción inter o predicción intra, el sumador 50 puede formar un bloque residual restando valores de muestras del bloque predictivo a unos valores de muestras correspondientes del bloque de vídeo actual. Los datos de vídeo residuales del bloque residual pueden estar incluidos en una o más TU. La unidad de procesamiento de transformada 52 puede utilizar una transformada para transformar los datos de vídeo residuales en coeficientes de transformada residuales. Por ejemplo, la unidad de procesamiento de transformada 52 puede usar una transformada de coseno discreta (DCT) o una transformada conceptualmente similar para transformar los datos de vídeo residuales. En algunos ejemplos, la unidad de procesamiento de transformada 52 puede convertir los datos de vídeo residuales de un dominio de píxel a un dominio de transformada, tal como un dominio de frecuencia.

[0180] La unidad de procesamiento de transformada 52 puede enviar los coeficientes de transformada resultantes a la unidad de cuantificación 54. La unidad de cuantificación 54 puede cuantificar los coeficientes de transformada generados por la unidad de procesamiento de transformada 52 a fin de reducir aún más la velocidad de bits. El proceso de cuantificación puede reducir la profundidad de bits asociada a algunos o la totalidad de los coeficientes. El grado de cuantificación puede modificarse ajustando un parámetro de cuantificación. En algunos ejemplos, la unidad de cuantificación 54 puede realizar una exploración de la matriz que incluye los coeficientes de transformada cuantificados. De forma alternativa, la unidad de codificación de entropía 56 puede llevar a cabo la exploración.

[0181] Tras la cuantificación, la unidad de codificación de entropía 56 puede realizar la codificación de entropía de los coeficientes de transformada cuantificados. En otras palabras, la unidad de codificación de entropía 56 puede realizar la codificación de entropía de elementos sintácticos que representan los coeficientes de transformada cuantificados. Por ejemplo, la unidad de codificación de entropía 56 puede realizar la codificación de entropía de los coeficientes de transformada cuantificados realizando una codificación de longitud variable adaptativa según el contexto (CAVLC), una codificación aritmética binaria adaptativa según el contexto (CABAC), una codificación aritmética binaria adaptativa según el contexto basada en la sintaxis (SBAC), una codificación de entropía por división de intervalos de probabilidad (PIPE) u otros procedimientos o técnicas de codificación de entropía. La unidad de codificación de entropía 56 también puede realizar la codificación de entropía de otros elementos sintácticos para el segmento actual que se está codificando.

[0182] El codificador de vídeo 20 puede generar un flujo de bits codificado que incluye una representación codificada de los datos de vídeo. El flujo de bits puede incluir elementos sintácticos sometidos a codificación de entropía generados por la unidad de codificación de entropía 56 y otros elementos sintácticos sometidos o no a codificación de entropía. El flujo de bits codificado puede transmitirse al decodificador de vídeo 30 o archivarse para su posterior transmisión o recuperación por el decodificador de vídeo 30.

[0183] La unidad de cuantificación inversa 58 y la unidad de procesamiento de transformada inversa 60 pueden aplicar una cuantificación inversa y una transformada inversa, respectivamente, para reconstruir el bloque residual en el dominio del píxel, para su posterior uso como bloque de referencia de una imagen de referencia. El sumador 62 puede reconstruir bloques de muestra sumando muestras, en bloques residuales generados por la unidad de procesamiento de transformada inversa 60, a muestras correspondientes de uno o más bloques predictivos generados por la unidad de procesamiento de predicción 48. En algunos ejemplos, el sumador 62 suma el bloque residual reconstruido a un bloque predictivo (por ejemplo, un bloque predictivo con compensación de movimiento generado por la unidad de compensación de movimiento 44) para generar un bloque de referencia para su

almacenamiento en la memoria de imágenes de referencia 66. De este modo, la memoria de imágenes de referencia 66 puede ser una memoria que almacena datos de vídeo. La unidad de estimación de movimiento 68 y la unidad de compensación de movimiento 70 pueden usar el bloque de referencia como bloque de referencia para realizar la predicción inter de un bloque en una trama o imagen de vídeo subsiguiente. La unidad de compensación de movimiento 70 puede calcular un bloque de referencia sumando el bloque residual a un bloque predictivo de una de las imágenes de referencia de una de las listas de imágenes de referencia. La unidad de compensación de movimiento 70 también puede aplicar uno o más filtros de interpolación al bloque residual reconstruido para calcular valores de píxel subentero y usarlos en la estimación de movimiento.

[0184] En el ejemplo de la FIG. 7, la unidad de filtro 64 puede aplicar uno o más filtros al bloque de muestra generado por el sumador 62. Por ejemplo, la unidad de filtro 64 puede aplicar un filtro de eliminación de bloques para filtrar los límites de bloque a fin de eliminar distorsiones de efecto pixelado del vídeo reconstruido. En algunos ejemplos, la unidad de filtro 64 utiliza filtros de bucle adicionales (en bucle o posbucle) además del filtro de eliminación de bloques.

[0185] La FIG. 8 es un diagrama de bloques que ilustra un ejemplo de decodificador de vídeo 30 que puede implementar una o más técnicas descritas en esta divulgación. En el ejemplo de la FIG. 8, el decodificador de vídeo 30 incluye una unidad de decodificación de entropía 80, una unidad de procesamiento de predicción 82, una unidad de cuantificación inversa 84, una unidad de procesamiento de transformada inversa 86, un sumador 88 y una memoria de imágenes de referencia 90. La unidad de procesamiento de predicción 82 incluye una unidad de compensación de movimiento 92 (es decir una unidad de compensación de movimiento y disparidad) y una unidad de procesamiento de predicción intra 94. En algunos ejemplos, el decodificador de vídeo 30 puede realizar una pasada de decodificación en general recíproca a la pasada de codificación descrita con respecto al codificador de vídeo 20 de la FIG. 7.

[0186] El decodificador de vídeo 30 recibe un flujo de bits de vídeo codificado que representa bloques de vídeo de un segmento codificado y elementos sintácticos asociados, desde el codificador de vídeo 20. La unidad de decodificación de entropía 80 del decodificador de vídeo 30 puede realizar la decodificación de entropía del flujo de bits para generar coeficientes cuantificados, vectores de movimiento, vectores de disparidad y otros elementos sintácticos. Más específicamente, la unidad de decodificación de entropía 80 puede realizar un proceso de análisis sintáctico para obtener elementos sintácticos a partir del flujo de bits de vídeo. Como parte de la realización del proceso de análisis sintáctico, la unidad de decodificación de entropía 80 puede realizar la decodificación de entropía de elementos sintácticos sometidos a codificación de entropía señalizados en el flujo de bits de vídeo. La unidad de decodificación de entropía 80 puede reenviar los elementos sintácticos asociados con vectores de movimiento, vectores de disparidad y otra información a la unidad de procesamiento de predicción 82. El decodificador de vídeo 30 puede recibir los elementos sintácticos en el nivel del segmento y/o el nivel de bloque de vídeo.

[0187] Cuando el segmento actual se codifica como un segmento intracodificado (por ejemplo, un segmento I, un segmento SI, etc.), la unidad de procesamiento de predicción intra 94 de la unidad de procesamiento de predicción 82 puede generar datos de predicción para un bloque de vídeo del segmento actual, basándose en un modo de predicción intra señalizado y unos datos de bloques previamente decodificados de la trama o imagen actual. Cuando el segmento actual está intercodificado (es decir, es un segmento P, un segmento B, un segmento SP, etc.), la unidad de compensación de movimiento 92 de la unidad de procesamiento de predicción 82 puede producir (es decir, generar) unos bloques predictivos para un bloque de vídeo del segmento actual, basándose en los vectores de movimiento, vectores de disparidad y elementos sintácticos recibidos desde la unidad de decodificación de entropía 80.

[0188] La unidad de procesamiento de predicción 82 del decodificador de vídeo 30 puede confeccionar listas de tramas de referencia, Lista 0 y Lista 1, usando (por ejemplo, de forma predeterminada) técnicas de construcción basadas en imágenes de referencia almacenadas en la memoria de imágenes de referencia 90 (que también se conoce como memoria intermedia de imágenes decodificadas (DPB)). La unidad de compensación de movimiento 92 puede generar bloques predictivos a partir de una o más imágenes de referencia de una o más de las listas de imágenes de referencia.

[0189] La unidad de compensación de movimiento 92 puede determinar información de predicción para un bloque de vídeo del segmento actual, analizando los vectores de movimiento y otros elementos sintácticos, y puede utilizar la información de predicción para generar los bloques predictivos para el bloque de vídeo actual que se está decodificando. Por ejemplo, la unidad de compensación de movimiento 92 puede usar algunos de los elementos sintácticos recibidos para determinar un modo de predicción (por ejemplo, predicción intra o predicción inter), para usar a fin de codificar los bloques de vídeo del segmento, elementos sintácticos para determinar un tipo de segmento de predicción inter o predicción entre vistas (por ejemplo, segmento B, segmento P o segmento GPB), elementos sintácticos para determinar información de construcción para una o más de las listas de imágenes de referencia para el segmento, elementos sintácticos para determinar vectores de movimiento y/o vectores de disparidad para cada bloque de vídeo intercodificado del segmento, elementos sintácticos para determinar un estado de predicción inter para cada bloque de vídeo intercodificado del segmento y elementos sintácticos para determinar

otra información para decodificar los bloques de vídeo del segmento actual.

[0190] La unidad de compensación de movimiento 92 también puede realizar la interpolación basándose en filtros de interpolación. La unidad de compensación de movimiento 92 puede usar filtros de interpolación como los usados por el codificador de vídeo 20 durante la codificación de los bloques de vídeo, para calcular valores interpolados para píxeles de subentero de bloques de referencia. En algunos ejemplos, la unidad de compensación de movimiento 92 puede determinar los filtros de interpolación utilizados por el codificador de vídeo 20 a partir de los elementos sintácticos recibidos, y puede utilizar los filtros de interpolación para generar bloques predictivos.

[0191] En algunos ejemplos, la unidad de compensación de movimiento 92 puede determinar un vector de disparidad para el bloque de vídeo actual. La unidad de compensación de movimiento 92 puede utilizar el vector de disparidad para el bloque actual a fin de realizar la predicción del vector de movimiento para el bloque de vídeo actual. Por ejemplo, la unidad de compensación de movimiento 92 puede utilizar el vector de disparidad para el bloque actual a fin de determinar un bloque correspondiente en una imagen de referencia entre vistas. Además, la unidad de compensación de movimiento 92 puede generar una lista de candidatos de vectores de movimiento (por ejemplo, una lista de candidatos de fusión o una lista de candidatos de AMVP) para el bloque de vídeo actual. La lista de candidatos de vectores de movimiento puede incluir uno o más candidatos que especifican uno o más vectores de movimiento del correspondiente bloque de referencia entre vistas. La unidad de compensación de movimiento 92 puede determinar, basándose en uno o más elementos sintácticos obtenidos del flujo de bits, un candidato seleccionado en la lista de candidatos. La unidad de compensación de movimiento 92 puede entonces determinar uno o más vectores de movimiento para el bloque de vídeo actual basándose en uno o más vectores de movimiento especificados por el candidato seleccionado. Por ejemplo, si la lista de candidatos es una lista de candidatos de fusión, la unidad de compensación de movimiento 92 puede determinar que un vector de movimiento del bloque de vídeo actual coincide con un vector de movimiento especificado por el candidato seleccionado. Si la lista de candidatos es una lista de candidatos de AMVP, la unidad de compensación de movimiento 92 puede determinar que un vector de movimiento del bloque de vídeo actual es igual a una suma de una diferencia de vector de movimiento (MVD) señalada en el flujo de bits y un vector de movimiento especificado por el candidato seleccionado.

[0192] En algunos ejemplos, la unidad de compensación de movimiento 70 puede usar el vector de disparidad para el bloque de vídeo actual a fin de llevar a cabo la predicción residual (por ejemplo, Predicción Residual Avanzada) para el bloque de vídeo actual. En dichos ejemplos, la unidad de compensación de movimiento 92 puede utilizar el vector de disparidad para determinar un indicador residual para el bloque actual. El bloque predictivo generado en última instancia por la unidad de compensación de movimiento 92 puede basarse en una diferencia entre un bloque predictivo para el bloque de vídeo actual y el indicador residual para el bloque de vídeo actual.

[0193] La unidad de compensación de movimiento 92 puede usar un proceso de derivación de NBDV para determinar el vector de disparidad para el bloque de vídeo actual. De acuerdo con una o más técnicas de esta divulgación, la unidad de compensación de movimiento 92 puede inicializar uno o más DDV para el segmento actual. La unidad de compensación de movimiento 92 puede usar el uno o más DDV para el segmento actual en el proceso de derivación de NBDV para determinar el vector de disparidad para el bloque de vídeo actual. En algunos ejemplos, la unidad de compensación de movimiento 92 inicializa un único DDV para el segmento actual y actualiza el DDV de tal forma que, para cada bloque de vídeo respectivo del segmento actual después del primer bloque de vídeo del segmento actual, el DDV para el segmento actual especifica el vector de disparidad para el bloque de vídeo que se halla justo antes del bloque de vídeo respectivo por orden de codificación.

[0194] De esta manera, la unidad de compensación de movimiento 92 puede almacenar un DDV para un segmento actual de una imagen actual. Además, la unidad de compensación de movimiento 92 puede derivar, basándose al menos en parte en el DDV, un NBDV para un primer bloque de vídeo del segmento actual. La unidad de compensación de movimiento 92 y otras unidades del decodificador de vídeo 30 pueden decodificar el primer bloque de vídeo basándose al menos en parte en el NBDV para el primer bloque de vídeo. Además, la unidad de compensación de movimiento 92 puede almacenar el NBDV para el primer bloque de vídeo como DDV, actualizando así el DDV. Después de almacenar el NBDV para el primer bloque de vídeo como DDV, la unidad de compensación de movimiento 92 puede derivar, basándose al menos en parte en el DDV actualizado, un NBDV para un segundo bloque de vídeo del segmento actual. La unidad de compensación de movimiento 92 y otras unidades del decodificador de vídeo 30 pueden decodificar el segundo bloque de vídeo basándose al menos en parte en el NBDV para el segundo bloque de vídeo. La unidad de compensación de movimiento 92 y otras unidades del decodificador de vídeo 30 pueden continuar realizando estas acciones para bloques de vídeo adicionales del segmento actual.

[0195] La unidad de cuantificación inversa 84 puede realizar un proceso de cuantificación inversa para realizar la cuantificación inversa, es decir, la descuantificación, de los coeficientes de transformada cuantificados señalados en el flujo de bits. El proceso de cuantificación inversa puede incluir el uso de un parámetro de cuantificación calculado por el codificador de vídeo 20 para cada bloque de vídeo en el segmento de vídeo, a fin de determinar un grado de cuantificación y, asimismo, un grado de cuantificación inversa que debería aplicarse. La unidad de procesamiento de transformada inversa 86 puede aplicar una transformada inversa a los coeficientes de transformada, con el fin de generar bloques residuales en el dominio del píxel. Por ejemplo, la unidad de procesamiento de transformada

inversa 86 puede aplicar una DCT inversa, una transformada de número entero inversa o un proceso de transformada inversa conceptualmente similar para transformar los coeficientes de transformada.

[0196] Una vez que la unidad de compensación de movimiento 92 ha generado un bloque predictivo para un bloque de vídeo actual, el decodificador de vídeo 30 pueden formar un bloque de vídeo decodificado sumando unos valores de muestras de unos bloques residuales generados por la unidad de procesamiento de transformada inversa 86 con los valores de muestras correspondientes de unos bloques predictivos generados por la unidad de compensación de movimiento 92. El sumador 88 representa el componente o los componentes que llevan a cabo esta operación de suma.

[0197] Si se desea, también puede aplicarse un filtro de eliminación de bloques para filtrar los bloques decodificados con el fin de eliminar distorsiones de efecto pixelado. También pueden utilizarse otros filtros de bucle (ya sea en el bucle de codificación o después del bucle de codificación) para suavizar las transiciones de píxeles o mejorar de otro modo la calidad del vídeo. Los bloques de vídeo decodificados de una trama o imagen dada se almacenan a continuación en una memoria de imágenes de referencia 90 (a veces denominada memoria intermedia de imágenes decodificadas), que almacena imágenes de referencia usadas para una subsiguiente compensación de movimiento. La memoria de imágenes de referencia 90 puede almacenar también datos de vídeo decodificado para su presentación en un dispositivo de visualización, tal como el dispositivo de visualización 32 de la FIG. 1. De este modo, la memoria de imágenes de referencia 90 puede ser una memoria que almacena datos de vídeo.

[0198] La FIG. 9A es un diagrama de flujo que ilustra un ejemplo de operación del codificador de vídeo 20 de acuerdo con un ejemplo de esta divulgación. En otros ejemplos, el codificador de vídeo 20 puede realizar operaciones similares a las de la FIG. 9A, pero que incluyen más, menos o diferentes acciones. Por ejemplo, en otros ejemplos, una o más acciones de la FIG. 9A pueden omitirse, reordenarse o repetirse. La FIG. 9A se describe con referencia a la FIG. 7. No obstante, el ejemplo de la FIG. 9A no está tan limitado.

[0199] En el ejemplo de la FIG. 9A, la unidad de procesamiento de predicción 48 puede almacenar solo un DDV para un segmento de una imagen actual de los datos de vídeo (152). El segmento incluye varios bloques. Además, la unidad de procesamiento de predicción 48 puede usar el DDV para el segmento en un proceso de derivación de NBDV para determinar un vector de disparidad para un bloque particular (154). El bloque particular es uno de los bloques del segmento. Además, la unidad de procesamiento de predicción 48 puede almacenar, como DDV para el segmento, el vector de disparidad (es decir, el NBDV) para el bloque particular (156).

[0200] Además, en el ejemplo de la FIG. 9A, el codificador de vídeo 20 puede codificar el bloque particular basándose al menos en parte en el vector de disparidad para el bloque particular (158). Por ejemplo, el codificador de vídeo 20 puede utilizar el NBDV para el bloque particular a fin de realizar una predicción de movimiento entre vistas y/o una predicción residual entre vistas para generar una representación codificada del bloque particular. En algunos ejemplos, el codificador de vídeo 20 continúa la operación de la FIG. 9A con respecto a unos bloques adicionales del segmento. Además, en algún ejemplo, después de almacenar el vector de disparidad para el bloque particular (es decir, un primer bloque) como DDV para el segmento, la unidad de procesamiento de predicción 48 puede usar el DDV para el segmento en un proceso de derivación de NBDV a fin de determinar un vector de disparidad para un segundo bloque del segmento. El codificador de vídeo 20 puede codificar el segundo bloque basándose al menos en parte en el vector de disparidad para el segundo bloque. En algunos ejemplos, el primer bloque y el segundo bloque son macrobloques. En otros ejemplos, el primer bloque y el segundo bloque son CU, el primer bloque y el segundo bloque son PU o el primer bloque y el segundo bloque son LCU.

[0201] La FIG. 9B es un diagrama de flujo que ilustra un ejemplo de operación del decodificador de vídeo 30, de acuerdo con un ejemplo de esta divulgación. En otros ejemplos, el decodificador de vídeo 30 puede realizar operaciones similares a las de la FIG. 9B, pero que incluyen más, menos o diferentes acciones. Por ejemplo, en otros ejemplos, una o más acciones de la FIG. 9A pueden omitirse, reordenarse o repetirse. La FIG. 9B se describe con referencia a la FIG. 8. No obstante, el ejemplo de la FIG. 9B no está tan limitado.

[0202] En el ejemplo de la FIG. 9B, la unidad de procesamiento de predicción 82 puede almacenar solo un DDV para un segmento de una imagen actual de los datos de vídeo (172). El segmento incluye varios bloques. Además, la unidad de procesamiento de predicción 82 puede usar el DDV para el segmento en un proceso de derivación de NBDV para determinar un vector de disparidad para un bloque particular (174). El bloque particular es uno de los bloques del segmento. Además, la unidad de procesamiento de predicción 82 puede almacenar, como DDV para el segmento, el vector de disparidad (es decir, el NBDV) para el bloque particular (176).

[0203] Además, en el ejemplo de la FIG. 9B, el decodificador de vídeo 30 puede decodificar el bloque particular basándose al menos en parte en el vector de disparidad (es decir, el NBDV) para el bloque particular (178). Por ejemplo, el decodificador de vídeo 30 puede usar el vector de disparidad para el bloque particular a fin de realizar una predicción de movimiento entre vistas y/o una predicción residual entre vistas como parte de un proceso para reconstruir un bloque de muestras para el bloque particular. En algunos ejemplos, después de almacenar el vector de disparidad para el bloque particular (es decir, un primer bloque) como DDV para el segmento, la unidad de procesamiento de predicción 82 puede usar el DDV para el segmento en un proceso de derivación de NBDV a fin de

determinar un vector de disparidad para un segundo bloque del segmento. El decodificador de vídeo 30 puede decodificar el segundo bloque basándose al menos en parte en el vector de disparidad para el segundo bloque. Por ejemplo, el decodificador de vídeo 30 puede utilizar el vector de disparidad (es decir, el NBDV) para el segundo bloque a fin de realizar la predicción de movimiento entre vistas y/o la predicción residual entre vistas como parte de un proceso para reconstruir un bloque de muestras para el segundo bloque. En algunos ejemplos, el bloque particular (es decir, un primer bloque) y el segundo bloque son macrobloques. En otros ejemplos, el primer bloque y el segundo bloque son CU, el primer bloque y el segundo bloque son PU o el primer bloque y el segundo bloque son LCU. En algunos ejemplos, el decodificador de vídeo 30 puede continuar la operación de la FIG. 9B con respecto a unos bloques adicionales del segmento.

[0204] La FIG. 10 es un diagrama de flujo que ilustra un ejemplo de operación de derivación de vector de disparidad de acuerdo con un ejemplo de esta divulgación. En otros ejemplos, un codificador de vídeo puede realizar operaciones similares a las de la FIG. 10, pero que incluyen más, menos o diferentes acciones. Por ejemplo, en otros ejemplos, una o más acciones de la FIG. 10 pueden omitirse, reordenarse o repetirse. Un codificador de vídeo, tal como el codificador de vídeo 20 o el decodificador de vídeo 30, puede realizar el ejemplo de operación de derivación de vector de disparidad de la FIG. 10. El codificador de vídeo puede realizar la operación de la FIG. 10 para determinar un vector de disparidad para un bloque actual de un segmento actual de una imagen actual. En diferentes ejemplos, el bloque actual puede ser un macrobloque, una CU, una PU, una LCU u otro tipo de bloque o bloque de vídeo.

[0205] En el ejemplo de la FIG. 10, el codificador de vídeo puede determinar un bloque vecino en el tiempo para el bloque actual (200). El bloque vecino en el tiempo puede ser un bloque de una imagen de referencia de una unidad de acceso diferente a la imagen actual. Además, el codificador de vídeo puede determinar si el bloque vecino en el tiempo tiene un vector de movimiento de disparidad (202). Como respuesta a la determinación de que el bloque vecino en el tiempo tiene un vector de movimiento de disparidad ("SÍ" en 202), el codificador de vídeo puede establecer el vector de disparidad para el bloque actual basándose en el vector de movimiento de disparidad del bloque vecino en el tiempo (204). Por ejemplo, el codificador de vídeo puede establecer la componente horizontal del vector de disparidad para el bloque actual como la componente horizontal del vector de movimiento de disparidad del bloque vecino en el tiempo y puede establecer que la componente vertical del vector de disparidad para el bloque actual sea igual a 0. En otros ejemplos, el codificador de vídeo puede establecer que el vector de disparidad para el bloque actual sea igual al vector de movimiento de disparidad para el bloque vecino en el tiempo.

[0206] Por otro lado, como respuesta a la determinación de que el bloque vecino en el tiempo no tiene un vector de movimiento de disparidad ("NO" en 202), el codificador de vídeo puede determinar si un bloque vecino en el espacio tiene un vector de movimiento de disparidad (206). El bloque vecino en el espacio puede ser un bloque que cubre una o más de las ubicaciones A, B, C o D de la FIG. 5. Como respuesta a la determinación de que el bloque vecino en el espacio tiene un vector de movimiento de disparidad ("SÍ" en 206), el codificador de vídeo puede establecer el vector de disparidad para el bloque actual basándose en el vector de movimiento de disparidad del bloque vecino en el tiempo (208). Por ejemplo, el codificador de vídeo puede establecer la componente horizontal del vector de disparidad para el bloque actual como la componente horizontal del vector de movimiento de disparidad del bloque vecino en el espacio y puede establecer que la componente vertical del vector de disparidad para el bloque actual sea igual a 0. En otros ejemplos, el codificador de vídeo puede establecer que el vector de disparidad para el bloque actual sea igual al vector de movimiento de disparidad para el bloque vecino en el espacio.

[0207] Como respuesta a la determinación de que el bloque vecino en el espacio no tiene un vector de movimiento de disparidad ("NO" en 206), el codificador de vídeo puede determinar si queda algún bloque vecino en el espacio por comprobar (210). Como respuesta a la determinación de que quedan uno o más bloques vecinos en el espacio ("SÍ" en 210), el codificador de vídeo puede repetir la acción (206) y, si procede, la acción (208) con respecto a otro bloque vecino en el espacio. De esta manera, el codificador de vídeo puede comprobar cada uno de los bloques vecinos en el espacio hasta que el decodificador de vídeo determine que uno de los bloques vecinos en el espacio tiene un vector de movimiento de disparidad o que no queda ningún bloque vecino en el espacio por comprobar. En respuesta a la determinación de que no queda ningún bloque vecino en el espacio por comprobar ("NO" en 210), el codificador de vídeo puede establecer el vector de disparidad para el bloque actual basándose en el DDV (212). Por ejemplo, el codificador de vídeo puede establecer que el vector de disparidad para el bloque actual sea igual al DDV.

[0208] Después de establecer el vector de disparidad en (204), (208), o (212), el codificador de vídeo puede establecer el DDV como el vector de disparidad para el bloque actual (214). De esta manera, el codificador de vídeo actualiza el DDV para su uso por un bloque subsiguiente en el segmento actual.

[0209] En uno o más ejemplos, las funciones descritas pueden implementarse en hardware, software, firmware o cualquier combinación de estos. Si se implementan en software, las funciones pueden almacenarse, como una o más instrucciones o código, en un medio legible por ordenador o transmitirse a través de este, y ejecutarse mediante una unidad de procesamiento basada en hardware. Los medios legibles por ordenador pueden incluir medios de almacenamiento legibles por ordenador, que corresponden a un medio tangible tal como unos medios de almacenamiento de datos, o unos 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 ordenador pueden corresponder en general a (1) unos medios de almacenamiento legibles por ordenador tangibles que son no 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 se puede acceder desde uno o más ordenadores o uno o más procesadores para recuperar instrucciones, código y/o estructuras de datos para implementar las técnicas descritas en la presente divulgación. Un producto de programa informático puede incluir un medio legible por ordenador.

[0210] A modo de ejemplo, y no de limitación, dichos medios de almacenamiento legibles por ordenador pueden comprender RAM, ROM, EEPROM, CD-ROM u otro almacenamiento de disco óptico, almacenamiento de disco magnético u otros dispositivos de almacenamiento magnético, memoria flash o cualquier otro medio que pueda utilizarse para almacenar un código de programa deseado en forma de instrucciones o estructuras de datos y al que pueda accederse mediante un ordenador. Además, cualquier conexión recibe adecuadamente la denominación de medios legibles por ordenador. Por ejemplo, si las instrucciones se transmiten desde un sitio web, un servidor u otra fuente remota mediante un cable coaxial, un cable de fibra óptica, un par trenzado, una línea de abonado digital (DSL) o unas tecnologías inalámbricas tales como infrarrojos, radio y microondas, entonces el cable coaxial, el cable de fibra óptica, el par trenzado, la DSL o las tecnologías inalámbricas tales como infrarrojos, radio y microondas se incluyen en la definición de medio. Sin embargo, debería entenderse que los medios de almacenamiento legibles por ordenador y los medios de almacenamiento de datos no incluyen conexiones, ondas portadoras, señales u otros medios transitorios, sino que, en cambio, se orientan a medios de almacenamiento tangibles no transitorios. El término disco, tal como se utiliza en el presente documento, incluye un disco compacto (CD), un disco láser, un disco óptico, un disco versátil digital (DVD), un disco flexible y un disco Blu-ray, de los cuales el disco flexible normalmente reproduce datos de magnéticamente, mientras que el resto de discos reproducen datos ópticamente con láseres. Las combinaciones de lo anterior deberían incluirse también dentro del alcance de los medios legibles por ordenador.

[0211] Las instrucciones pueden ser ejecutadas por uno o más procesadores, tales como uno o más procesadores de señales digitales (DSP), microprocesadores de propósito general, circuitos integrados específicos de la aplicación (ASIC), matrices lógicas programables *in situ* (FP-GA) u otros circuitos lógicos integrados o discretos equivalentes. Por consiguiente, el término "procesador", como se usa en el presente documento, puede referirse a cualquiera de las estructuras anteriores o a cualquier otra estructura adecuada para la implementación de las técnicas descritas en el presente documento. Además, en algunos aspectos, la funcionalidad descrita en el presente documento puede proporcionarse dentro de módulos de hardware y/o software dedicados configurados para la codificación y la decodificación, o incorporarse en un códec combinado. Además, las técnicas podrían implementarse completamente en uno o más circuitos o elementos lógicos.

[0212] Las técnicas de la presente divulgación se pueden implementar en una amplia variedad de dispositivos o aparatos, que incluyen un teléfono inalámbrico, un circuito integrado (CI) o un conjunto de CI (por ejemplo, un conjunto de chips). Aunque en esta divulgación se describen varios componentes, módulos o unidades para enfatizar aspectos funcionales de dispositivos configurados para realizar las técnicas divulgadas, estos no requieren necesariamente su realización mediante diferentes unidades de hardware. En su lugar, como se ha descrito anteriormente, diversas unidades pueden combinarse en una unidad de hardware de códec o proporcionarse por medio de un grupo de unidades de hardware interoperativas, que incluyen uno o más procesadores como los descritos anteriormente, conjuntamente con software y/o firmware adecuados.

REIVINDICACIONES

1. Un procedimiento de decodificación de datos de vídeo, el procedimiento comprendiendo:

almacenar (172) un vector de disparidad de un bloque particular de un segmento de una imagen actual de los datos de vídeo como un vector de disparidad derivado (DDV) para el segmento, en el que el segmento incluye múltiples bloques y sólo hay un DDV para el segmento;
determinar (174), realizando un proceso de derivación de Vector de Disparidad Basado en Bloques Vecinos (NBDV), un vector de disparidad para un bloque actual, en el que el bloque actual es uno de los bloques del segmento, en el que realizar el proceso de derivación de NBDV comprende establecer el vector de disparidad para el bloque actual como el DDV para el segmento cuando el proceso de derivación de NBDV no identifica ningún vector de movimiento de disparidad disponible a partir de bloques vecinos en el tiempo o a partir de bloques vecinos en el espacio; y
almacenar (176), como el DDV para el segmento, el vector de disparidad para el bloque actual.

2. El procedimiento de la reivindicación 1, que comprende además decodificar (178) el bloque actual basándose al menos en parte en el vector de disparidad para el bloque actual.

3. El procedimiento de la reivindicación 1, en el que el bloque actual es un primer bloque, y el procedimiento comprende además, después de almacenar (176), como el DDV para el segmento, el vector de disparidad para el primer bloque:

usar el DDV para el segmento en un proceso de derivación de NBDV para determinar un vector de disparidad para un segundo bloque, en el que el segundo bloque es uno de los bloques del segmento; y
decodificar el segundo bloque basándose al menos en parte en el vector de disparidad para el segundo bloque.

4. El procedimiento de la reivindicación 1, en el que realizar el procedimiento de derivación de NBDV comprende:

determinar si un bloque vecino en el tiempo tiene un vector de movimiento de disparidad;
como respuesta a la determinación de que el bloque vecino en el tiempo tiene un vector de movimiento de disparidad, derivar el vector de disparidad para el bloque actual basándose en el vector de movimiento de disparidad del bloque vecino en el tiempo;
como respuesta a la determinación de que el bloque vecino en el tiempo no tiene un vector de movimiento de disparidad, determinar si algún bloque vecino en el espacio de una pluralidad de bloques vecinos en el espacio tiene un vector de movimiento de disparidad; y
como respuesta a la determinación de que un bloque vecino en el espacio particular de la pluralidad de bloques vecinos en el espacio tiene un vector de movimiento de disparidad, derivar el vector de disparidad para el bloque actual basándose en el vector de movimiento de disparidad del bloque vecino en el espacio particular.

5. El procedimiento de la reivindicación 1, que comprende además actualizar el DDV para el segmento a fin de especificar el vector de disparidad para el bloque actual como respuesta al cumplimiento de una condición, siendo la condición una de las siguientes:

el bloque actual se codifica mediante predicción de movimiento entre vistas;
el bloque actual se intercodifica;
el segmento se intercodifica;
el bloque actual se codifica con modo de salto o modo directo;
una magnitud del vector de disparidad para el bloque actual es mayor que una magnitud del DDV para el segmento; o
el DDV para el segmento no es cero.

6. El procedimiento de la reivindicación 1, que comprende además:

perfeccionar, basándose en píxeles de profundidad en una componente de vista de profundidad, el vector de disparidad para el bloque actual; y
actualizar el DDV para el segmento a fin de especificar el vector de disparidad perfeccionado para el bloque actual.

7. Un procedimiento de codificación de datos de vídeo, el procedimiento comprendiendo:

almacenar (152) un vector de disparidad de un bloque particular de un segmento de una imagen actual de los datos de vídeo como un vector de disparidad derivado (DDV) para el segmento, en el que el segmento incluye múltiples bloques y sólo hay un DDV para el segmento;

determinar (154), realizando un proceso de derivación de Vector de Disparidad Basado en Bloques Vecinos (NBDV), un vector de disparidad para un bloque actual, en el que el bloque actual es uno de los bloques del segmento, en el que realizar el proceso de derivación de NBDV comprende establecer el vector de disparidad para el bloque actual como el DDV para el segmento cuando el proceso de derivación de NBDV no identifica ningún vector de movimiento de disparidad disponible a partir de bloques vecinos en el tiempo o bloques vecinos en el espacio; y
almacenar (156), como el DDV para el segmento, el vector de disparidad para el bloque actual.

8. Un dispositivo de codificación de vídeo, que comprende:

una memoria que almacena datos de vídeo; y
uno o más procesadores configurados para:

almacenar un vector de disparidad de un bloque particular de un segmento de una imagen actual de los datos de vídeo como un vector de disparidad derivado (DDV) para el segmento, en el que el segmento incluye múltiples bloques y sólo hay un DDV para el segmento;
determinar, realizando un proceso de derivación de Vector de Disparidad Basado en Bloques Vecinos (NBDV), un vector de disparidad para un bloque actual, en el que el bloque actual es uno de los bloques del segmento, en el que realizar el proceso de derivación de NBDV comprende establecer el vector de disparidad para el bloque actual como el DDV para el segmento cuando el proceso de derivación de NBDV no identifica ningún vector de movimiento de disparidad disponible a partir de bloques vecinos en el tiempo o bloques vecinos en el espacio; y
almacenar, como el DDV para el segmento, el vector de disparidad para el bloque actual.

9. El dispositivo de codificación de vídeo de la reivindicación 8, en el que el uno o más procesadores están configurados para codificar el bloque actual basándose al menos en parte en el vector de disparidad para el bloque actual.

10. El dispositivo de codificación de vídeo de la reivindicación 8, en el que el uno o más procesadores están configurados para decodificar el bloque actual basándose al menos en parte en el vector de disparidad para el bloque actual.

11. El dispositivo de codificación de vídeo de la reivindicación 8, en el que el uno o más procesadores están configurados para:

determinar si un bloque vecino en el tiempo tiene un vector de movimiento de disparidad;
como respuesta a la determinación de que el bloque vecino en el tiempo tiene un vector de movimiento de disparidad, derivar el vector de disparidad para el bloque actual basándose en el vector de movimiento de disparidad del bloque vecino en el tiempo;
como respuesta a la determinación de que el bloque vecino en el tiempo no tiene un vector de movimiento de disparidad, determinar si algún bloque vecino en el espacio de una pluralidad de bloques vecinos en el espacio tiene un vector de movimiento de disparidad; y
como respuesta a la determinación de que un bloque vecino en el espacio particular de la pluralidad de bloques vecinos en el espacio tiene un vector de movimiento de disparidad, derivar el vector de disparidad para el bloque actual basándose en el vector de movimiento de disparidad del bloque vecino en el espacio particular.

12. El dispositivo de codificación de vídeo de la reivindicación 8, en el que el uno o más procesadores están configurados para actualizar el DDV una vez para cada bloque del segmento que se codifica después del bloque actual.

13. El dispositivo de codificación de vídeo de la reivindicación 8, en el que el uno o más procesadores están configurados para actualizar el DDV para el segmento a fin de especificar el vector de disparidad para el bloque actual como respuesta al cumplimiento de una condición, siendo la condición una de las siguientes:

el bloque actual se codifica mediante predicción de movimiento entre vistas;
el bloque actual se intercodifica;
el segmento se intercodifica;
el bloque actual se codifica con modo de salto o modo directo;
una magnitud del vector de disparidad para el bloque actual es mayor que una magnitud del DDV para el segmento; o
el DDV para el segmento no es cero.

14. Un medio legible por ordenador no transitorio que tiene instrucciones almacenadas que, al ejecutarse, hacen que uno o más procesadores de un dispositivo de codificación de vídeo lleven a cabo el procedimiento de cualquiera de las reivindicaciones 1 a 7.

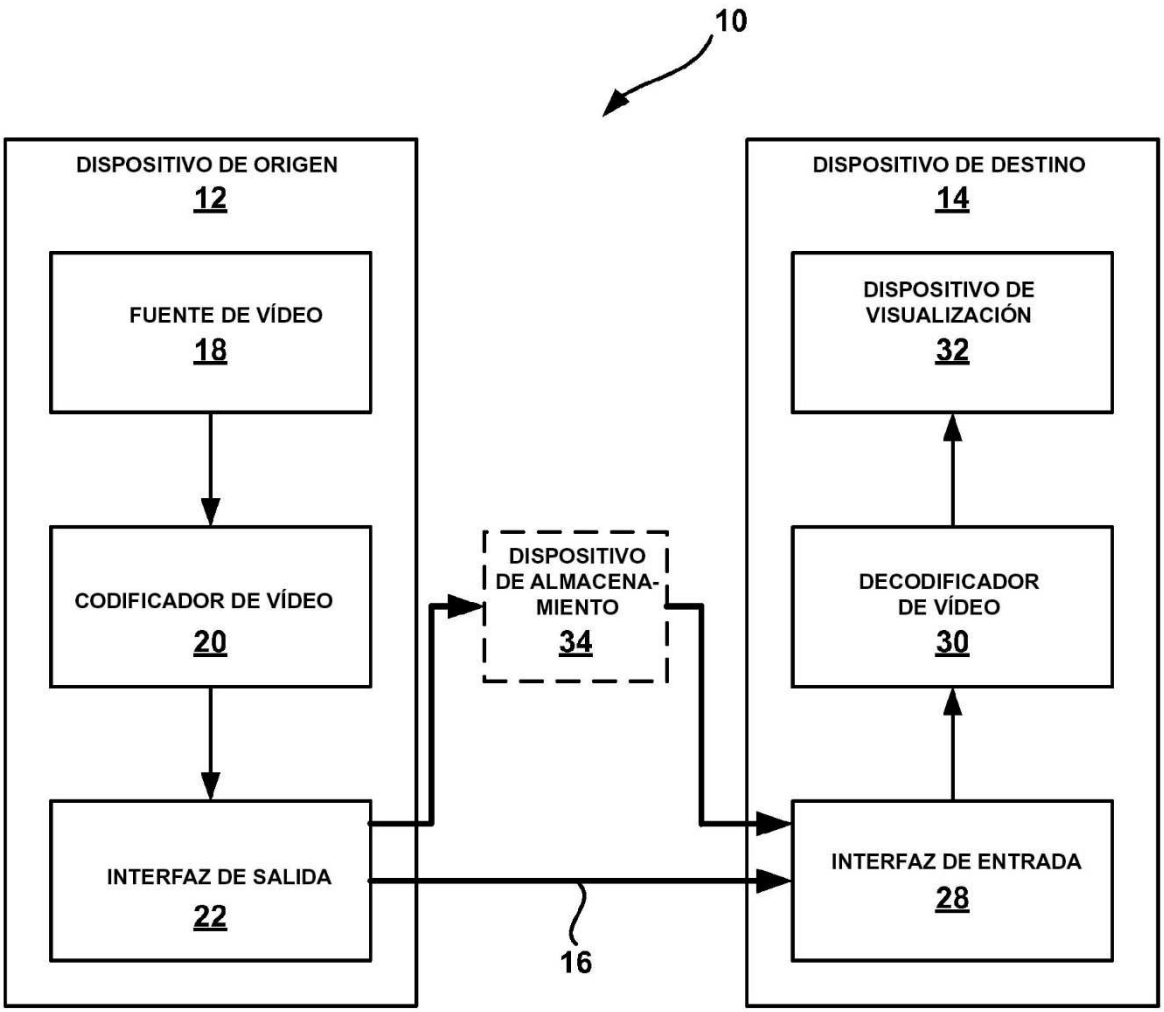
10
DISPOSITIVO DE ORIGEN
12
FUENTE DE VÍDEO
18
CODIFICADOR DE VÍDEO
20
INTERFAZ DE SALIDA
22
DISPOSITIVO DE ALMACENA-MIENTO
34
16
DISPOSITIVO DE DESTINO
14
DISPOSITIVO DE VISUALIZACIÓN
32
DECODIFICADOR DE VÍDEO
30
INTERFAZ DE ENTRADA
28

FIG. 1

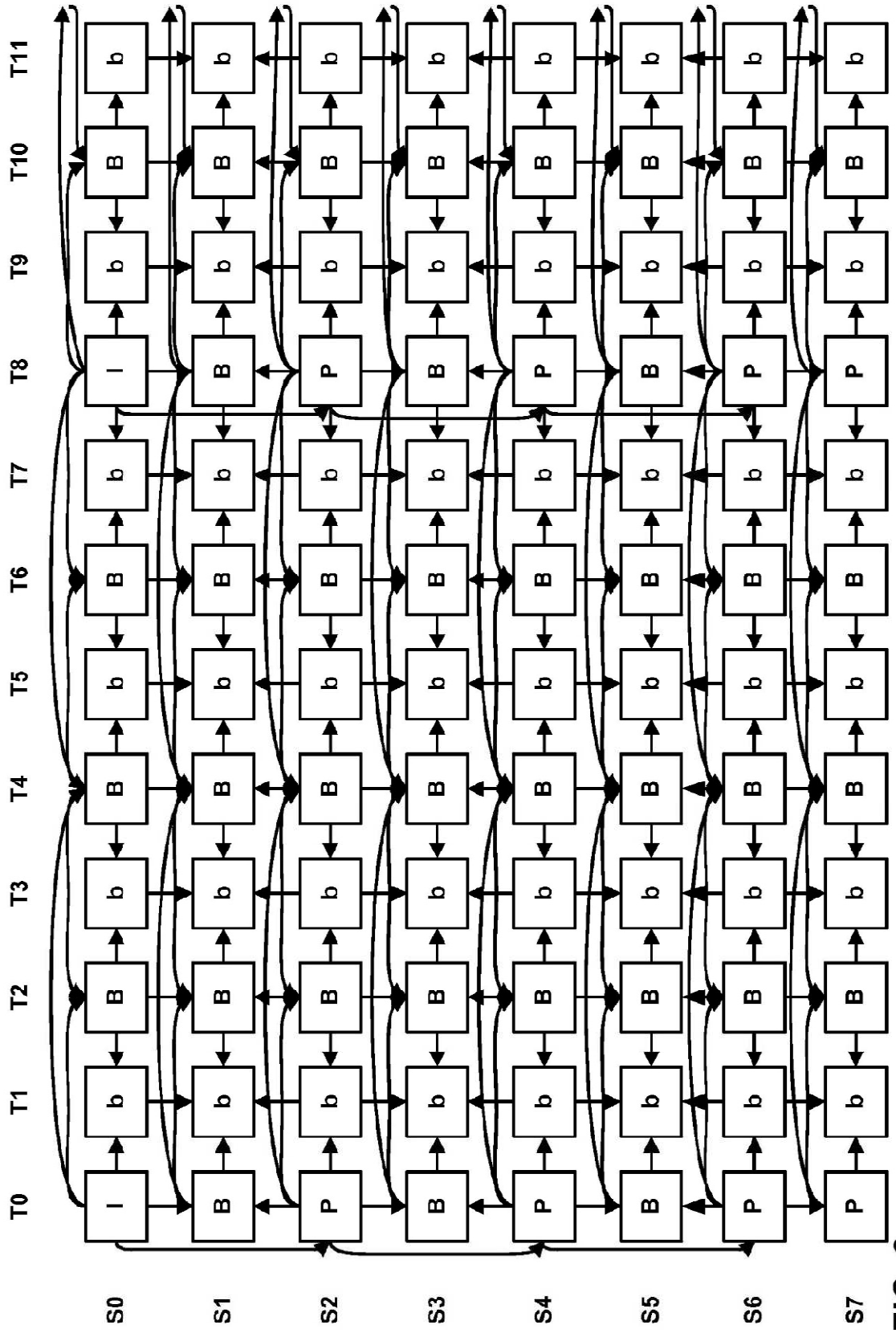

FIG. 2

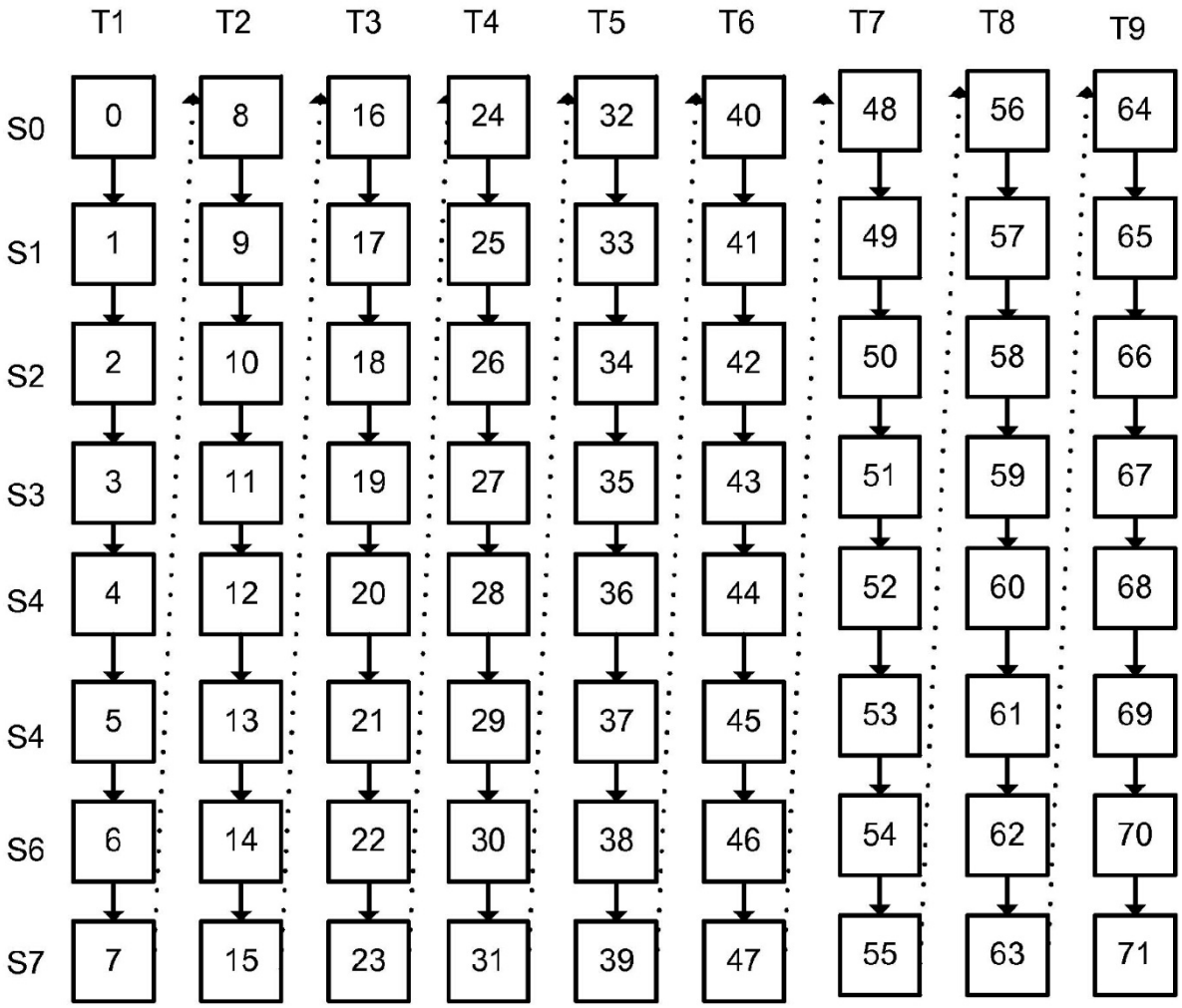
T1
T2
T3
T4
T5
T6
T7
T8
T9
S0
S1
S2
S3
S4
S4
S6
S7
0
8
16
24
32
40
48
56
64
1
9
17
25
33
41
49
57
65
2
10
18
26
34
42
50
58
66
3
11
19
27
35
43
51
59
67
4
12
20
28
36
44
52
60
68
5
13
21
29
37
45
53
61
69
6
14
22
30
38
46
54
62
70
7
15
23
31
39
47
55
63
71

FIG. 3

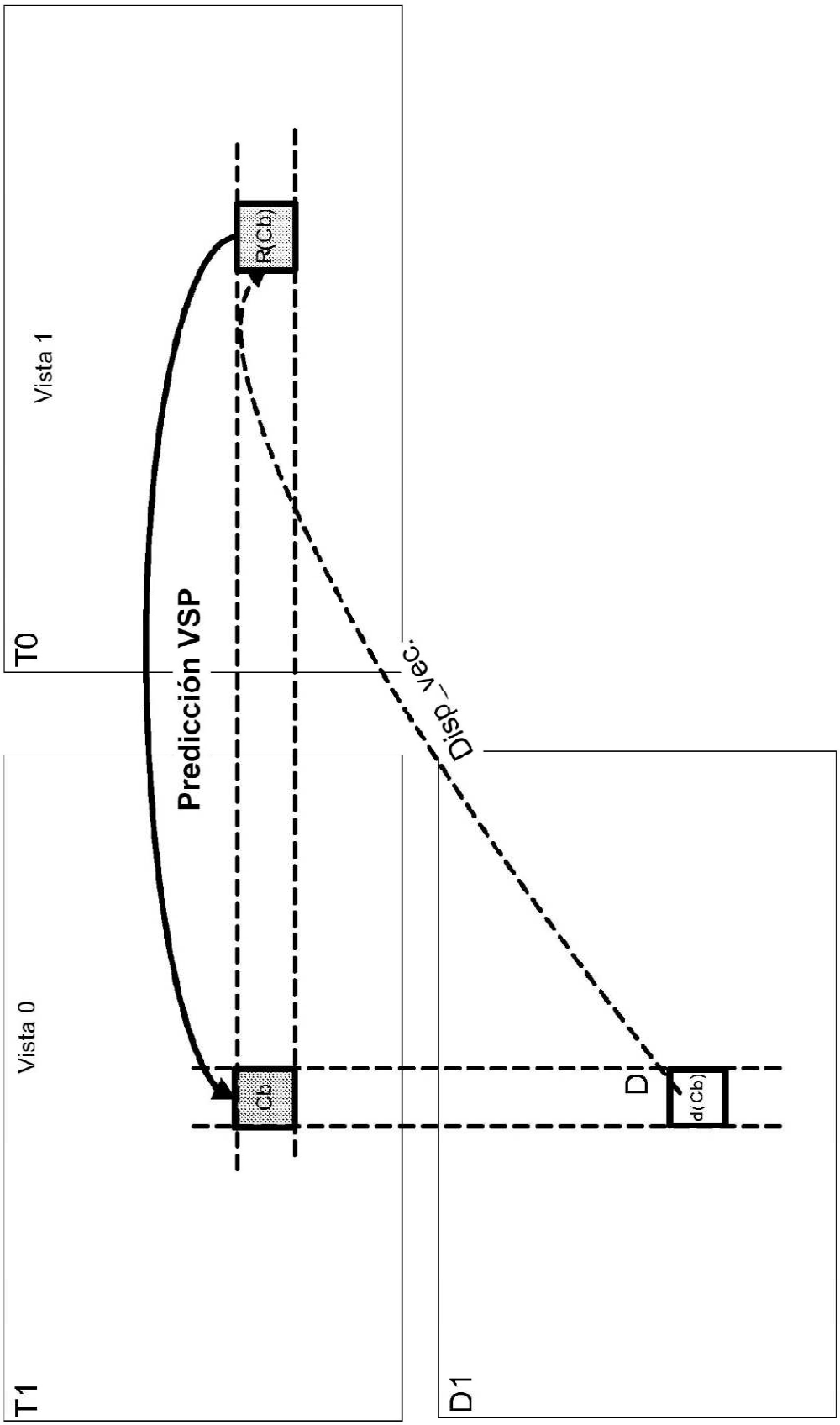

FIG. 4

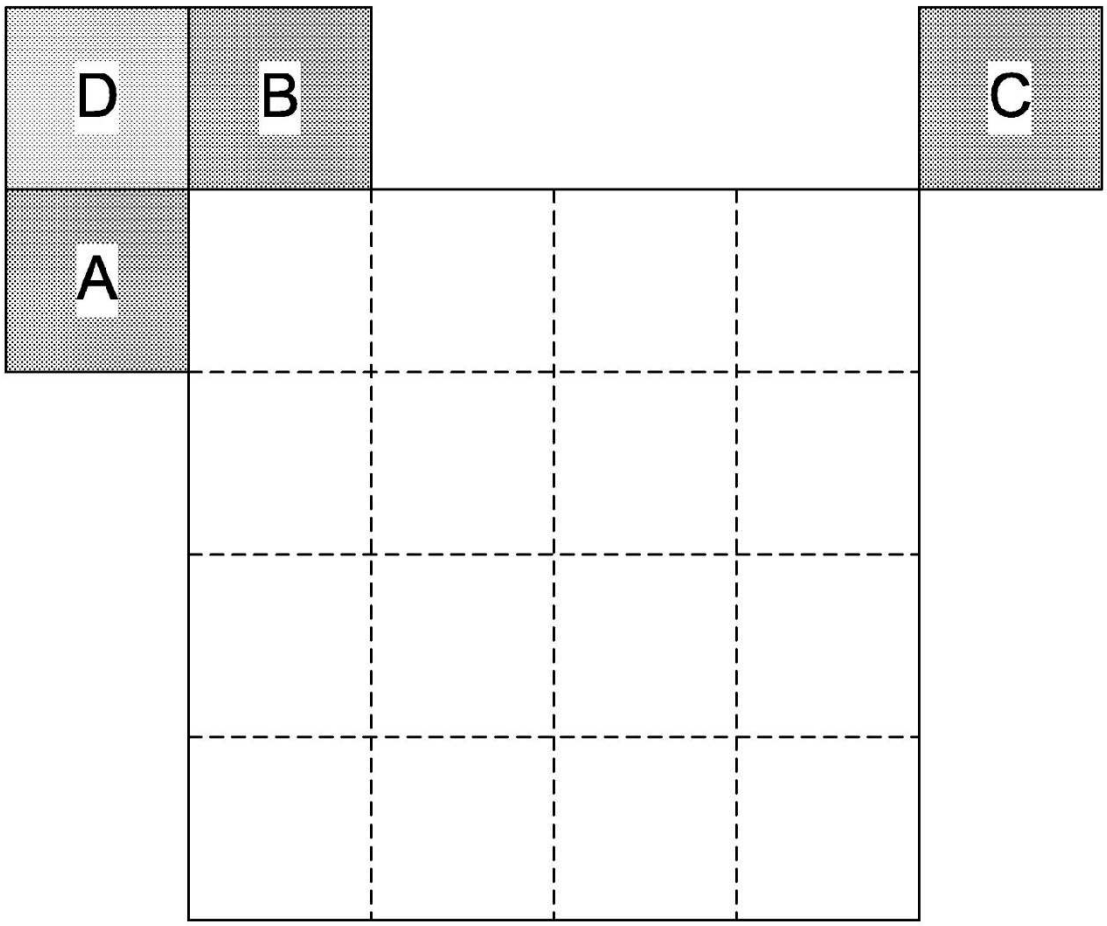

FIG. 5

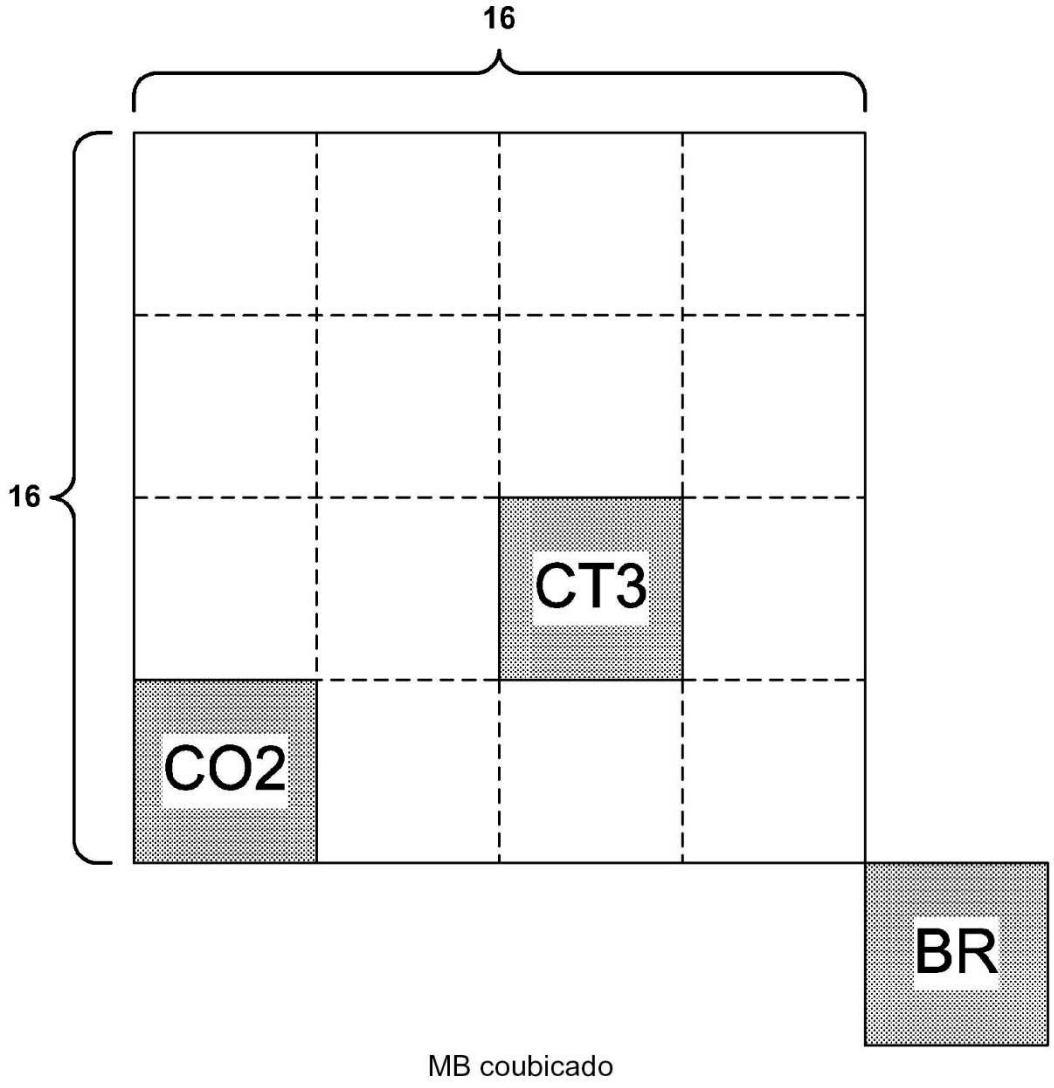

FIG. 6

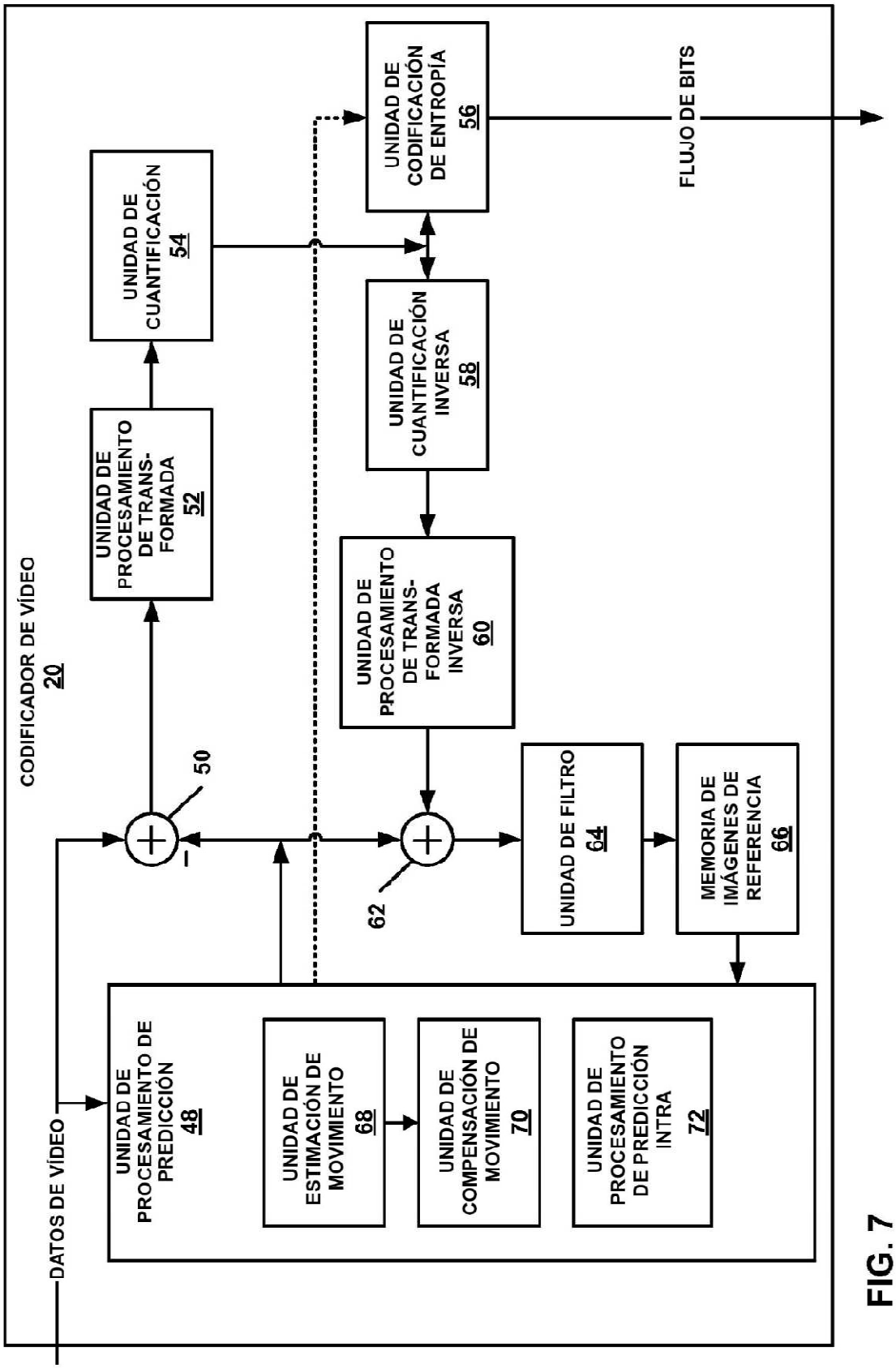
CODIFICADOR DE VÍDEO
20
DATOS DE VÍDEO
UNIDAD DE PROCESAMIENTO DE PREDICCIÓN
48
UNIDAD DE ESTIMACIÓN DE MOVIMIENTO
68
UNIDAD DE COMPENSACIÓN DE MOVIMIENTO
70
UNIDAD DE PROCESAMIENTO DE PREDICCIÓN INTRA
72
50
UNIDAD DE PROCESAMIENTO DE TRANS-FORMADA
52
UNIDAD DE CUANTIFICACIÓN
54
UNIDAD DE CODIFICACIÓN DE ENTROPÍA
56
FLUJO DE BITS
UNIDAD DE CUANTIFICACIÓN INVERSA
58
UNIDAD DE PROCESAMIENTO DE TRANS-FORMADA INVERSA
60
62
UNIDAD DE FILTRO
64
MEMORIA DE IMÁGENES DE REFERENCIA
66

FIG. 7

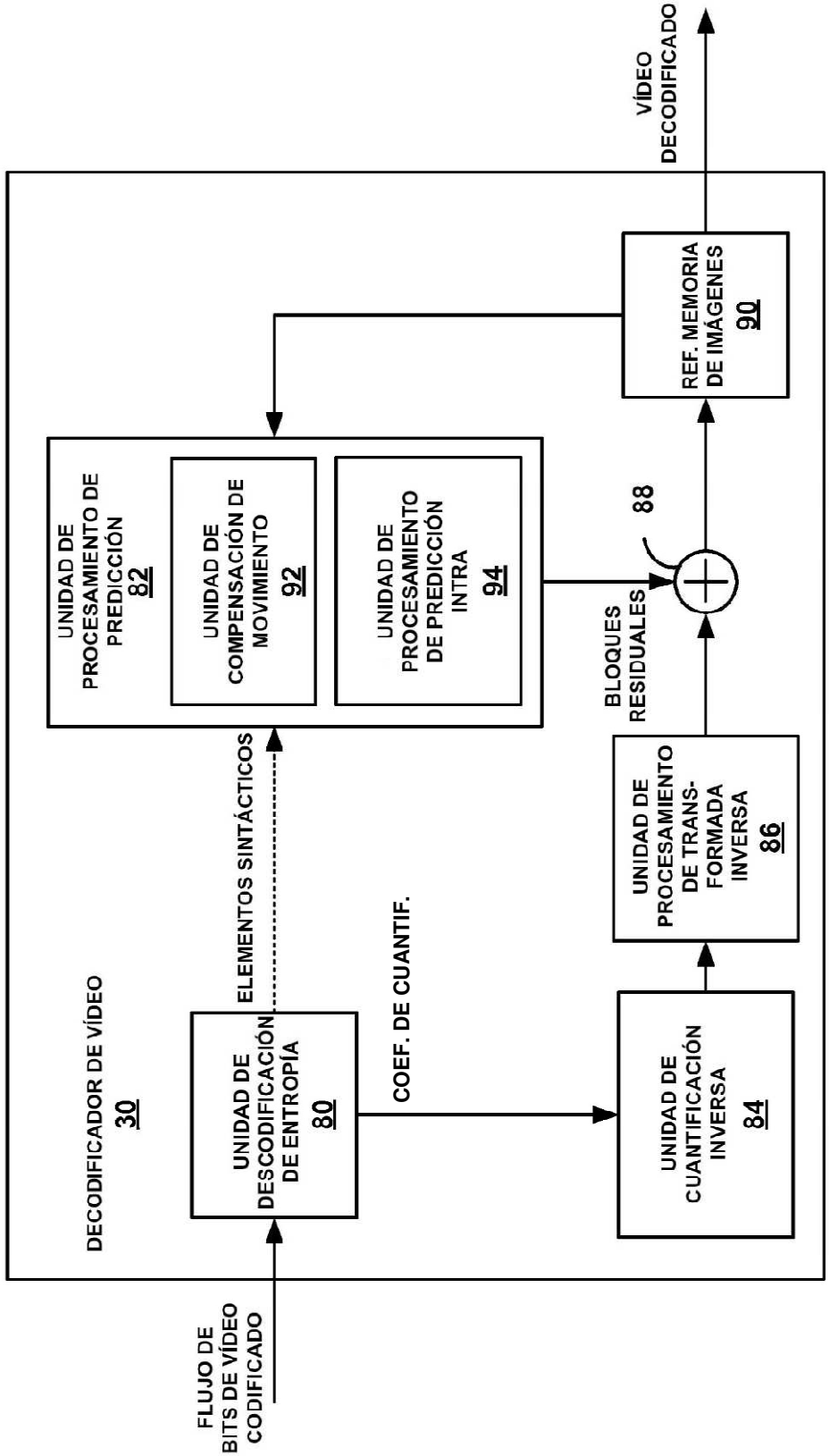
VÍDEO DECODIFICADO
REF. MEMORIA DE IMÁGENES 90
UNIDAD DE PROCESAMIENTO DE PREDICCIÓN 82
UNIDAD DE COMPENSACIÓN DE MOVIMIENTO 92
UNIDAD DE PROCESAMIENTO DE PREDICCIÓN INTRA 94
88
BLOQUES RESIDUALES
UNIDAD DE PROCESAMIENTO DE TRANS-FORMADA INVERSA 86
UNIDAD DE CUANTIFICACIÓN INVERSA 84
ELEMENTOS SINTÁCTICOS
COEF. DE CUANTIF.
UNIDAD DE DESCODIFICACIÓN DE ENTROPÍA 80
DECODIFICADOR DE VÍDEO 30
FLUJO DE BITS DE VÍDEO CODIFICADO

FIG. 8

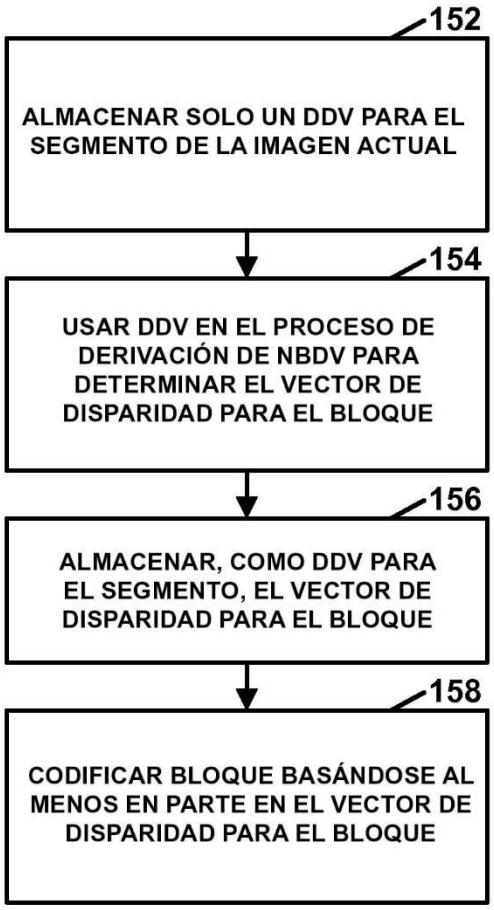
152
ALMACENAR SOLO UN DDV PARA EL SEGMENTO DE LA IMAGEN ACTUAL
154
USAR DDV EN EL PROCESO DE DERIVACIÓN DE NBDV PARA DETERMINAR EL VECTOR DE DISPARIDAD PARA EL BLOQUE
156
ALMACENAR, COMO DDV PARA EL SEGMENTO, EL VECTOR DE DISPARIDAD PARA EL BLOQUE
158
CODIFICAR BLOQUE BASÁNDOSE AL MENOS EN PARTE EN EL VECTOR DE DISPARIDAD PARA EL BLOQUE

FIG. 9A

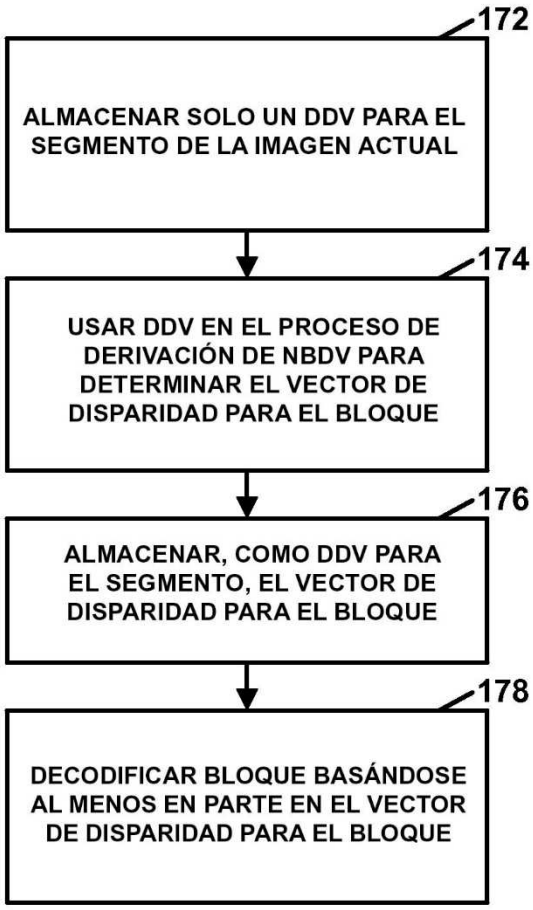
172
ALMACENAR SOLO UN DDV PARA EL SEGMENTO DE LA IMAGEN ACTUAL
174
USAR DDV EN EL PROCESO DE DERIVACIÓN DE NBDV PARA DETERMINAR EL VECTOR DE DISPARIDAD PARA EL BLOQUE
176
ALMACENAR, COMO DDV PARA EL SEGMENTO, EL VECTOR DE DISPARIDAD PARA EL BLOQUE
178
DECODIFICAR BLOQUE BASÁNDOSE AL MENOS EN PARTE EN EL VECTOR DE DISPARIDAD PARA EL BLOQUE

FIG. 9B

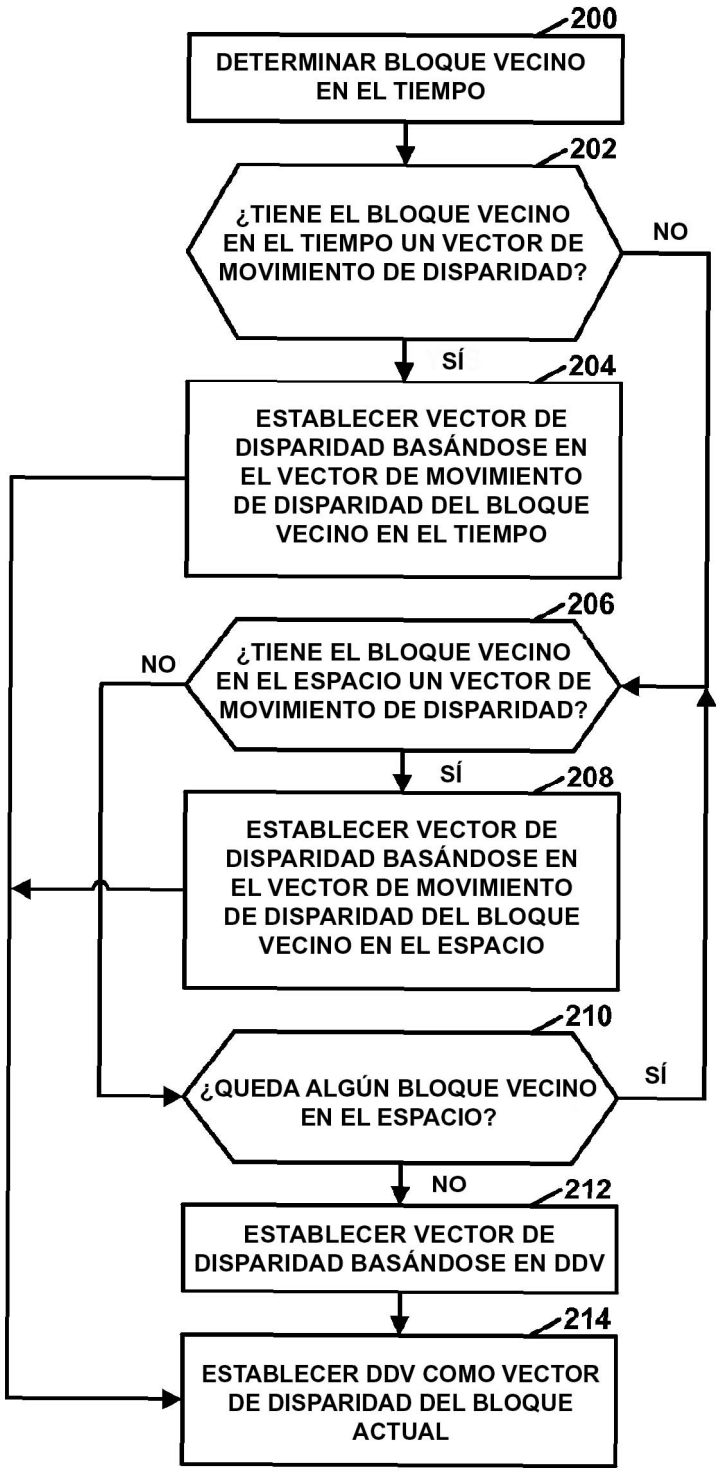
200
DETERMINAR BLOQUE VECINO EN EL TIEMPO
202
¿TIENE EL BLOQUE VECINO EN EL TIEMPO UN VECTOR DE MOVIMIENTO DE DISPARIDAD?
NO
SÍ
204
ESTABLECER VECTOR DE DISPARIDAD BASÁNDOSE EN EL VECTOR DE MOVIMIENTO DE DISPARIDAD DEL BLOQUE VECINO EN EL TIEMPO
206
¿TIENE EL BLOQUE VECINO EN EL ESPACIO UN VECTOR DE MOVIMIENTO DE DISPARIDAD?
NO
SÍ
208
ESTABLECER VECTOR DE DISPARIDAD BASÁNDOSE EN EL VECTOR DE MOVIMIENTO DE DISPARIDAD DEL BLOQUE VECINO EN EL ESPACIO
210
¿QUEDA ALGÚN BLOQUE VECINO EN EL ESPACIO?
SÍ
NO
212
ESTABLECER VECTOR DE DISPARIDAD BASÁNDOSE EN DDV
214
ESTABLECER DDV COMO VECTOR DE DISPARIDAD DEL BLOQUE ACTUAL

FIG. 10