(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 704 885**

(51) Int. CI.:

H04N 19/52 (2014.01)
H04N 19/44 (2014.01)
H04N 19/56 (2014.01)
H04N 19/53 (2014.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **26.02.2014 PCT/US2014/018716**

(87) Fecha y número de publicación internacional: **04.09.2014 WO14134181**

(96) Fecha de presentación y número de la solicitud europea: **26.02.2014 E 14711357 (5)**

(97) Fecha y número de publicación de la concesión europea: **10.10.2018 EP 2962466**

(54) Título: **Derivación del vector de disparidad de bloque vecino en codificación de vídeo 3D**

(30) Prioridad:

26.02.2013 US 201361769716 P
27.02.2013 US 201361770263 P
27.02.2013 US 201361770268 P
04.03.2013 US 201361772321 P
19.03.2013 US 201361803384 P
24.04.2013 US 201361815656 P
25.02.2014 US 201414189679

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
20.03.2019

(73) Titular/es:

QUALCOMM INCORPORATED (100.0%)
5775 Morehouse Drive
San Diego, CA 92121-1714, US

(72) Inventor/es:

ZHANG, LI;
CHEN, YING y
KANG, JEWON

(74) Agente/Representante:

FORTEA LAGUNA, Juan José

DESCRIPCIÓN

Derivación del vector de disparidad de bloque vecino en codificación de vídeo 3D

CAMPO TÉCNICO

[0001] Esta divulgación se refiere a la codificación y descodificación de vídeo.

ANTECEDENTES

[0002] Las capacidades de vídeo digital se pueden incorporar a una amplia gama de dispositivos, incluyendo televisores digitales, sistemas de radiodifusión 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 eficiencia (HEVC), actualmente en desarrollo, y las extensiones de dichas normas, para transmitir, recibir y almacenar información de vídeo digital de forma más eficaz.

[0003] Las ampliaciones de algunas de las normas mencionadas anteriormente, incluyendo la H.264/AVC, proporcionan técnicas para la codificación de vídeo multivista a fin de producir vídeo estéreo o tridimensional ("3D"). En particular, se han propuesto técnicas para la codificación de multivista para su uso en AVC, con la norma de codificación de vídeo ajustable a escala (SVC) (que es la ampliación ajustable a escala a H.264/AVC), y la norma de codificación de vídeo multivista (MVC) (que se ha convertido en la ampliación multivista a H.264/AVC).

[0004] Típicamente, el vídeo estéreo se logra usando dos vistas, por ejemplo, una vista izquierda y una vista derecha. Una imagen de la vista izquierda se puede visualizar de forma sustancialmente simultánea con una imagen de la vista derecha para lograr un efecto de vídeo tridimensional. Por ejemplo, un usuario puede usar gafas pasivas polarizadas que filtran la vista izquierda de la vista derecha. De forma alternativa, 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.

[0005] En "CE5.h related: Relacionado con CE5.h:Depth-oriented Neighboring Block Disparity Vector (DoNBDV) with virtual depth retrieval [Vector de disparidad de bloques vecino orientado en profundidad (DoNBDV) con recuperación de profundidad virtual]" de Yu-Lin Chang, Chi-Ling Wu, Yu-Pao Tsai, Shawmin Lei (Equipo de Colaboración Conjunta en el Desarrollo de Ampliación de Codificación de Vídeo 3D de ITU-T SG 16 WP 3 e ISO/CEI JTC 1/SC 29/WG 11 2.ª conferencia: Shanghai, CN, 13-19 de octubre de 2012, número de documento JCT3V-B0090), se propone un nuevo vector de disparidad estimado, vector de disparidad de bloque vecino orientado a la profundidad (DoNBDV) para potenciar la exactitud del NBDV utilizando el mapa de profundidad codificado.

[0006] En "CE1.h: Backward View Synthesis Prediction using Neighbouring Blocks [CE1.h: Predicción de síntesis de vista inversa usando bloques vecinos]" de Dong Tian *et al*. (Equipo de Colaboración Conjunta en el Desarrollo de Ampliación de Codificación de Vídeo 3D de ITU-T SG 16 WP 3 e ISO/IEC JTC 1/SC 29/WG 11 3.ª conferencia: Ginebra, CH, 16-23 de enero de 2013, documento número JCT3V-C0152), se propone un enfoque de predicción de síntesis de vista de distorsión inversa usando bloques vecinos para derivar un bloque de profundidad para realizar la operación de distorsión inversa, con un nuevo candidato de fusión que indica el modo BVSP añadido en la lista de candidatos de fusión.

[0007] En "3D-CE5.a related: Simplification on the disparity vector derivation [Relacionado con 3D-CE5.a: Simplificación de la derivación del vector de disparidad]" de Gun Bang *et al*. (Equipo de Colaboración Conjunta en el Desarrollo de Ampliación de Codificación de Vídeo 3D de ITU-T SG 16 WP 3 e ISO/IEC JTC 1/SC 29/WG 11 2.ª conferencia: Shanghai, CN, 13-19 de octubre de 2012, documento n.º JCT3V-B0073), se propone un procedimiento de derivación de vector de disparidad simplificado para guardar el ancho de banda de acceso a la memoria. Se afirma que la implementación del proceso propuesto se puede lograr compartiendo una muestra de profundidad para derivar el vector de disparidad en todos los subbloques dentro de cada MB.

[0008] En "3D-CE5.h related: Constrained DV for inter-view data access [Relacionado con 3D-CE5.h: DV restringido para el acceso de datos entre vistas]" de Yi-Wen Chen *et al*. (Equipo de Colaboración Conjunta en el Desarrollo de Ampliación de Codificación de Vídeo 3D de ITU-T SG 16 WP 3 e ISO/IEC JTC 1/SC 29/WG 11 2.ª conferencia: Shanghai, CN, 13-19 de octubre de 2012, documento n.º JCT3V-B0087), se afirma que, en la codificación de vídeo 3D basada en HEVC, HTM 4.0.1, se usa un vector de disparidad (DV) derivado para ubicar el bloque correspondiente en la imagen de la vista vecina para la predicción de movimiento entre vistas y la predicción residual entre vistas y, teniendo en cuenta que los datos de entrada se rectifican para evitar la desalineación de la geometría de la cámara, el componente vertical del DV usado para la predicción residual entre vistas se restringe a cero, pero la componente vertical del DV usado para la predicción del parámetro de movimiento entre vistas no se restringe a cero. Yi-Wen Chen

et al. proponen restringir la componente vertical del DV derivado para el acceso de datos entre vistas a cero para la unificación y simplificación.

[0009] En "MV-HEVC: Vertical length restriction of inter-view vector for HEVC simple 3D extension [MV-HEVC: Restricción de la longitud vertical del vector entre vistas para la ampliación 3D simple de HEVC]" de Ohji Nakagami *et al*. (Equipo de Colaboración Conjunta en el Desarrollo de Ampliación de Codificación de Vídeo 3D de ITU-T SG 16 WP 3 e ISO/IEC JTC 1/SC 29/WG 11 2.ª conferencia: Shanghai, CN, 13-19 de octubre de 2012, documento n.º JCT3V-B0037_r1), se propone una restricción de longitud vertical del vector entre vistas a fin de reducir la complejidad del decodificador. Ohji Nakagami *et al*. creen que hace que la implementación del decodificador sea más fácil en relación con el proceso de compensación de movimiento sin ninguna pérdida de rendimiento de codificación significativa porque la mayoría de las secuencias 3D se capturan por cámaras colocadas horizontalmente y se rectifican antes de la entrada del codificador.

SUMARIO

[0010] En general, esta divulgación describe técnicas para codificación de vídeo 3D. En particular, esta divulgación se refiere a la derivación del vector de disparidad de bloque vecino (NBDV) y la predicción de síntesis de vistas basada en bloques (BVSP) en la codificación de vídeo 3D.

[0011] La invención se define por las reivindicaciones independientes que se adjuntan.

[0012] Los detalles de uno o más ejemplos se exponen en los dibujos adjuntos y en la descripción siguiente.

BREVE DESCRIPCIÓN DE LOS DIBUJOS

[0013]

La FIG. 1 es un diagrama de bloques que ilustra un sistema de codificación y descodificación de vídeo de ejemplo que puede utilizar las técnicas descritas en esta divulgación.

La FIG. 2 es un diagrama conceptual que ilustra un orden de descodificación de multivistas de ejemplo.

La FIG. 3 es un diagrama conceptual que ilustra una estructura de predicción de ejemplo para la codificación de multivistas.

La FIG. 4 es una visualización conceptual de la predicción de la síntesis de vista basada en bloques en base a la distorsión inversa.

La FIG. 5 es un diagrama conceptual que muestra bloques vecinos espaciales usados para la derivación del vector de disparidad de bloque vecino.

La FIG. 6 es un diagrama de bloques que ilustra un codificador de vídeo de ejemplo que puede implementar las técnicas descritas en esta divulgación.

La FIG. 7 es un diagrama de bloques que ilustra un decodificador de vídeo de ejemplo que puede implementar las técnicas descritas en esta divulgación.

La FIG. 8 es un diagrama conceptual que muestra bloques vecinos para la predicción de vectores de movimiento basada en profundidad.

La FIG. 9 es un diagrama de flujo que ilustra un procedimiento de ejemplo de la divulgación.

La FIG. 10 es un diagrama de flujo que ilustra otro procedimiento de ejemplo de la divulgación.

DESCRIPCIÓN DETALLADA

[0014] En general, esta divulgación describe técnicas para codificación de vídeo multivista más profundidad (por ejemplo, 3D) en base a códecs avanzados, incluyendo la codificación de dos o más vistas con el códec H.264/de codificación de vídeo avanzado (AVC) (por ejemplo, en una ampliación 3D de H.264/AVC a veces denominada 3D-AVC). En algunos ejemplos, se proponen técnicas relacionadas con la predicción de síntesis de vista y la derivación del vector de disparidad en la codificación de vídeo multivista basada en AVC 3D. Sin embargo, las técnicas de esta divulgación pueden ser genéricamente aplicables a otras técnicas de codificación de vídeo multivista y/o 3D, incluyendo las ampliaciones multivista y 3D emergentes de la norma de codificación de vídeo de alta eficacia (HEVC).

[0015] Cuando se emplea la descodificación de textura primero, las propuestas actuales para 3D-AVC carecen de técnicas para derivar vectores de disparidad exactos. En particular, los vectores de disparidad derivados de las

propuestas actuales para un proceso de derivación del vector de disparidad de bloque vecino (NBDV) pueden producir vectores de disparidad inexactos. Además, no existen técnicas actuales para usar vectores de disparidad derivados para la codificación de predicción de síntesis de vistas basada en bloques (BVSP) cuando se emplea la codificación de textura primero.

[0016] En vista de estos inconvenientes, esta divulgación propone técnicas para posibilitar BVSP para codificadores de vídeo y decodificadores de vídeo compatibles con 3D-AVC cuando se codifica el componente de vista de textura no básica antes del componente de vista de profundidad no básica correspondiente. Además, la ganancia de codificación de otros modos de intercodificación también se mejora debido a la derivación de un vector de disparidad elaborado, como se proporciona por las técnicas de esta divulgación.

[0017] La FIG. 1 es un diagrama de bloques que ilustra un sistema de codificación y descodificación de vídeo 10 de ejemplo que se puede configurar para realizar las técnicas de predicción de síntesis de vista y derivación del vector de disparidad descritas en esta divulgación. Como se muestra en la FIG. 1, el sistema 10 incluye un dispositivo de origen 12 que genera datos de vídeo codificados que un dispositivo de destino 14 va a descodificar en un momento posterior. El dispositivo de origen 12 y el dispositivo de destino 14 pueden comprender cualquiera entre una amplia gama de dispositivos, incluyendo ordenadores de sobremesa, ordenadores plegables (es decir, portátiles), ordenadores de tipo tableta, decodificadores, equipos telefónicos tales como los denominados teléfonos "inteligentes", los denominados paneles "inteligentes", televisores, cámaras, dispositivos de visualización, reproductores de medios digitales, consolas de videojuegos, dispositivos de transmisión continua 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.

[0018] El sistema 10 puede funcionar de acuerdo con diferentes normas de codificación de vídeo, una norma patentada o cualquier otra forma de codificación multivista. A continuación se describen algunos ejemplos de normas de codificación de vídeo, y no se deben considerar limitantes. Entre las normas de codificación de vídeo se incluyen ITU-T H.261, Visual del MPEG-1 de la ISO/IEC, ITU-T H.262 o Visual del MPEG-2 de la ISO/IEC, ITU-T H.263, Visual del MPEG-4 de la ISO/IEC e ITU-T H.264 (también conocida como AVC del MPEG-4 de la ISO/IEC), incluyendo sus ampliaciones de codificación de vídeo ajustable a escala (SVC) y de codificación de vídeo multivista (MVC). El último borrador conjunto de MVC se describe en "Advanced video coding for generic audiovisual services [Codificación de vídeo avanzada para servicios audiovisuales genéricos]", Recomendación ITU-T H.264, marzo de 2010. Otro borrador conjunto del MVC se describe en "Advanced video coding for generic audiovisual services [Codificación de vídeo avanzada para servicios audiovisuales genéricos]", Recomendación ITU-T H.264, junio de 2011. Algunas normas de codificación de vídeo adicionales incluyen MVC+D y 3D-AVC, que se basan en AVC. Además, se ha desarrollado una nueva norma de codificación de vídeo, concretamente, la Codificación de Vídeo de Alta Eficacia (HEVC), por el Equipo de Colaboración Conjunta en Codificación de Vídeo (JCT-VC) del Grupo de Expertos en Codificación de Vídeo (VCEG) de la ITU-T y el Grupo de Expertos en Imagen en Movimiento (MPEG) de ISO/IEC.

[0019] Para los propósitos de ilustración solamente, las técnicas descritas en esta divulgación se describen con ejemplos de acuerdo a la norma H.264, tales como la 3D-AVC. Sin embargo, las técnicas descritas en esta divulgación no se deben considerar limitadas a estas normas de ejemplo, y se pueden ampliar a otras normas de codificación de vídeo para codificación de multivista o codificación de vídeo 3D (por ejemplo, 3D-HEVC), o a técnicas relacionadas con la codificación de multivista o codificación de vídeo 3D que no se basan necesariamente en una norma de codificación de vídeo particular. Por ejemplo, las técnicas descritas en esta divulgación se implementan mediante codificadores/decodificadores de vídeo (códecs) para la codificación de multivista, donde la codificación de multivista incluye la codificación de dos o más vistas.

[0020] El dispositivo de destino 14 puede recibir los datos de vídeo codificados que se van a decodificar, por medio de un enlace 16. El enlace 16 puede comprender cualquier tipo de medio o dispositivo capaz de desplazar los datos de vídeo codificados desde el dispositivo de origen 12 hasta el dispositivo de destino 14. En un ejemplo, el enlace 16 puede comprender un medio de comunicación para posibilitar al dispositivo de origen 12 transmitir datos de vídeo codificados directamente al dispositivo de destino 14 en tiempo real. Los datos de vídeo codificados se pueden modular de acuerdo con una norma de comunicación, tal como un protocolo de comunicación inalámbrica, y transmitir al dispositivo de destino 14. El medio de comunicación puede comprender cualquier medio de comunicación, inalámbrica o alámbrica, 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 amplia o una red global tal como Internet. El medio de comunicación puede incluir enrutadores, conmutadores, estaciones base o cualquier otro equipo que pueda ser útil para facilitar la comunicación desde el dispositivo de origen 12 hasta el dispositivo de destino 14.

[0021] De forma alternativa, los datos codificados se pueden transmitir desde la interfaz de salida 22 a un dispositivo de almacenamiento 34. De forma similar, una interfaz de entrada puede acceder a los datos codificados desde el dispositivo de almacenamiento 34. El dispositivo de almacenamiento 34 puede incluir cualquiera de una variedad de medios de almacenamiento de datos de acceso distribuido o local, tales como una unidad de disco duro, unos discos Blu-ray, unos DVD, unos CD-ROM, una memoria flash, una memoria volátil o no volátil o cualquier otro medio de almacenamiento digital adecuado para almacenar datos de vídeo codificados. En otro ejemplo, el dispositivo de almacenamiento 34 puede corresponder a un servidor de archivos o a otro dispositivo de almacenamiento intermedio

que puede contener 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 por medio de transmisión en continuo o descarga. El servidor de archivos puede ser cualquier tipo de servidor capaz de almacenar datos de vídeo codificados y transmitir esos datos de vídeo codificados al dispositivo de destino 14. Los servidores de archivos de ejemplo incluyen un servidor web (por ejemplo, para un sitio web), un servidor FTP, dispositivos de almacenamiento conectado en red (NAS) o una unidad de disco local. El dispositivo de destino 14 puede acceder a los datos de vídeo codificados a través de cualquier conexión de datos estándar, incluyendo una conexión a Internet. Esto puede incluir un canal inalámbrico (por ejemplo, una conexión 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 codificados almacenados en un servidor de archivos. La transmisión de datos de vídeo codificados desde el dispositivo de almacenamiento 34 puede ser una transmisión en continuo, una transmisión de descarga o una combinación de ambas.

[0022] Las técnicas de esta divulgación para la predicción de síntesis de vistas y la derivación del vector de disparidad no se limitan necesariamente a aplicaciones o configuraciones inalámbricas. Las técnicas se pueden aplicar a la codificación de vídeo como soporte a cualquiera de una variedad de aplicaciones de multimedia, tales como radiodifusiones de televisión por 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, descodificación de vídeo digital almacenado en un medio de almacenamiento de datos, u otras aplicaciones. En algunos ejemplos, el sistema 10 se puede configurar para admitir transmisión de vídeo unidireccional o bidireccional para apoyar aplicaciones tales como la transmisión en continuo de vídeo, la reproducción de vídeo, la radiodifusión de vídeo y/o la videotelefonía.

[0023] 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 casos, la interfaz de salida 22 puede incluir un modulador/desmodulador (módem) y/o un transmisor. En el dispositivo de origen 12, la fuente de vídeo 18 puede incluir una fuente tal como un dispositivo de captación de vídeo, por ejemplo, una videocámara, un archivo de vídeo que contiene vídeo previamente captado, una interfaz de vídeo en tiempo real 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 el 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 se pueden aplicar a aplicaciones inalámbricas y/o alámbricas.

[0024] El vídeo capturado, precapturado o generado por ordenador se puede codificar por el codificador de vídeo 20. Los datos de vídeo codificados se pueden transmitir directamente al dispositivo de destino 14 por medio de la interfaz de salida 22 del dispositivo de origen 12. Los datos de vídeo codificados se pueden almacenar también (o de forma alternativa) en el dispositivo de almacenamiento 34 para un posterior acceso por el dispositivo de destino 14 u otros dispositivos, para su descodificación y/o reproducción.

[0025] 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 recibe los datos de vídeo codificados por el enlace 16. Los datos de vídeo codificados comunicados por el enlace 16, o proporcionados en el dispositivo de almacenamiento 34, pueden incluir una variedad 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 descodificación de los datos de vídeo. Dichos elementos sintácticos se pueden incluir con los datos de vídeo codificados, transmitidos en un medio de comunicación, almacenados en un medio de almacenamiento o almacenados en un servidor de archivos.

[0026] El dispositivo de visualización 32 se puede integrar 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 descodificados ante un usuario y puede comprender cualquiera de 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.

[0027] Aunque no se muestra en la FIG. 1, en algunos aspectos, el codificador de vídeo 20 y el decodificador de vídeo 30 se pueden integrar, cada uno, en un codificador y decodificador de audio, y pueden incluir unidades MUX-DEMUX adecuadas, u otro hardware y software, para ocuparse de la codificación tanto de audio como de vídeo en un flujo de datos común o en flujos de datos separados. Si procede, en algunos ejemplos, las unidades MUX-DEMUX pueden ser conformes al protocolo de multiplexador 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 se pueden implementar, cada uno, como cualquiera entre una variedad de circuitos codificadores adecuados, tales como uno o más microprocesadores, procesadores de señales digitales (DSP), circuitos integrados específicos de la aplicación (ASIC), formaciones de puertas programables

in situ (FPGA), lógica discreta, software, hardware, firmware o cualquier combinación de los mismos. Por ejemplo, las técnicas descritas en esta divulgación se pueden describir desde la perspectiva de un aparato o un dispositivo. Como ejemplo, el aparato o dispositivo puede incluir el decodificador de vídeo 30 (por ejemplo, el dispositivo de destino 14 como parte de un dispositivo de comunicación inalámbrica), y el decodificador de vídeo 30 puede incluir uno o más procesadores configurados para implementar las técnicas descritas en esta divulgación (por ejemplo, descodificar datos de vídeo de acuerdo con las técnicas descritas en esta divulgación). Como otro ejemplo, el aparato o dispositivo puede incluir un microprocesador o un circuito integrado (IC) que incluye el decodificador de vídeo 30, y el microprocesador o IC puede ser parte del dispositivo de destino 14 u otro tipo de dispositivo. Lo mismo se puede aplicar al codificador de vídeo 20 (es decir, un aparato o dispositivo como el dispositivo de origen 12 y/o un microcontrolador o IC incluye el codificador de vídeo 20, donde el codificador de vídeo 20 se configura para codificar datos de vídeo de acuerdo con las técnicas descritas en esta divulgación).

[0029] Cuando las técnicas se implementan parcialmente en software, un dispositivo puede almacenar instrucciones para el software en un medio adecuado no transitorio legible por ordenador, y ejecutar las instrucciones en hardware usando uno o más procesadores para realizar las técnicas de esta divulgación. Cada uno de entre el codificador de vídeo 20 y el decodificador de vídeo 30 se puede incluir en uno o más codificadores o decodificadores, cualquiera de los cuales se puede integrar como parte de un codificador/decodificador (CÓDEC) combinado en un dispositivo respectivo.

[0030] Una secuencia de vídeo incluye típicamente una serie de imágenes de vídeo de una vista. Un grupo de imágenes (GOP) comprende en general una serie de una o más imágenes de vídeo. Un GOP puede incluir datos sintácticos en una cabecera del GOP, una cabecera de una o más imágenes del GOP, o en otras ubicaciones, que describen un número de imágenes incluidas en el GOP. Cada imagen puede incluir datos sintácticos de imagen que describen un modo de codificación para la imagen respectiva. Un codificador de vídeo 20 funciona típicamente en bloques de vídeo dentro de imágenes de vídeo individuales a fin de codificar los datos de vídeo. Un bloque de vídeo puede corresponder a un macrobloque, una división de un macrobloque y, posiblemente, un subbloque de una división, como se define en la norma H.264, o una unidad de codificación (CU), una unidad de predicción (PU) o una unidad de transformación (TU), como se define en la norma HEVC; sin embargo, las técnicas descritas en esta divulgación no se limitan a estos ejemplos de bloques. Los bloques de vídeo pueden tener tamaños fijos o variables y pueden diferir en tamaño de acuerdo con una norma de codificación especificada. Cada imagen de vídeo puede incluir una pluralidad de segmentos. Cada segmento puede incluir una pluralidad de bloques.

[0031] Como un ejemplo, la norma ITU-T H.264 admite la intrapredicción en diversos tamaños de bloque, tales como 16 por 16, 8 por 8 o 4 por 4 para componentes de luma, y 8x8 para componentes de croma, así como la interpredicción en diversos tamaños de bloque, tales como 16x16, 16x8, 8x16, 8x8, 8x4, 4x8 y 4x4 para componentes de luma y tamaños escalados correspondientes para componentes de croma. En esta divulgación, "NxN" y "N por N" se pueden usar de manera intercambiable para referirse a las dimensiones de píxel del bloque en lo que respecta a las dimensiones vertical y horizontal, (por ejemplo, 16x16 píxeles o 16 por 16 píxeles). En general, un bloque de 16x16 tendrá 16 píxeles en una dirección vertical (y = 16) y 16 píxeles en una dirección horizontal (x = 16). Asimismo, un bloque de NxN 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 se pueden disponer en filas y columnas. Además, no es necesario que los bloques tengan necesariamente el mismo número de píxeles en la dirección horizontal y en la dirección vertical. Por ejemplo, los bloques pueden comprender NxM píxeles, donde M no es necesariamente igual a N.

[0032] Cuando el bloque se codifica en modo intra (por ejemplo, intrapredicción), el bloque puede incluir datos que describen un modo de intrapredicción para el bloque. Como otro ejemplo, cuando el bloque se codifica en modo inter (por ejemplo, interpredicción), el bloque puede incluir información que define un vector de movimiento para el bloque. Este vector de movimiento se refiere a una imagen de referencia en la misma vista (por ejemplo, un vector de movimiento temporal), o se refiere a una imagen de referencia en otra vista (por ejemplo, un vector de movimiento de disparidad). Los datos que definen el vector de movimiento para un bloque describen, 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, una precisión de un cuarto de píxel o una precisión de un octavo de píxel). Además, cuando se interpredice, el bloque puede incluir información del índice de referencia, tal como una imagen de referencia a la que apunta el vector de movimiento, y/o una lista de imágenes de referencia (por ejemplo, RefPicList0 o RefPicList1) para el vector de movimiento.

[0033] En la norma H.264, tras la codificación intrapredictiva o interpredictiva, el codificador de vídeo 20 calcula los datos residuales para los macrobloques. Los datos residuales pueden corresponder a diferencias de píxeles entre píxeles de la imagen no codificada y a valores de predicción para el macrobloque en H.264.

[0034] Tras cualquier transformación para producir coeficientes de transformación, el codificador de vídeo 20 realiza la cuantificación de los coeficientes de transformación, en algunos ejemplos. La 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, proporcionando compresión adicional. El proceso de cuantificación reduce

la profundidad de bits asociada a algunos o a la totalidad de los coeficientes. Por ejemplo, un valor de n bits se redondea hacia abajo a un valor de m bits durante la cuantificación, donde n es mayor que m.

[0035] En algunos ejemplos, el codificador de vídeo 20 utiliza un orden de escaneado predefinido para escanear los coeficientes de transformada cuantificados para producir un vector en serie que se pueda codificar por entropía. En otros ejemplos, el codificador de vídeo 20 realiza un escaneado adaptable. Después de escanear los coeficientes de transformada cuantificados para formar un vector unidimensional, en algunos ejemplos, la entropía del codificador de vídeo 20 codifica el vector unidimensional de acuerdo con la codificación de longitud variable adaptable al contexto (CAVLC), la codificación aritmética binaria adaptable al contexto (CABAC), la codificación aritmética binaria adaptable al contexto basada en la sintaxis (SBAC), la codificación por entropía por división de intervalos de probabilidad (PIPE) u otra metodología de codificación por entropía, como algunos ejemplos. El codificador de vídeo 20 también codifica por entropía elementos sintácticos asociados a los datos de vídeo codificados, para su uso por el decodificador de vídeo 30 en la descodificación de los datos de vídeo.

[0036] Para realizar la CABAC, el codificador de vídeo 20 puede asignar un contexto dentro de un modelo contextual a un símbolo que se va a transmitir. El contexto se puede referir, por ejemplo, a si los valores vecinos del símbolo son distintos de cero o no. Para realizar la CAVLC, el codificador de vídeo 20 puede seleccionar un código de longitud variable para un símbolo que se va a transmitir. Las palabras de código en la VLC se pueden construir de modo que los códigos relativamente más cortos corresponden a símbolos más probables, mientras que los códigos más largos corresponden a símbolos menos probables. De esta forma, el uso de la VLC puede lograr un ahorro en bits con respecto, por ejemplo, al uso de palabras de código de igual longitud para cada símbolo que se va a transmitir. La determinación de la probabilidad se puede basar en un contexto asignado al símbolo.

[0037] El decodificador de vídeo 30 implementa el inverso de las técnicas del codificador de vídeo 20. Por ejemplo, el decodificador de vídeo 30 decodifica el flujo de bits de vídeo codificado y determina los bloques residuales mediante cuantificación inversa y transformación inversa. El decodificador de vídeo 30 suma los bloques residuales con bloques de imágenes decodificadas previamente para determinar los valores de píxeles para los bloques dentro de la imagen.

[0038] Como se describe anteriormente, las técnicas descritas en esta divulgación están dirigidas a 3D-AVC. Para comprender mejor las técnicas, a continuación se describen algunas técnicas de codificación H.264/AVC, codificación de vídeo multivista desde la perspectiva de la ampliación H.264/MVC y la norma de Codificación de Vídeo de Alta Eficacia (HEVC) y técnicas de 3D-AVC.

[0039] Para H.264/Codificación Avanzada de Vídeo (AVC), la codificación o descodificación de vídeo (por ejemplo, codificación) se implementa en macrobloques, donde un macrobloque representa una porción de una trama que se interpredice o se intrapredice (es decir, interpredicción codificada o decodificada o intrapredicción codificada o decodificada). Por ejemplo, en H.264/AVC, cada macrobloque (MB) en inter (por ejemplo, macrobloque interpredicho) se puede dividir en cuatro formas diferentes: una división de 16x16 MB, dos divisiones de 16x8 MB, dos divisiones de 8x16 MB o cuatro divisiones de 8x8 MB. Las diferentes divisiones de MB de un MB pueden tener diferentes valores del índice de referencia para cada dirección (es decir, RefPicList0 o RefPicList1). Cuando un MB no se divide en múltiples (más de 1) divisiones de MB, tiene solamente un vector de movimiento para toda la división de MB en cada dirección.

[0040] Como parte de la codificación de vídeo (codificación o descodificación), el codificador de vídeo 20 y el decodificador de vídeo 30 se configuran para construir una o dos listas de imágenes de referencia, denominadas RefPicList0 y RefPicList1. La(s) lista(s) de imágenes de referencia identifican imágenes de referencia que se pueden usar para interpredecir macrobloques de una trama o un segmento. Por ejemplo, el codificador de vídeo 20 puede señalar un índice de referencia y un identificador de lista de imágenes de referencia. El decodificador de vídeo 30 puede recibir el índice de referencia y el identificador de la lista de imágenes de referencia y determinar la imagen de referencia que se va a usar para descodificar por interpredicción el macrobloque actual del índice de referencia y el identificador de la lista de imágenes de referencia.

[0041] Cuando un MB se divide en cuatro divisiones de 8x8 MB , cada división de 8x8 MB se puede dividir además en subbloques. Existen cuatro formas diferentes de obtener subbloques de una división de 8x8 MB: un subbloque 8x8, dos subbloques 8x4, dos subbloques 4x8 o cuatro subbloques 4x4. Cada subbloque puede tener un vector de movimiento diferente en cada dirección, pero comparte el mismo índice de imágenes de referencia para cada dirección. La manera en que una división de 8x8 MB se divide en subbloques se denomina división de subbloques.

[0042] Para la codificación de vídeo multivista existen múltiples normas de codificación de vídeo diferentes. Para evitar confusiones, cuando esta divulgación describe genéricamente la codificación de vídeo multivista, esta divulgación usa la frase "codificación de vídeo multivista". En general, en la codificación de vídeo multivista, existe una vista básica y una o más vistas de potenciación o dependientes. La vista básica es totalmente descodificable sin referencia a ninguna de las vistas dependientes (es decir, la vista básica solamente se interpredice con vectores de movimiento temporales). Esto permite que un códec que no está configurado para la codificación de vídeo multivista todavía reciba al menos una vista que sea totalmente descodificable (es decir, la vista básica se puede extraer y las otras vistas se descartan, permitiendo que un decodificador no configurado para la codificación de vídeo multivista todavía decodifique el

contenido del vídeo aunque sin experiencia en 3D). La una o más vistas de potenciación o dependientes se pueden interpredecir con respecto a la vista básica o con respecto a otra vista de potenciación o vista dependiente (es decir, se predice la compensación de disparidad), o con respecto a otras imágenes en la misma vista (es decir, se predice el movimiento compensado).

[0043] Mientras que la "codificación de vídeo multivista" se usa genéricamente, el acrónimo MVC está asociado con una ampliación de H.264/AVC. En consecuencia, cuando la divulgación usa el acrónimo MVC, la divulgación se refiere específicamente a la ampliación de la norma de codificación de vídeo H.264/AVC. La ampliación MVC de H.264/AVC se basa en los vectores de movimiento de disparidad como otro tipo de vector de movimiento además de los vectores de movimiento temporales. También se ha desarrollado otra norma de codificación de vídeo, denominada MVC más profundidad (MVC+D), por JCT-3V y MPEG. MVC+D aplica las mismas herramientas de codificación de bajo nivel que las de MVC para tanto la textura como la profundidad, y la descodificación de la profundidad es independiente de la descodificación de la textura y viceversa. Por ejemplo, en MVC, una trama se representa solamente por un componente de vista, denominado componente de vista de textura, o simplemente textura. En MVC+D, existen dos componentes de vista: el componente de vista de textura y el componente de vista de profundidad, o simplemente textura y profundidad. Por ejemplo, en MVC+D, cada vista incluye una vista de textura y una vista de profundidad, donde la vista incluye una pluralidad de componentes de vista, la vista de textura incluye una pluralidad de componentes de vista de textura y la vista de profundidad incluye una pluralidad de componentes de vista de profundidad.

[0044] Cada componente de vista de textura está asociado con un componente de vista de profundidad para formar un componente de vista de una vista. El componente de vista de profundidad representa la profundidad relativa de los objetos en el componente de vista de textura. En MVC+D, el componente de vista de profundidad y el componente de vista de textura se pueden decodificar por separado. Por ejemplo, el decodificador de vídeo 30 puede implementar dos casos de un códec MVC, en el que un primer códec decodifica los componentes de vista de textura y un segundo códec decodifica los componentes de vista de profundidad. Estos dos códecs se pueden ejecutar de forma independiente uno del otro, porque los componentes de vista de textura y los componentes de vista de profundidad se codifican por separado.

[0045] En MVC+D, un componente de vista de profundidad siempre sigue inmediatamente al componente de vista de textura asociado (por ejemplo, correspondiente). De esta manera, MVC+D admite la codificación de textura primero, donde el componente de vista de textura se decodifica antes del componente de vista de profundidad.

[0046] Un componente de vista de textura y su componente de vista de profundidad asociado (por ejemplo, correspondiente) puede incluir el mismo valor de recuento de orden de imágenes (POC) y la ID de la vista (es decir, el valor POC y la ID de la vista de un componente de vista de textura y su componente de vista de profundidad asociado es el mismo). El valor POC indica el orden de visualización del componente de vista de textura y la ID de la vista indica la vista a la que pertenecen el componente de vista de textura y el componente de vista de profundidad.

[0047] Una orden de descodificación MVC típica (es decir, orden de flujo de bits) se muestra en la figura. 2. La disposición de orden de descodificación se denomina codificación de tiempo primero. Se debe tener en cuenta que el orden de descodificación de las unidades de acceso puede no ser idéntico al orden de salida o de visualización. En la FIG. 2, S0-S7 se refieren, cada uno, a diferentes vistas del vídeo de multivista. Cada una de T0-T8 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 T0, una segunda unidad de acceso puede incluir todas las vistas S0-S7 para la instancia de tiempo T1 y así sucesivamente.

[0048] Para propósitos de brevedad, la divulgación puede usar las siguientes definiciones:

componente de vista: Una *representación codificada* de una *vista* en una única *unidad de acceso*. Cuando una vista incluye tanto representaciones de textura como de profundidad codificadas, un componente de vista puede incluir un componente de vista de textura y un componente de vista de profundidad.

componente de vista de textura: Una *representación codificada* de la textura de una vista en una única *unidad de acceso.*

componente de vista de profundidad: Una *representación codificada* de la profundidad de una vista en una única *unidad de acceso.*

[0049] Como se analiza anteriormente, en el contexto de esta divulgación, el componente de vista, el componente de vista de textura, y el componente de vista de profundidad se pueden denominar, en general, una capa. En la FIG. 2, 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 y así sucesivamente. 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 dentro de un conjunto de imágenes de una vista se pueden considerar como correspondientes entre sí. Por ejemplo, el componente de vista de textura dentro 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, el componente de vista de profundidad corresponde a su componente de vista de textura en el conjunto, y viceversa). Como se usa en esta divulgación, un componente de vista de textura que corresponde a un componente de vista de profundidad se puede considerar como el componente de vista de textura y el componente de vista de profundidad que son parte de una misma vista de una única unidad de acceso.

[0050] El componente de vista de textura incluye el contenido de la imagen real que se visualiza. Por ejemplo, el componente de vista de textura puede incluir los componentes de luma (Y) y croma (Cb y Cr). El componente de vista de profundidad puede indicar profundidades relativas de los píxeles en su componente de vista de textura correspondiente. Como un ejemplo, el componente de vista de profundidad es una imagen en escala de grises que incluye solamente valores de luma. En otras palabras, el componente de vista de profundidad puede que no transmita ningún contenido de imagen, pero en lugar de eso proporciona una medida de las profundidades relativas de los píxeles en el componente de vista de textura.

[0051] Por ejemplo, un píxel blanco puro en el componente de vista de profundidad indica que su correspondiente píxel o píxeles en el componente de vista de textura correspondiente está más cerca de la perspectiva del observador, y un píxel negro puro en el componente de vista de profundidad indica que su correspondiente píxel o píxeles en el 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 muy gris en el componente de vista de profundidad indica que su píxel correspondiente en el componente de vista de textura está más alejado que un píxel ligeramente gris en el componente de vista de profundidad. Dado que solamente es necesaria la escala de grises para identificar la profundidad de los píxeles, el componente de vista de profundidad no necesita incluir componentes de croma, ya que los valores de color para el componente de vista de profundidad pueden no servir para ningún propósito.

[0052] El componente de vista de profundidad que usa solamente valores de luma (por ejemplo, valores de intensidad) para identificar la profundidad se proporciona con propósitos ilustrativos y no se debe considerar limitante. En otros ejemplos, se puede utilizar cualquier técnica para indicar las profundidades relativas de los píxeles en el componente de vista de textura.

[0053] Una estructura de predicción MVC típica (incluyendo tanto la predicción entre imágenes dentro de cada vista como la predicción entre vistas entre las vistas) para la codificación de vídeo multivista se muestra en la figura. 3. Las direcciones de predicción se indican mediante flechas, el objeto al que se apunta que usa el objeto desde el que se apunta como referencia de predicción. En MVC, la predicción entre vistas se admite mediante la compensación de movimiento de disparidad, que usa la sintaxis de la compensación de movimiento H.264/AVC, pero permite usar una imagen de una vista diferente como imagen de referencia.

[0054] En el ejemplo de la FIG. 3, se ilustran ocho vistas (que tienen ID de vista de "S0" hasta "S7"), y se ilustran doce ubicaciones temporales (de "T0" hasta "T11") para cada vista. Es decir, cada fila en la FIG. 3 corresponde a una vista, mientras que cada columna indica una ubicación temporal.

[0055] Aunque la MVC tiene una denominada vista básica que se puede decodificar mediante los decodificadores de la H.264/AVC y se podrían admitir también los pares de vistas en estéreo por la MVC, la ventaja de MVC es que podría admitir un ejemplo que usa más de dos vistas como una entrada de vídeo 3D y que decodifica este vídeo 3D representado por las múltiples vistas. Un renderizador de un cliente que tiene un decodificador de MVC puede esperar contenido de vídeo 3D con múltiples vistas.

[0056] Las imágenes en la FIG. 3 se indican en la intersección de cada fila y cada columna. La norma H.264/AVC puede usar el término trama para representar una porción del vídeo. Esta divulgación puede usar los términos imagen y trama de forma intercambiable.

[0057] Las imágenes de la FIG. 3 se ilustran usando un bloque que incluye una letra, designando la letra si la imagen correspondiente está intracodificada (es decir, una imagen I), o intercodificada en una dirección (es decir, como una imagen P) o en múltiples direcciones (es decir, como una imagen B). En general, las predicciones se indican mediante flechas, donde las imágenes a las que se apunta usan la imagen desde la que se apunta como referencia de predicción. Por ejemplo, la imagen P de la vista S2 en la ubicación temporal T0 se predice a partir de la imagen I de la vista S0 en la ubicación temporal T0.

[0058] Al igual que con la codificación de vídeo de vista única, las imágenes de una secuencia de vídeo de codificación de vídeo de multivista se pueden codificar predictivamente con respecto a las imágenes en diferentes ubicaciones temporales. Por ejemplo, la imagen b de la vista S0 en la ubicación temporal T1 tiene una flecha apuntando a la misma desde la imagen I de la vista S0 en la ubicación temporal T0, indicando que la imagen b se predice a partir de la imagen I. Adicionalmente, sin embargo, en el contexto de la codificación de vídeo de multivista, las imágenes se pueden predecir entre vistas. Es decir, un componente de vista puede usar los componentes de vista de otras vistas

como referencia. En la MVC, por ejemplo, la predicción entre vistas se realiza como si el componente de vista en otra vista es una referencia de interpredicción. Las posibles referencias entre vistas se señalan en la ampliación de MVC del conjunto de parámetros de secuencia (SPS) y se pueden modificar por el proceso de construcción de la lista de imágenes de referencia, que habilita el ordenamiento flexible de las referencias de interpredicción o de predicción entre vistas. La predicción entre vistas también es una característica de la ampliación propuesta de multivista de HEVC, incluyendo 3D-HEVC (multivista más profundidad).

[0059] La FIG. 3 proporciona diversos ejemplos de predicción entre vistas. Las imágenes de la vista S1, en el ejemplo de la FIG. 3, se ilustran como predichas a partir de imágenes en diferentes ubicaciones temporales de la vista S1, así como entre vistas predichas a partir de imágenes de las vistas S0 y S2 en las mismas ubicaciones temporales. Por ejemplo, la imagen b de la vista S1 en la ubicación temporal T1 se predice a partir de cada una de las imágenes B de la vista S1 en las ubicaciones temporales T0 y T2, así como las imágenes b de las vistas S0 y S2 en la ubicación temporal T1.

[0060] En algunos ejemplos, la FIG. 3 se puede ver como una ilustración de los componentes de vista de textura. Por ejemplo, las imágenes I, P, B y b ilustradas en la FIG. 2 se pueden considerar componentes de vista de textura para cada una de las vistas. De acuerdo con las técnicas descritas en esta divulgación, para cada uno de los componentes de vista de textura ilustrados en la FIG. 3 existe un componente de vista de profundidad correspondiente. En algunos ejemplos, los componentes de vista de profundidad se pueden predecir de una manera similar a la ilustrada en la FIG. 3 para los componentes de vista de textura correspondientes.

[0061] La codificación de dos vistas también se puede admitir por MVC. Una de las ventajas de la MVC es que un codificador de MVC puede tomar más de dos vistas como una entrada de vídeo 3D y un decodificador de MVC puede descodificar dicha representación de multivista. Como tal, cualquier renderizador con un decodificador MVC puede decodificar contenidos de vídeo 3D con más de dos vistas.

[0062] Como se analiza anteriormente, en MVC, se permite la predicción entre vistas entre imágenes en la misma unidad de acceso (lo que quiere decir, en algunos casos, con la misma instancia de tiempo). Cuando se codifica una imagen en una de las vistas no básicas, se puede añadir una imagen a una lista de imágenes de referencia si esta está en una vista diferente pero dentro de una misma instancia de tiempo. Una imagen de referencia de predicción entre vistas se puede disponer en cualquier posición de una lista de imágenes de referencia, tal como cualquier imagen de referencia de interpredicción. Como se muestra en la FIG. 3, un componente de vista puede usar los componentes de vista de otras vistas como referencia. En MVC, la predicción entre vistas se realiza como si el componente de vista de otra vista fuera una referencia de interpredicción.

[0063] En MVC, se permite la predicción entre vistas entre imágenes en la misma unidad de acceso (es decir, con la misma instancia de tiempo). Cuando se codifica una imagen en una de las vistas no básicas, se puede añadir una imagen a una lista de imágenes de referencia si esta está en una vista diferente pero con una misma instancia de tiempo. Una imagen de referencia de predicción entre vistas se puede disponer en cualquier posición de una lista de imágenes de referencia, tal como cualquier imagen de referencia de interpredicción.

[0064] Como se muestra en la FIG. 3, un componente de vista puede usar los componentes de vista de otras vistas como referencia. Esto se llama predicción entre vistas. En MVC, la predicción entre vistas se realiza como si el componente de vista de otra vista fuera una referencia de interpredicción.

[0065] En el contexto de codificación de vídeo multivista, existen dos tipos de vectores de movimiento. Uno es un vector de movimiento normal que apunta a imágenes de referencia temporales. La interpredicción temporal correspondiente es la predicción compensada por movimiento (MCP). El otro tipo de vector de movimiento es un vector de movimiento de disparidad que apunta a imágenes en una vista diferente (es decir, imágenes de referencia intervista). La interpredicción correspondiente es la predicción compensada por disparidad (DCP).

[0066] En la actualidad, un Equipo de Colaboración Conjunta en Codificación de Vídeo 3D (JCT-3V) de VCEG y MPEG está desarrollando 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 intervista en MVC. El software 3D-ATM más reciente para 3D-AVC se puede descargar desde el siguiente enlace: [3D-ATM versión 6.2]: http://mpeg3dv.research.nokia.com/svn/mpeg3dv/tags/3DV-ATMv6.2/

[0067] La norma de codificación de vídeo 3D basado en AVC (3D-AVC) está actualmente en desarrollo por JCT-3V, y la versión más reciente de 3D-AVC ahora está disponible al público: MM Hannuksela, Y. Chen, T. Suzuki, J.-R. Ohm, G.J. Sullivan, "3D-AVC draft text 5 [Borrador de texto de 3D-AVC 5]", JCT3V-C1002, Ginebra, CH, enero de 2013. Está disponible en el siguiente enlace http://phenix.it-sudparis.eu/jct2/doc_end_user/documents/3_Geneva/wg11/JCT3V-C1002-v3.zip.

[0068] La 3D-AVC es compatible con la H.264/AVC de tal forma que la parte de textura de la vista básica es totalmente descodificable para el decodificador H.264/AVC. Por ejemplo, los componentes de vista de textura en los componentes de vista de la vista básica solamente se pueden interpredecir con otros componentes de vista de textura en la misma

vista básica. Es posible que los componentes de vista de textura en la vista básica no se predigan entre vistas. También, es posible que el componente de vista de textura en la vista básica no requiera el componente de vista de profundidad correspondiente para propósitos de descodificación.

[0069] Para los componentes vista potenciados en 3D-AVC, en algunas otras técnicas de ejemplo, la profundidad se puede codificar antes de la textura y un componente de vista de textura se puede codificar en base a la información del componente de vista de profundidad, que también se conoce como codificación de profundidad primero. Sin embargo, cada componente de vista de textura se codifica antes que los componentes de vista de profundidad respectivos en el orden de codificación de textura primero, tal como en MVC+D descrita anteriormente. En otras palabras, en algunas otras técnicas de ejemplo, en 3D-AVC, el componente de vista de textura de la vista básica se codifica primero, seguido del componente de vista de profundidad asociado de la vista básica, seguido del componente de vista de profundidad de una primera vista de potenciación o dependiente, seguido del componente de vista de textura asociado de la primera vista de potenciación o dependiente, seguido del componente de vista de profundidad de una segunda vista de potenciación o dependiente, seguido del componente de vista de textura asociado de la segunda vista de potenciación o dependiente, y así sucesivamente .

[0070] Por ejemplo, las órdenes de codificación de los componentes de vista de textura y profundidad en 3D-AVC se ejemplifican como sigue. En los siguientes ejemplos, T0 y D0, respectivamente, se refieren a los componentes de vista de textura y profundidad de la vista básica, y Ti y Di, respectivamente, se refieren a los componentes de vista de textura y profundidad de la i-ésima vista dependiente. En los siguientes ejemplos, se consideran tres vistas.

[0071] En un primer ejemplo, las vistas consideradas son T0, D0, D1, D2, T1 y T2. En este ejemplo, las vistas básicas (T0 y D0) se codifican con el orden de codificación de textura primero, mientras que las vistas dependientes se codifican con el orden de codificación de profundidad primero. Actualmente se usa un orden de codificación híbrido en condiciones de prueba comunes de 3D-AVC. En otro ejemplo, el orden de codificación es T0, D0, T1, D1, T2 y D2. Es decir, todos los componentes de vista se codifican con el orden de codificación de textura primero. Si la predicción entre vistas está habilitada para Ti, la vista de 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 de la vista de textura de referencia.

[0072] Algunas otras técnicas de 3D-AVC requerían codificación de profundidad primero porque derivar un vector de disparidad para un bloque del componente de vista de textura requería el correspondiente componente de vista de profundidad. A continuación se describe dicha derivación del vector de disparidad por medio 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 la vista dependiente se usan para la derivación del vector de disparidad para la codificación del componente de vista de textura. Esto es porque la vista de profundidad de la vista dependiente está disponible, debido al orden de codificación de profundidad primero. Las herramientas de codificación de bajo nivel usadas son la predicción entre vistas de síntesis de vista basada en bloques en bucle (BVSP) y la predicción de vectores de movimiento basada en profundidad (D-MVP) en 3D-AVC. Un codificador de vídeo, por ejemplo, el decodificador de vídeo 30 puede usar el vector de disparidad convertido a partir de los valores de profundidad de la vista de profundidad (a veces llamado mapa de profundidad) en la vista dependiente (a veces llamada trama dependiente). En el software de referencia 3D-AVC, típicamente, los resultados del proceso de conversión del valor del mapa de profundidad real a una disparidad con una vista particular se almacenan en tablas de consulta con parámetros de cámara.

[0073] La FIG. 4 es un diagrama conceptual de la BVSP en base a la distorsión inversa. BVSP se propuso originalmente en "3DV-CE1.a: Block-Based View Synthesis Prediction for 3DV-ATM [3DV-CE1.a: Predicción de síntesis de vista basada en bloques para 3DV-ATM]" (JCT3V-A0107) por W. Su, *et al*., que se puede descargar desde el siguiente enlace: http://phenix.it-sudparis.eu/jct2/doc_end_user/documents/1_Stockholm/wg11/JCT3V-A0107-v1.zip. Haciendo referencia a la FIG. 4, se supone 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 VSP (predicción de síntesis de vista). Los componentes de mapa de profundidad D0 y D1 son mapas de profundidad respectivos asociados con T0 y T1.

[0074] En la vista dependiente T1, los valores de muestra del bloque codificado Cb actualmente se predicen a partir del área de referencia R(Cb) (predicción VSP) que consiste en los valores de muestra de la vista básica T0. El vector de desplazamiento (Disp_vec) entre las muestras actuales que se van a codificar y las muestras de referencia se indica como un 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.

[0075] El proceso de conversión de un valor de profundidad a un vector de disparidad se puede realizar por ejemplo con las siguientes ecuaciones:

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

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

donde j e i son coordenadas espaciales locales dentro de Cb, d *Cb*(*j,i*) es un valor de mapa de profundidad en la imagen de mapa de profundidad de una vista 1, Z es el valor de profundidad correspondiente real y D es la componente horizontal de un vector de disparidad derivado a una vista particular 0. Los parámetros f, b, Zcerca y Zlejos son parámetros que especifican la configuración de la cámara, es decir, la distancia focal usada (f), la separación de la cámara (b) entre la vista # 1 y la vista # 0, y el intervalo de profundidad (Zcerca, Zlejos) representan parámetros de conversión del mapa de profundidad.

[0076] Se debe tener en cuenta que, en algunos ejemplos, la componente vertical del vector de disparidad derivado se establece en 0. También, en algunas implementaciones de 3DV-ATM, las ecuaciones (1) y (2) ya se han calculado previamente para cada valor del mapa de profundidad (0...255) y se almacenaron como una tabla de consulta.

[0077] En la siguiente sección se analizarán varias cuestiones de implementación de BVSP. Una cuestión implica la indicación de bloques BVSP. Los bloques BVSP se indican de la siguiente manera:

- Se usa un indicador en el nivel de MB para señalar si el MB actual se codifica con el modo de salto/directo convencional o si se codifica con el modo de salto/directo pero se predice a partir de un componente de referencia sintético.

- Para cada división de MB (de 16x16 a 8x8), se usa un índice de referencia (o un indicador, como en algunas propuestas para 3D-AVC) en cada lista de imágenes de referencia para señalar la imagen de referencia. Cuando se codifica una división en el modo BVSP, las diferencias del vector de movimiento no se señalan, ya que no existen vectores de movimiento para los bloques codificados BVSP.

[0078] Cuando el indicador o bien el índice de referencia indica un componente de referencia sintético, se aplica la predicción de una división como se describe en el siguiente punto. Para cada división de MB, con su tamaño indicado por NxM (en el que N o M serán 8 o 16), si la división de MB se codifica con el modo BVSP, una división de MB actual se divide adicionalmente en varias subregiones con un tamaño igual a KxK (en el que K puede ser 8x8, como en algunas propuestas para 3D-AVC, 4x4, 2x2 o 1x1). Para cada subregión, se deriva un vector de disparidad separado y cada subregión se predice a partir de un bloque ubicado por el vector de disparidad derivado en la imagen de referencia entre vistas, es decir, R(cb) en la FIG. 4. En algunas condiciones de prueba comunes de ejemplo, K se define como 4. Se debe tener en cuenta que los vectores de disparidad derivados no se almacenan para los bloques codificados BVSP, ya que no existe ninguna herramienta de codificación que use dichos vectores.

[0079] Otra cuestión de implementación implica el proceso de derivación del vector de disparidad. Cuando se aplica el orden de codificación de profundidad primero, el vector de disparidad derivado se puede obtener al convertir un valor de profundidad del bloque de profundidad correspondiente 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 dentro de un bloque de profundidad, el valor máximo de cuatro píxeles de esquina dentro 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, los modos de BVSP se inhabilitarán, ya que la vista de profundidad no básica correspondiente no está disponible cuando se decodifica la vista de textura no básica.

[0080] A continuación se analizará la predicción de vectores de movimiento basada en profundidad (D-MVP) en 3D-AVC para los modos inter normales. D-MVP se refiere a un procedimiento de predicción de vectores de movimiento que incorpora los datos de mapa de profundidad asociados en la vista actual, que está disponible debido al orden de codificación de profundidad primero. El procedimiento se aplica con los componentes de vista de textura en vistas dependientes.

[0081] 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, si un vector de movimiento temporal o vector de movimiento de disparidad) se identifica primero a partir de los índices de referencia de los vectores de movimiento en bloques vecinos, y por tanto se determina el tipo de predicción de movimiento. Como se muestra en la FIG. 8, los bloques vecinos para una división actual pueden incluir, en orden, un bloque izquierdo (indicado como 'A'), un bloque arriba (indicado como 'B'), un bloque arriba a la derecha (indicado como 'C') y un bloque arriba a la izquierda (indicado como 'D') con respecto al bloque actual. El vector de

movimiento en el bloque arriba a la izquierda se puede usar cuando uno de los otros tres bloques vecinos no contenga un vector de movimiento y, por tanto, se considera que no está disponible.

[0082] Suponiendo que los vectores de movimiento de los tres bloques vecinos están disponibles, los vectores de movimiento en los tres bloques vecinos se emplean para la predicción de vectores de movimiento del bloque actual. En la predicción temporal, si todos sus vectores de movimiento tienen el mismo tipo y tienen los mismos índices de referencia, se usa directamente un filtro de mediana, como en H.264/AVC. De otro modo, si los vectores de movimiento pertenecen a tipos diferentes y tienen índices de referencia diferentes, se deriva además un vector de movimiento para el bloque actual. Cuando la imagen de referencia actual es una imagen de referencia entre vistas, se verifican los tipos de vectores de movimiento y sus índices de referencia en las posiciones de bloques vecinos. Si todos los bloques vecinos tienen el mismo tipo y los mismos índices de referencia, se aplica el filtro de mediana. En ambos casos, si están disponibles menos de tres bloques vecinos, los vectores de movimiento para los bloques no disponibles se derivan adicionalmente para que tres bloques vecinos se vuelvan disponibles.

[0083] Un vector de movimiento derivado para un bloque vecino se llama un vector de movimiento derivado, y se genera como sigue. Si el vector de movimiento actual es un vector de movimiento de disparidad, y el vector de movimiento del bloque vecino tiene un tipo diferente al del vector de movimiento actual (o no está disponible), el vector de movimiento derivado del bloque vecino se establece para que sea un vector de movimiento de disparidad, que se convierte a partir del componente de vista de profundidad correspondiente. Se usa el bloque correspondiente del componente de vista de profundidad de la misma vista, y el valor máximo de los valores de profundidad de las cuatro esquinas de este bloque correspondiente se convierte en un valor de disparidad, que se convierte en la componente horizontal del vector de movimiento derivado. La componente vertical del vector de movimiento derivado se establece para que sea cero.

[0084] Si el vector de movimiento actual es un vector de movimiento temporal, el valor de disparidad (derivado como se analiza anteriormente) se usa para determinar un vector de movimiento temporal del bloque de referencia en la vista de referencia (básica), y se establece el vector de movimiento derivado para que sea el vector de movimiento temporal. Si se considera que el vector de movimiento temporal no está disponible (por ejemplo, en el caso de un bloqueo intra, o si el vector de movimiento no apunta a una imagen de referencia en la vista de referencia alineada con la imagen de referencia actual), el vector de movimiento derivado se establece en cero.

[0085] A continuación se analizará la predicción de movimiento entre vistas en 3D-AVC para los modos de salto y directo. La predicción de movimiento entre vistas en 3D-AVC se realiza en salto de P, salto de B, modo directo de B 16x16 y modo directo de B 8x8. Un vector de disparidad se puede derivar inicialmente a partir de los bloques vecinos, así como derivar un vector de disparidad convertido a partir de los valores de profundidad del componente de vista de profundidad de la misma vista.

[0086] Si un bloque vecino espacial disponible contiene un vector de movimiento de disparidad, este vector de movimiento de disparidad se convierte en el vector de disparidad para el bloque actual. De otro modo, para los bloques vecinos que no contienen un vector de movimiento de disparidad, un vector de movimiento de disparidad que se va a usar para el bloque actual se convierte a partir de los valores de profundidad correspondientes a la misma vista (similar a la conversión en D-MVP). En algunos ejemplos, se aplica un filtro de mediana a tres bloques vecinos para obtener un vector de disparidad.

[0087] El vector derivado se puede usar para obtener un vector de movimiento temporal con respecto al bloque de referencia en la vista de referencia (básica). Si el vector de movimiento temporal no está disponible, primero se puede derivar el índice de referencia, y el proceso D-MVP, analizado anteriormente, se aplica para producir un predictor de vector de movimiento.

[0088] A continuación se analizará la derivación del vector de disparidad basado en bloques vecino (NBDV). La NBDV se usa como un procedimiento de derivación del vector de disparidad en 3D-HEVC cuando se usa el orden de codificación de textura primero para todas las vistas. En el diseño actual de 3D-HEVC, la derivación de NBDV también se usa para recuperar datos de profundidad del mapa de profundidad de la vista de referencia.

[0089] Una versión de la descripción del software de referencia, así como el borrador de trabajo de 3D-HEVC es que esté disponible como sigue: Gerhard Tech, Krzysztof Wegner, Ying Chen, Sehoon Yea, "3D-HEVC Test Model Description draft 2 [Borrador 2 de la Descripción del modelo de prueba de 3D-HEVC]", JCT3V-B1005, Equipo de Colaboración Conjunta en Desarrollo de Ampliación de Codificación de Vídeo 3D de ITU-T SG 16 WP 3 e ISO/IEC JTC 1/SC 29/WG 11, 2.ª conferencia: Shanghai, CN, octubre de 2012.

[0090] Se usa un vector de disparidad (DV) como estimador de la disparidad entre dos vistas. Es decir, un vector de disparidad es un puntero, con respecto a un bloque en una imagen actual, a un bloque correspondiente en una imagen ya codificada en la misma instancia de tiempo. 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 predictor. Siguiendo esta idea, el proceso de derivación de NBDV usa la información de disparidad de los bloques vecinos para estimar el vector de disparidad en diferentes vistas.

[0091] Para realizar la derivación de NDBD, se definen inicialmente los bloques vecinos candidatos. Se utilizan dos conjuntos de bloques candidatos vecinos. Un conjunto es de bloques vecinos espaciales y el otro conjunto es de bloques vecinos temporales. Cada uno de los bloques candidatos vecinos espaciales y temporales se verifica luego en un orden predefinido determinado por la prioridad de la correlación entre el bloque actual y el bloque candidato. Una vez que un vector de movimiento de disparidad (es decir, el vector de movimiento apunta a una imagen de referencia entre vistas) se encuentra en los candidatos, el vector de movimiento de disparidad se convierte en un vector de disparidad.

[0092] a continuación se analizarán los ejemplos específicos de derivación de NBDV en 3D-HEVC. 3D-HEVC adoptó primero el procedimiento de vector de disparidad de (basado en) bloque vecino (NBDV) propuesto en JCT3V-A0097. Se incluyeron vectores de disparidad implícita con un proceso de derivación de NBDV simplificado en JCTVC-A0126. Además de eso, en JCT3V-B0047, la derivación de NBDV se simplifica adicionalmente eliminando los vectores de disparidad implícitos almacenados en la memoria intermedia de imágenes decodificadas, pero también mejoró la ganancia de codificación con la selección de imagen del punto de acceso aleatorio (RAP).

[0093] JCT3V-A0097: 3D-CE5.h: Disparity vector generation results [3D-CE5.h: Resultados de la generación del vector de disparidad], L. Zhang, Y. Chen, M. Karczewicz (Qualcomm).

[0094] JCT3V-A0126: 3D-CE5.h: Simplification of disparity vector derivation for HEVC-based 3D video coding [3D-CE5.h: Simplificación de la derivación del vector de disparidad para codificación de vídeo 3D basada en HEVC], J. Sung, M. Koo, S. Yea (LG).

[0095] JCT3V-B0047: 3D-CE5.h related: Improvements for disparity vector derivation [en referencia a 3D-CE5.h: Mejoras para la derivación de vector de disparidad], J. Kang, Y. Chen, L. Zhang, M. Karczewicz (Qualcomm).

[0096] En algunas propuestas de derivación de NBDV, se usan cinco bloques vecinos espaciales para la derivación del vector de disparidad. Como se muestra en la FIG. 5, los cinco bloques vecinos espaciales son los bloques debajo a la izquierda, izquierda, arriba a la derecha, arriba y arriba a la izquierda de la PU 500 actual, como se indica por A0, A1, B0, B1 o B2. Cabe destacar que son los mismos que los usados en los modos de combinación en HEVC. Por lo tanto, no se requiere acceso adicional a la memoria.

[0097] Antes de comprobar los bloques vecinos temporales, se realiza primero un proceso de construcción de la lista de imágenes candidatas. Todas las imágenes de referencia de la vista actual se pueden tratar como imágenes candidatas. Una imagen de referencia coubicada se inserta primero en la lista de imágenes candidatas, seguido del resto de las imágenes candidatas en el orden ascendente del índice de referencia. Cuando las imágenes de referencia con el mismo índice de referencia en ambas listas de imágenes de referencia están disponibles, la imagen de referencia que está en la misma lista de imágenes de referencia que la imagen coubicada precede a la otra imagen de referencia que tiene el índice de referencia coincidente. Para cada imagen candidata en la lista de imágenes candidatas, se determinan tres regiones candidatas para derivar los bloques vecinos temporales.

[0098] Cuando se codifica un bloque con predicción de movimiento entre vistas, un vector de disparidad se deriva para seleccionar un bloque correspondiente en una vista diferente. El vector de disparidad derivado en la predicción de movimiento entre vistas se puede denominar un vector de disparidad implícita (IDV). Aunque el bloque se codifique con predicción de movimiento, los IDV no se descartan con el propósito de codificación de un bloque siguiente.

[0099] Típicamente, el proceso de derivación de NBDV implica la comprobación de los vectores de movimiento de disparidad en los bloques vecinos temporales, los vectores de movimiento de disparidad en los bloques vecinos espaciales, y luego el IDV, en ese orden. Una vez que se encuentra el vector de disparidad, se termina el proceso.

[0100] A continuación se analizará VSP inversa en 3D-HEVC. En 3D-HEVC, cuando se aplica el orden de codificación de textura primero, para cada unidad de predicción (PU), un vector de disparidad se podría derivar del proceso de derivación de NBDV con o sin considerar los valores de profundidad en la vista de profundidad de referencia. Después de obtener un vector de disparidad, se refinará adicionalmente para cada subregión MxN (en la que M/N puede ser igual a 8 o 4, por ejemplo) de una PU si se codifica con el modo de BVSP.

[0101] El proceso de refinamiento incluye dos pasos: 1) seleccione un valor de profundidad máxima del bloque de profundidad MxN en la vista de profundidad de referencia que se ubica por el vector de disparidad derivado; 2) convertir el valor de profundidad en un componente horizontal del vector de disparidad refinado mientras se mantiene el componente vertical del vector de disparidad refinado para que sea 0. Después de que el vector de disparidad se refine para una subregión MxN de una PU, el vector de disparidad refinado se usa para ubicar un bloque en la vista de textura de referencia para la compensación de movimiento.

[0102] A continuación se analizará el proceso de derivación de NBDV en 3D-AVC. Como se describe en la solicitud de patente de EE. UU. n.º 14/189,177, presentada el 25 de febrero de 2014, la derivación de NBDV a nivel de MB se puede usar para derivar un vector de disparidad para el MB actual. El vector de disparidad derivado se puede usar

además para la predicción de vectores de movimiento. Una vez que se identifica un vector de movimiento de disparidad, es decir, uno de los bloques vecinos temporales o espaciales usa la imagen de referencia entre vistas, se devuelve como el vector de disparidad para el MB actual.

[0103] A continuación se describen las técnicas de la solicitud de patente de EE. UU. n.º 14/189,177 con más detalle. Algunas técnicas previas de 3D-AVC requerían que el componente de vista de profundidad del componente de vista de textura estuviera disponible para la derivación del vector de disparidad (es decir, requiere una codificación de profundidad primero para vistas dependientes o de potenciación), lo que da lugar a problemas tales como la latencia de descodificación, la complejidad de la implementación, falta de escalabilidad a otras normas de codificación de vídeo, ineficiencias en el ancho de banda si no se necesitan componentes de vista de profundidad y otros inconvenientes potenciales.

[0104] Las técnicas descritas en la solicitud de patente de EE. UU. n.º 14/189,177 permiten la derivación del vector de disparidad que no requiere que se base en el componente de vista profundidad correspondiente. De esta manera, las técnicas permiten la codificación de textura primero en 3D-AVC para vistas dependientes con derivación del vector de disparidad. Para lograr una derivación del vector de disparidad, las técnicas descritas en la solicitud de patente de EE. UU. n.º 14/189,177 se basan en la información de vectores de movimiento de bloques vecinos. Como ejemplo, si un vector de movimiento para un bloque vecino es un vector de movimiento de disparidad, las técnicas utilizan el vector de movimiento de disparidad del bloque vecino como un vector de disparidad para el bloque actual. De esta manera, el codificador de vídeo 20 y el decodificador de vídeo 30 pueden determinar un vector de disparidad para el macrobloque actual de un componente de vista de textura sin necesidad de basarse en el componente de vista de profundidad correspondiente.

[0105] El codificador de vídeo 20 y el decodificador de vídeo 30 se pueden configurar para implementar las técnicas descritas en la solicitud de patente de EE. UU. n.º 14/189,177. Por ejemplo, el codificador de vídeo 20 y el decodificador de vídeo 30 se pueden configurar para implementar técnicas que posibiliten la codificación eficaz de 3D-AVC permitiendo que la vista de textura se codifique primero para cada componente de vista. El codificador de vídeo 20 y el decodificador de vídeo 30 pueden derivar un vector de disparidad usando una noción del NBDV que considera más de un vector de movimiento de disparidad disponible de los bloques vecinos espaciales/temporales del bloque actual cuando los datos de profundidad correspondientes no están disponibles (o aún no están disponibles) en 3D-AVC debido a un orden de codificación de textura primero.

[0106] Como ejemplo, el decodificador de vídeo 30 puede recibir un flujo de bits codificado en un proceso de codificación de vídeo compatible con AVC 3D generado con la codificación de textura primero de vistas dependientes. En este ejemplo, el proceso de codificación de vídeo compatible con 3D-AVC se refiere a un proceso de codificación de vídeo que usa herramientas de codificación de vídeo definidas en la norma de codificación de vídeo 3D-AVC. La codificación de textura primero de las vistas dependientes se refiere al caso donde los componentes de vista de textura se codifican antes de los componentes de vista de profundidad correspondientes (es decir, T0, D0, T1, D1 y así sucesivamente).

[0107] El decodificador de vídeo 30 puede decodificar un componente de vista de textura de una vista dependiente de las vistas dependientes en el proceso de codificación de vídeo compatible con AVC 3D. En este ejemplo, para decodificar el componente de vista de textura, el decodificador de vídeo 30 se puede configurar para evaluar la información de movimiento de uno o más bloques vecinos de un bloque actual en el componente de vista de textura para determinar si al menos un bloque vecino se predice entre vistas con un vector de movimiento de disparidad que se refiere a una imagen de referencia entre vistas en una vista distinta de la vista dependiente. También, para decodificar el componente de vista de textura, el decodificador de vídeo 30 se puede configurar para derivar un vector de disparidad para el bloque actual en base al vector de movimiento de disparidad para uno de los bloques vecinos. El decodificador de vídeo 30 puede decodificar un componente de vista de profundidad que corresponde al componente de vista de textura subsiguiente a decodificar el componente de vista de textura.

[0108] Como otro ejemplo, el codificador de vídeo 20 puede codificar un componente de vista de textura de una vista dependiente en un proceso de codificación de vídeo compatible con 3D-AVC. En este ejemplo, para codificar el componente de vista de textura, el codificador de vídeo 20 se puede configurar para evaluar la información de movimiento de uno o más bloques vecinos de un bloque actual en el componente de vista de textura para determinar si al menos un bloque vecino se predice entre vistas con un vector de movimiento de disparidad que se refiere a una imagen de referencia entre vistas en una vista distinta de la vista dependiente. También, para codificar el componente de vista de textura, el codificador de vídeo 20 se puede configurar para derivar un vector de disparidad para el bloque actual en base al vector de movimiento de disparidad para uno de los bloques vecinos.

[0109] El codificador de vídeo 20 puede codificar un componente de vista de profundidad que corresponde al componente de vista de textura subsiguiente a la codificación del componente de vista de textura. El codificador de vídeo 20 también puede generar para la salida un flujo de bits codificado con una codificación de textura primero de vistas dependientes que incluye el componente de vista de textura codificada y el componente de vista de profundidad codificada.

[0110] Las propuestas actuales para el 3D-AVC presentan los siguientes problemas. Cuando se utiliza el procedimiento de NBDV descrito en la solicitud de patente de EE. UU. n.º 14/189,177, la BVSP se vuelve menos eficaz, principalmente debido al motivo de que los vectores de disparidad no siempre son lo suficientemente exactos. También, los vectores de disparidad derivados (por ejemplo, de NBDV) para los bloques de BVSP pueden proporcionar vectores de disparidad más exactos para el bloque que se va a codificar. Sin embargo, el uso de dichos vectores de disparidad derivados en BVSP no se ha empleado previamente con la derivación de NBDV.

[0111] En vista de estos inconvenientes, esta divulgación proporciona soluciones para habilitar BVSP para codificadores de vídeo y decodificadores de vídeo compatibles con 3D-AVC cuando el componente de vista de textura no básica se codifica antes que el componente de vista de profundidad no básica correspondiente. Además, la ganancia de codificación de otros modos de intercodificación también se mejora debido a la derivación de un vector de disparidad elaborado, como se proporciona por las técnicas de esta divulgación.

[0112] Inicialmente, se propone un proceso de derivación de NBDV mejorado en esta divulgación. El proceso de derivación de NBDV de esta divulgación se modifica para incorporar BVSP, aunque determinados aspectos de esta divulgación no necesariamente requieren BVSP. La siguiente descripción se describirá con referencia al decodificador de vídeo 30, aunque se debe entender que cada una de las técnicas a continuación también se puede implementar por el codificador de vídeo 20. Tanto el decodificador de vídeo 30 como el codificador de vídeo 20 se pueden implementar con uno o más procesadores configurados para ejecutar las técnicas de esta divulgación. En algunos ejemplos, el uno o más procesadores del decodificador de vídeo 30 y el codificador de vídeo 20 se pueden configurar para ejecutar software almacenado en uno o más medios de almacenamiento legibles por ordenador no transitorios. También se debe entender, en el contexto de esta divulgación, un "codificador de vídeo" es un término genérico que se aplica tanto a un codificador de vídeo como a un decodificador de vídeo. Asimismo, el término "codificación de vídeo" se podría referir a la codificación de vídeo o bien a la descodificación de vídeo.

[0113] Como un primer proceso de derivación de NBDV de ejemplo de esta divulgación, el decodificador de vídeo 30 se configura para determinar si un bloque vecino espacial o temporal se codifica con el modo de BVSP. Esta comprobación se realiza para cada bloque vecino espacial o temporal definido para el proceso de derivación de NBDV. Si un bloque vecino se codificado con el modo de BVSP, el decodificador de vídeo 30 designa el vector de movimiento que pertenece al bloque vecino codificado con el modo de BVSP como un vector de movimiento de disparidad, independientemente de si el bloque vecino se ubica en la imagen actual o en una imagen diferente. En otras palabras, el decodificador de vídeo 30 designa ambos vectores de movimiento en bloques vecinos codificados con BVSP, y los vectores de movimiento en bloques vecinos codificados con predicción de entre vistas como vectores de movimiento de disparidad disponibles durante el proceso de derivación de NBDV.

[0114] En un segundo proceso de derivación de NBDV de ejemplo de esta divulgación, el proceso de derivación de NBDV a nivel de MB para la división de un MB usando las técnicas de la solicitud de patente de EE. UU. n.º 14/189,177 se puede mejorar comprobando el uso de modos de BVSP en bloques vecinos.

[0115] En un primer aspecto del segundo ejemplo, el decodificador de vídeo 30 se puede configurar para derivar un vector de disparidad usando derivación de NBDV que usa cada bloque vecino temporal y/o espacial, cuando cualquiera de las siguientes dos condiciones a continuación es verdadera: (1) si el bloque vecino se codifica usando la predicción de entre vistas, o (2) si el bloque vecino se codifica usando el modo de BVSP. En el caso de que el bloque vecino se codifique usando la predicción de entre vistas, el decodificador de vídeo 30 designa el vector de movimiento de disparidad asociado con el bloque vecino como el vector de disparidad para el bloque actual. Si el bloque vecino se codificó usando el modo de BVSP, el decodificador de vídeo 30 designa el vector de disparidad generado durante la descodificación del bloque vecino como el vector de disparidad para el bloque actual.

[0116] En un segundo aspecto del segundo ejemplo de la divulgación, el decodificador de vídeo 30 se puede emplear para un procedimiento de comprobación de dos etapas. Primero, el decodificador de vídeo 30 se configura para comprobar si al menos uno de los bloques vecinos espaciales y/o temporales se codifica usando la predicción entre vistas. Si no, el decodificador de vídeo 30 luego comprueba si al menos uno de los bloques vecinos espaciales y/o temporales se codifica usando el modo de BVSP.

[0117] En un tercer aspecto del segundo ejemplo de la divulgación, el proceso de comprobación de dos etapas descrito anteriormente se invierte. Es decir, el decodificador de vídeo 30 se configura para comprobar primero si al menos uno de los bloques vecinos se codifica usando el modo de BVSP. Si no, el decodificador de vídeo 30 se configura para comprobar luego si al menos uno de todos los bloques vecinos espaciales (y/o) temporales se codifican usando la predicción de entre vistas.

[0118] Cada uno de los aspectos descritos anteriormente del segundo ejemplo se puede aplicar a algoritmos de derivación de NBDV a nivel de división, así como algoritmos de derivación de NBDV a nivel de MB.

[0119] En un tercer proceso de derivación de NBDV de ejemplo de esta divulgación, el proceso de derivación de NBDV a nivel de MB se puede refinar adicionalmente añadiendo una etapa adicional al final del proceso que incluye el acceso a la vista de profundidad de referencia. En algunos ejemplos, el decodificador de vídeo 30 se puede configurar para

emplear esta etapa adicional a los bloques codificados usando el modo de BVSP, a los bloques interpredichos no codificados usando el modo de BVSP, o a todos los bloques interpredichos.

[0120] En un primer aspecto del tercer ejemplo, el decodificador de vídeo 30 se puede configurar para seleccionar un valor de profundidad a partir de un bloque de profundidad en la vista de referencia de profundidad, y convertir ese valor de profundidad en un vector de disparidad actualizado. Este vector de disparidad actualizado se aplica luego a todas las divisiones de MB dentro del MB actual. En algunos ejemplos, el decodificador de vídeo 30 se puede configurar además para almacenar este vector de disparidad actualizado/refinado como el vector de movimiento de disparidad final de la división de MB o MB actual.

[0121] En un cuarto proceso de derivación de NBDV de ejemplo de esta divulgación, el decodificador de vídeo 30 se puede configurar para almacenar el vector de disparidad derivado (es decir, derivado usando la derivación de NBDV) del MB como el vector de movimiento para el MB actual después de que se decodifica el MB actual .

[0122] En un quinto proceso de derivación de NBDV de ejemplo de esta divulgación, el decodificador de vídeo 30 se puede configurar para asignar memoria adicional para los MB que incluyen al menos una división de MB codificada usando el modo de BVSP. En este caso, los vectores de disparidad de los MB se pueden almacenar y no es necesario sobrescribir ningún vector de movimiento descodificado de las divisiones de MB.

[0123] En un primer aspecto del quinto ejemplo, un vector de movimiento adicional por MB se asigna para almacenar el vector de disparidad del MB derivado del proceso de derivación de NBDV en el caso de que al menos una división de MB se codifica usando el modo de BVSP.

[0124] En un segundo aspecto del quinto ejemplo, cuando se emplea el proceso de derivación de NBDV, el decodificador de vídeo 30 se puede configurar, además, para usar el vector de disparidad asociado con el MB que contiene este bloque vecino como un vector de disparidad para el MB actual cuando el bloque vecino se codifica usando el modo de BVSP.

[0125] En un sexto ejemplo, el proceso de derivación de NBDV de esta divulgación, el decodificador de vídeo 30 se puede configurar para emplear cualquier combinación de las técnicas de ejemplo anteriores para codificar herramientas que dependen de un vector de disparidad, tales como D-MVP de codificación en modos inter normales y predicción de vectores de movimiento entre vistas en modos de salto y directo. En este ejemplo, los resultados de un proceso de derivación de NBDV mejorado se usan para las herramientas de codificación. Por ejemplo, durante el proceso de D-MVP, el vector de disparidad resultante generado a partir del proceso de derivación de NBDV mejorado de esta divulgación se puede usar para reemplazar el resultado de otros procesos de derivación de NBDV (por ejemplo, el resultado del proceso de derivación de NBDV descrito en la solicitud de patente de EE. UU. n.º 14/189,177).

[0126] Se debe entender que cualquiera de los ejemplos anteriores, y los aspectos de los ejemplos, se pueden realizar juntos en cualquier combinación.

[0127] Un proceso de BVSP mejorado también se propone en esta divulgación usando el proceso de derivación de NBDV mejorado como se describe anteriormente. Sin embargo, debido al motivo de que el proceso de BVSP mejorado produce vectores de disparidad más exactos, los resultados de la derivación de NBDV también se podrían mejorar también (es decir, se puede mejorar la exactitud de los vectores de disparidad derivados).

[0128] En un primer proceso de BVSP de ejemplo de la divulgación, durante el proceso de BVSP, se combinan un proceso de derivación de NBDV a nivel de MB y un proceso de refinamiento del vector de disparidad a nivel de subregión. En primer lugar, el decodificador de vídeo 30 se puede configurar para derivar un vector de disparidad para cada MB. El decodificador de vídeo 30 se puede configurar para derivar un vector de disparidad usando el proceso de derivación de NBDV mejorado descrito anteriormente o con el proceso de NBDV como se describe en la solicitud de patente de EE. UU. n.º 14/189,177.

[0129] Seguidamente, el decodificador de vídeo 30 se puede configurar para refinar el vector de disparidad para cada subregión 8x8 de la división de MB o MB actual. El decodificador de vídeo 30 puede usar los vectores de disparidad refinados para la compensación de movimiento para cada subregión de la división de MB o MB actual codificado con el modo de BVSP. El refinamiento del vector de disparidad para cada subregión del MB o de la división de MB depende del componente de vista de profundidad de la vista de referencia. En un ejemplo, para cada subregión, se identifica un bloque de profundidad correspondiente en un componente de profundidad de la vista de referencia mediante el vector de disparidad del proceso de derivación de NBDV. El valor máximo de los cuatro píxeles de la esquina en el bloque de profundidad correspondiente se convierte en el componente horizontal de un vector de disparidad refinado. La componente vertical del vector de disparidad refinado se establece en 0. Téngase en cuenta que aquí el vector de disparidad refinado es el mismo que el vector de disparidad derivado para las subregiones codificadas con el modo de BVSP, como se analiza anteriormente.

[0130] En un primer aspecto del primer ejemplo de BVSP, el tamaño de subregión puede ser KxK, siendo K diferente de 8, por ejemplo, 16x16, 4x4, 2x2 o 1x1. En un segundo aspecto del primer ejemplo de BVSP, el decodificador de

vídeo 30 se configura para refinar el vector de disparidad una vez para una división de MB, o refinar el vector de disparidad para cada región 8x8 dentro de una división de MB, incluso cuando K es menor que 8.

[0131] En un tercer aspecto del primer ejemplo de BVSP, el decodificador de vídeo 30 se puede configurar para seleccionar el valor de profundidad a partir de uno o más píxeles de profundidad del bloque de referencia identificado por el vector de disparidad (producido por el proceso de derivación de NBDV) en el componente de vista de profundidad de la vista de referencia. En algunos ejemplos, el decodificador de vídeo 30 se puede configurar para no seleccionar píxeles de profundidad completa dentro o cerca del centro del bloque de profundidad identificado en la vista de profundidad de referencia (que se ubica en el vector de disparidad derivado del proceso de derivación de NBDV). En cambio, en un ejemplo, el decodificador de vídeo 30 se puede configurar para seleccionar píxeles de profundidad ubicados en las posiciones de las esquinas del bloque de profundidad identificado.

[0132] En un cuarto aspecto del primer ejemplo de BVSP, el decodificador de vídeo 30 se puede configurar para heredar, para la componente vertical del vector de disparidad refinado, la componente vertical del vector de disparidad derivado del proceso de derivación de NBDV. En un quinto aspecto del primer ejemplo de BVSP, el decodificador de vídeo 30 se puede configurar para establecer la componente vertical del vector de disparidad refinado para que sea igual a 0.

[0133] En un segundo proceso de BVSP de ejemplo de esta divulgación, el decodificador de vídeo 30 se configura para almacenar, para una división de MB codificada que usa el modo de BVSP, el vector de disparidad refinado de cada subregión como el vector de movimiento para la división de MB después de que se codifica el MB actual. En un ejemplo, este proceso se realiza en el caso de que el tamaño de la subregión de MB sea mayor que o igual a 4x4. Los vectores de disparidad almacenados se pueden usar en el proceso de derivación de NBDV de esta divulgación descrito anteriormente.

[0134] En un primer aspecto del segundo ejemplo de BVSP, cuando el tamaño definido de una subregión es 8x8 (es decir, cuando k es igual a 8), el decodificador de vídeo 30 se puede configurar para almacenar el vector de disparidad refinado como el vector de movimiento de una división de MB solamente cuando el tamaño de la división de MB es igual a 8x8. Para otras divisiones de MB, el decodificador de vídeo 30 se configura para almacenar el vector de disparidad derivado del MB a partir del proceso de derivación de NBDV como un vector de movimiento de disparidad después de que se codifica el MB actual.

[0135] En un segundo aspecto del segundo ejemplo de BVSP, el decodificador de vídeo 30 se puede configurar además para reemplazar el índice de imágenes de referencia por el índice que indica la imagen de referencia entre vistas.

[0136] En un tercer aspecto del segundo ejemplo de BVSP, cada una de las técnicas de BVSP descritas anteriormente se puede aplicar directamente una vez que se codifica una división de MB codificada con el modo de BVSP. Es decir, por ejemplo, el decodificador de vídeo 30 no necesita esperar hasta que todo el MB se decodifique totalmente.

[0137] En un tercer proceso de BVSP de ejemplo de esta divulgación, el decodificador de vídeo 30 se puede configurar para asignar memoria adicional para los MB que incluye al menos una división de MB que se codifica usando el modo de BVSP. En este caso, el vector de disparidad refinado de cada subregión usado para la compensación de movimiento de la división de BVSP se puede almacenar y no es necesario sobrescribirlo.

[0138] En un primer aspecto del tercer ejemplo de BVSP, el decodificador de vídeo 30 se puede configurar para asignar hasta (16/K)*(16/K) vectores de movimiento adicionales por MB para una división de MB para almacenar los vectores de disparidad refinados si la división se codifica usando el modo de BVSP.

[0139] En un segundo aspecto del tercer ejemplo de BVSP, en el proceso de derivación de NBDV, cuando un bloque vecino se codifica usando el modo de BVSP, el vector de disparidad refinado asociado a la subregión que contiene este bloque vecino se usará como un vector de disparidad para el MB actual.

[0140] Cada uno de los procedimientos anteriores también se puede aplicar a 3D-HEVC con los MB reemplazados por unidades de codificación (CU) y las divisiones de MB reemplazadas por unidades de predicción (PU).

1 implementación de ejemplo

[0141] La siguiente sección de la divulgación analizará implementaciones de ejemplo. Al configurar 3D-AVC de forma que se habilite el orden de codificación de textura primero, el proceso de descodificación de BVSP puede incluir las siguientes etapas. De forma alternativa, el procedimiento descrito en la sección 1.2 a continuación se puede aplicar a otras herramientas de codificación que requieren un vector de disparidad, por ejemplo, D-MVP en 3D-AVC.

Sección 1.1 Indicación de ejemplo del modo de BVSP

[0142] En un ejemplo, se puede reutilizar la indicación de que una o más divisiones de MB se codifican usando el modo de BVSP como se usa en las propuestas anteriores para 3D-AVC que usan el orden de codificación de profundidad primero. En otro ejemplo, se puede señalar un indicador, en lugar del índice de referencia, en cada división de MB para indicar si se usa BVSP o la interpredicción convencional (la predicción temporal o la predicción entre vistas).

Sección 1.2 Proceso de derivación del vector de disparidad de ejemplo

[0143] El proceso de derivación de NBDV se describe en la subsección 1.2.1 y 1.2.2 a continuación. El proceso de generación de vectores de disparidad refinados que se usan para la compensación de movimiento para cada subregión en divisiones codificadas de BVSP se describe en la subsección 1.2.3 a continuación.

[0144] El proceso de derivar un vector disparidad a partir del proceso de derivación de NBDV se puede invocar antes de la descodificación de un macrobloque. En otro ejemplo, además, el proceso de refinación de los vectores de disparidad se puede invocar para divisiones de MB codificadas con el modo de BVSP. En otro ejemplo, el proceso de derivación de un vector de disparidad a partir de NBDV se puede invocar cuando se codifica una división de MB con el modo de BVSP y/u otros modos inter.

[0145] En otro ejemplo, las técnicas de la subsección 1.2.1 y 1.2.3 a continuación se realizan en orden, y luego los vectores de disparidad refinados de subregiones generadas a partir de las técnicas de la subsección 1.2.3 se usan para la compensación de movimiento de divisiones de MB codificadas por BVSP. Además, los vectores de disparidad generados a partir de las técnicas de la subsección 1.2.1 a continuación se pueden aplicar en D-MVP.

[0146] En otro ejemplo, las técnicas de las subsecciones 1.2.1, 1.2.2 y 1.2.3 se invocan en orden. El vector de disparidad actualizado generado a partir de las técnicas de la sección 1.2.2 a continuación se usa luego para todas las divisiones de MB codificadas con BVSP o bien con otros modos inter. Para las divisiones de MB codificadas con el modo de BVSP, el vector de disparidad actualizado luego se usa para obtener los vectores de disparidad refinados de las subregiones que se usan para la compensación por movimiento de las divisiones de MB codificadas con BVSP.

Subsección 1.2.1 Vector de disparidad derivado de un proceso de derivación de NBDV para un macrobloque sin información de profundidad

[0147] Como se describe en la solicitud de patente de EE. UU. n.º 14/189,177, un proceso de derivación de NBDV a nivel de MB se puede aplicar sin considerar ninguna información de profundidad. Inicialmente, el decodificador de vídeo 30 se puede configurar para seleccionar un determinado número de candidatos de imagen temporal, y se comprueban varios bloques vecinos temporales predefinidos en la imagen candidata, seguidos de bloques vecinos espaciales predefinidos. Para cada bloque vecino, si se predice desde una vista diferente o bien un componente de referencia sintético en RefPicList0 (es decir, codificado usando el modo de BVSP), se devuelve el vector de movimiento de disparidad o el vector de disparidad derivado asociado/vector de disparidad refinado del bloque vecino como el vector de disparidad del MB actual si la componente horizontal del vector de disparidad derivado/vector de disparidad refinado no es igual a 0.

[0148] En otro ejemplo, el decodificador de vídeo 30 se puede configurar para comprobar tanto la lista de imágenes de referencia 0 (RefPicList0) como la lista de imágenes de referencia 1 (RefPicList1) en cualquier orden.

[0149] En otro ejemplo, el decodificador de vídeo 30 se puede configurar para comprobar todos los bloques vecinos espaciales (y/o) temporales, ya sea que usen o no la predicción de entre vistas. Si no, el decodificador de vídeo 30 comprueba si los bloques vecinos se codifican usando el modo de BVSP en una ronda diferente de comprobación de los bloques vecinos.

[0150] En otro ejemplo, el decodificador de vídeo 30 se configura para comprobar primero si los bloques vecinos se codifican usando el modo de BVSP en la primera ronda de comprobación de todos los bloques vecinos. Si no se codifica ningún bloque usando el modo de BVSP, el decodificador de vídeo 30 luego comprueba todos los bloques vecinos espaciales (y/o) temporales para determinar si se codifican usando la predicción entre vistas.

[0151] En otro ejemplo, cuando un bloque vecino no se codifica con la predicción entre vistas o bien el modo de BVSP, el decodificador de vídeo 30 devuelve el vector de disparidad del MB que contiene el bloque vecino, si está disponible, como el vector de disparidad. El vector de disparidad del MB está disponible solamente cuando al menos una de las divisiones de MB se codifica usando el modo de BVSP.

Subsección 1.2.1.1 Selección de bloques vecinos temporales/espaciales

[0152] Se pueden usar procedimientos similares como los que se describen en la solicitud de patente de EE. UU. n.º 14/189,177, y anteriormente, para seleccionar que bloques vecinos temporales/espaciales usar para la derivación de NBDV. Téngase en cuenta que un bloque vecino temporal en una imagen temporal también se puede predecir con BVSP, y el vector de disparidad de la división de MB o del MB codificado con BVSP también se considera disponible.

Subsección 1.2.2 Proceso de actualización del vector de disparidad a nivel de MB con información de profundidad

[0153] El decodificador de vídeo 30 se puede configurar para actualizar un vector de disparidad derivado que usa los procesos de derivación de NBDV descritos anteriormente usando las siguientes técnicas. Primero, se indica el tamaño de un MB como K x K (en el que K podría ser 16), la posición superior izquierda de la división de MB actual con respecto a la imagen actual como (x, y), y el vector de disparidad derivado a partir de NBDV en la subsección 1.2.1 para el MB actual como (DV[0], DV[1]), en el que DV[0] y DV[1] indican la componente horizontal y vertical del vector de disparidad. Un valor de profundidad (D) se selecciona entre cuatro píxeles de esquina en la vista de profundidad de referencia:

$$D = \text{máx}\ (D0, D1, D2, D3)$$

[0154] La función máx(.) devuelve el valor máximo de Di (siendo i de 0 a 3) y Di indica el iésimo valor de píxel que se ubica en:

i = 0: ((x + (DV[0] >> P)) >> reduced_resolution_flag, (y + (DV[1] >> P)) >> reduced_resolution_flag)

i = 1: ((x + (DV[0] >> P) + K-1) >> reduced_resolution_flag, (y + (DV[1] >> P)) >> reduced_resolution_flag)

i = 2: ((x + (DV[0] >> P)) >> reduced_resolution_flag, (y + (DV[1] >> P) + K-1) >> reduced_resolution_flag)

i = 3: ((x + (DV[0] >> P) + K-1) >> reduced_resolution_flag, (y + (DV[1] >> P) + K-1) >> reduced_resolution_flag)

[0155] El elemento sintáctico reduced_resolution_flag igual a 1 especifica que los componentes de vista de profundidad de un par de componentes de vista tienen una resolución espacial más baja que el componente de luma del componente de vista de textura del mismo par de componentes de vista, y el ancho y la altura de los componentes de vista de profundidad son ambos la mitad del ancho y la altura de todos los componentes de vista de textura. El elemento sintáctico reduced_resolution_flag igual a 0 especifica que cuando están presentes tanto los componentes de vista de profundidad como los componentes de vista de textura, tienen la misma resolución espacial. P indica la precisión de los vectores de disparidad que es igual a 2 cuando el vector de disparidad tiene una precisión de cuarto de píxel, 1 para la precisión de medio píxel y 0 para la precisión de píxel entero.

[0156] En otro ejemplo, máx(D0, D3) se puede usar para seleccionar el valor de profundidad.

[0157] En otro ejemplo, se pueden usar otros píxeles dentro del MB coubicado en la vista de profundidad de referencia.

[0158] El decodificador de vídeo 30 se puede configurar luego para convertir el componente horizontal del vector de disparidad actualizado a partir del valor de profundidad seleccionado para la subregión dentro de la región de MB actual. El componente vertical del vector de disparidad actualizado se establece en 0.

[0159] En otro ejemplo, la componente vertical del vector de disparidad actualizado se puede establecer en la componente vertical del vector de disparidad derivado de NBDV.

[0160] En otro ejemplo, K puede ser igual a 8, 4 o 2.

[0161] El vector de disparidad actualizado se puede usar para todas las divisiones dentro del MB actual.

[0162] Sección 1.2.3 Refinamiento del vector de disparidad para cada división de MB codificado con el modo de BVSP

[0163] El decodificador de vídeo 30 también se puede configurar para derivar un vector de disparidad refinado para cada subregión de cada división de MB, en el caso de que la división de MB se codifique con el modo de BVSP (es decir, predicho a partir de un componente de referencia sintético).

[0164] El decodificador de vídeo 30 se puede configurar para refinar un vector de disparidad derivado que usa los procesos de derivación de NBDV descritos anteriormente usando las siguientes técnicas. Primero, se indica el tamaño de la subregión como K x K (en la que K podría ser 8), la posición superior izquierda de una subregión dentro de la división de MB actual con respecto a la imagen actual como (x, y), y el vector de disparidad derivado a partir del proceso de derivación de NBDV (o el vector de disparidad actualizado generado después de realizar las técnicas de

la subsección 1.2.2) para MB actuales como (DV[0], DV[1]), en los que DV[0] y DV[1] indican la componente horizontal y vertical del vector de disparidad. Un valor de profundidad (D) se selecciona entre cuatro píxeles de esquina en la vista de profundidad de referencia:

$$D = \text{máx}\ (D0, D1, D2, D3)$$

[0165] La función máx(•) devuelve el valor máximo de Di (siendo i de 0 a 3) y Di indica el valor del píxel iésimo que se ubica en:

i = 0: ((x + (DV[0] >> P)) >> reduced_resolution_flag, (y + (DV[1] >> P)) >> reduced_resolution_flag)

i = 1: ((x + (DV[0] >> P) + K-1) >> reduced_resolution_flag, (y + (DV[1] >> P)) >> reduced_resolution_flag)

i = 2: ((x + (DV[0] >> P)) >> reduced_resolution_flag, (y + (DV[1] >> P) + K-1) >> reduced_resolution_flag)

i = 3: ((x + (DV[0] >> P) + K-1) >> reduced_resolution_flag, (y + (DV[1] >> P) + K-1) >> reduced_resolution_flag)

[0166] Para esta subsección, el elemento sintáctico reduced_resolution_flag y P se definen como en la sección 1.2.2.

[0167] En otro ejemplo, máx(D0, D3) se puede usar para seleccionar el valor de profundidad.

[0168] En otro ejemplo, se pueden usar otros píxeles dentro del MB coubicado en la vista de profundidad de referencia.

[0169] El decodificador de vídeo 30 se puede configurar además para convertir el componente horizontal del vector de disparidad refinado a partir del valor de profundidad seleccionado para la subregión dentro de la región de MB actual. La componente vertical del vector de disparidad refinado se establece en 0. En otro ejemplo, la componente vertical del vector de disparidad refinado se puede establecer en la componente vertical del vector de disparidad derivado a partir de NBDV.

[0170] En otro ejemplo, K puede ser igual a 4, 2 o 1.

Sección 1.3 Predicción de una división de MB codificado con el modo de BVSP

[0171] Para cada subregión dentro de la división de MB actual codificado con el modo de BVSP, el decodificador de vídeo 30 se puede configurar para usar el vector de disparidad refinado para obtener el bloque de predicción en la vista de textura de referencia. En otro ejemplo, para cada división de MB codificado con el modo de BVSP, el decodificador de vídeo 30 se puede configurar para usar el vector de disparidad derivado a partir del proceso de derivación de NBDV para obtener el bloque de predicción en la vista de textura de referencia. El bloque residual de la división de MB actual y los bloques de predicción juntos se usan para reconstruir la división de MB.

1.4 Asignación de vectores de movimiento para divisiones de MB codificados con el modo de BVSP

[0172] Después de que un MB en la vista de textura no básica se decodifica totalmente, el decodificador de vídeo 30 puede almacenar los vectores de disparidad refinados, como se describe en la subsección 1.2.3, se almacenan como los vectores de movimiento para cada subregión de una división de MB que se codifica usando el modo de BVSP.

[0173] En otro ejemplo, después de que se haya decodificado totalmente un MB en la vista de textura no básica, el decodificador de vídeo 30 puede almacenar el vector de disparidad derivado a partir del proceso de derivación de NBDV como se describe en la subsección 1.2.1 o el vector de disparidad actualizado con la información de profundidad tomada en consideración como se describe en la subsección 1.2.2 como el vector de movimiento para todas las divisiones de MB que se codifican usando el modo de BVSP.

[0174] En otro ejemplo, después de que se descodifica una división de MB codificado con el modo de BVSP en la vista de textura no básica, el decodificador de vídeo 30 puede almacenar directamente el vector de disparidad derivado a partir del proceso de derivación de NBDV a nivel de MB como se describe en la subsección 1.2.1 o el vector de disparidad actualizado con información de profundidad tomada en consideración como se describe en la subsección 1.2.2 como el vector de movimiento para esta división de MB.

[0175] En otro ejemplo, después de que se descodifica una división de MB codificado con el modo de BVSP en la vista de textura no básica, el decodificador de vídeo 30 puede almacenar directamente los vectores de disparidad

refinados para cada subregión, como se describe en la subsección 1.2.3, como el vector de movimiento para esta subregión.

[0176] En otro ejemplo, en lugar de almacenar el vector de disparidad derivado a partir del proceso de derivación de NBDV (con posible refinamiento), como se describe en la subsección 1.2.2, o almacenar los vectores de disparidad refinados como los vectores de movimiento para divisiones codificadas de BVSP, el decodificador de vídeo 30 puede asignar memoria adicional para almacenar esta información.

[0177] En otro ejemplo, el decodificador de vídeo 30 asigna un vector de movimiento por MB para almacenar el vector de disparidad derivado a partir del proceso de derivación de NBDV (con posible refinamiento) como se describe en la subsección 1.2.2 si el MB actual tiene al menos una división codificada con el modo de BVSP. Durante el proceso de derivación de NBDV, cuando un bloque vecino usa el modo de BVSP, el vector de disparidad asociado con el MB que contiene este bloque vecino se usará como un vector de disparidad para el MB actual.

[0178] En otro ejemplo, cuatro vectores de movimiento se asignan por MB para almacenar los vectores de disparidad refinados para cada subregión cuando el tamaño de la subregión es igual a 8x8. Durante el proceso de derivación de NBDV, cuando un bloque vecino usa el modo de BVSP, el decodificador de vídeo 30 usa el vector de disparidad asociado con la subregión que contiene este bloque vecino como un vector de disparidad para el MB actual.

[0179] La FIG. 6 es un diagrama de bloques que ilustra un ejemplo de un codificador de vídeo que puede implementar las técnicas descritas en esta divulgación. Por ejemplo, la Fig. 6 ilustra el codificador de vídeo 20 que puede realizar la intra e intercodificación de bloques de vídeo dentro de segmentos de vídeo. Por ejemplo, el codificador de vídeo 20 puede realizar codificación de interpredicción o codificación de intrapredicción. La intracodificación se basa en la predicción espacial para reducir o eliminar la redundancia espacial en el vídeo dentro de una trama o imagen de vídeo dada. La intercodificación se basa en la predicción temporal o en la predicción entre vistas para reducir o eliminar la redundancia temporal dentro de tramas o imágenes adyacentes de una secuencia de vídeo o la redundancia entre imágenes en diferentes vistas. El modo intra (modo I) puede referirse a cualquiera de varios modos de compresión espacial. Los modos inter, tales como la predicción unidireccional (modo P) o la predicción bidireccional (modo B), pueden referirse a cualquiera de varios modos de compresión temporal.

[0180] En el ejemplo de la FIG. 6, el codificador de vídeo 20 incluye una memoria de datos de vídeo 40, una unidad de procesamiento de predicción 42, una memoria de imagen de referencia 64, un sumador 50, una unidad de procesamiento de transformada 52, una unidad de procesamiento de cuantificación 54 y una unidad de codificación de entropía 56. La unidad de procesamiento de predicción 42 incluye una unidad de estimación de movimiento 44, una unidad de compensación de movimiento 46 y una unidad de intrapredicción 48. Para la reconstrucción de bloques de vídeo, el codificador de vídeo 20 también incluye una unidad de procesamiento de cuantificación inversa 58, una unidad de procesamiento de transformada inversa 60 y un sumador 62. También se puede incluir un filtro de eliminación de bloques (no mostrado en la FIG. 6) para filtrar fronteras de bloques para eliminar distorsiones de efecto pixelado del vídeo reconstruido. Si se desea, el filtro de eliminación de bloques filtrará típicamente la salida del sumador 62. También se pueden usar filtros de bucle adicionales (en bucle o tras el bucle), además del filtro de eliminación de bloques.

[0181] La memoria de datos de vídeo 40 puede almacenar datos de vídeo que se van a codificar por los componentes del codificador de vídeo 20. Los datos de vídeo almacenados en la memoria de datos de vídeo 40 se pueden obtener, por ejemplo, a partir de la fuente de vídeo 18. La memoria de imágenes de referencia 64 es un ejemplo de una memoria intermedia de imágenes de descodificación (DPB que almacena datos de vídeo de referencia para su uso en la codificación de datos de vídeo por el codificador de vídeo 20 (por ejemplo, en los modos de intra o intercodificación, también denominados modos de codificación de intra o interpredicción). La memoria de datos de vídeo 40 y la memoria imágenes de referencia 64 se pueden formar por cualquiera de una variedad de dispositivos de memoria, tales como memoria dinámica de acceso aleatorio (DRAM), incluyendo DRAM síncrona (SDRAM), RAM magnetorresistiva (MRAM), RAM resistiva (RRAM) u otros tipos de dispositivos de memoria. La memoria de datos de vídeo 40 y la memoria de imágenes de referencia 64 se pueden proporcionar por el mismo dispositivo de memoria o por dispositivos de memoria separados. En diversos ejemplos, la memoria de datos de vídeo 40 puede estar en un chip con otros componentes del codificador de vídeo 20, o fuera de chip con respecto a esos componentes.

[0182] El codificador de vídeo 20 recibe datos de vídeo, y una unidad de división (no mostrada) divide los datos en bloques de vídeo. Esta división también puede incluir la división en segmentos, mosaicos u otras unidades más grandes, así como la división de bloques de vídeo (por ejemplo, divisiones de macrobloques y subbloques de divisiones). El codificador de vídeo 20 ilustra, en general, los componentes que codifican bloques de vídeo dentro de un segmento de vídeo que se va a codificar. El segmento se puede dividir en múltiples bloques de vídeo (y, posiblemente, en conjuntos de bloques de vídeo denominados mosaicos). La unidad de procesamiento de predicción 42 puede seleccionar uno de una pluralidad de posibles modos de codificación, tal como uno de una pluralidad de modos de intracodificación (modos de codificación de intrapredicción) o uno de una pluralidad de modos de intercodificación (modos de codificación de interpredicción), para el bloque de vídeo actual en base a los resultados del error (por ejemplo, la tasa de codificación y el nivel de distorsión). La unidad de procesamiento de predicción 42

puede proporcionar el bloque con intra o intercodificación resultante al sumador 50 para generar datos de bloques residuales, y al sumador 62 para reconstruir el bloque codificado para su uso como una imagen de referencia.

[0183] La unidad de intrapredicción 48, dentro de la unidad de procesamiento de predicción 42, puede realizar la codificación intrapredictiva del bloque de vídeo actual con respecto a uno o más bloques vecinos en la misma trama o segmento que el bloque que se va a codificar, para proporcionar compresión espacial. La unidad de estimación del movimiento 44 y la unidad de compensación del movimiento 46 dentro de la unidad de procesamiento de predicción 42 realizan la codificación interpredictiva del bloque de vídeo actual con respecto a uno o más bloques predictivos de una o más imágenes de referencia para proporcionar una compresión temporal.

[0184] La unidad de estimación del movimiento 44 se puede configurar para determinar el modo de interpredicción para un segmento de vídeo de acuerdo con un patrón predeterminado para una secuencia de vídeo. El patrón predeterminado puede designar segmentos de vídeo de la secuencia como segmentos P o segmentos B. La unidad de estimación del movimiento 44 y la unidad de compensación del movimiento 46 pueden estar sumamente integradas, pero se ilustran por separado para propósitos conceptuales. La estimación del movimiento, realizada por la unidad de estimación del movimiento 44, es el proceso de generación de vectores de movimiento, que estiman el movimiento de los bloques de vídeo. Un vector de movimiento, por ejemplo, puede indicar el desplazamiento de un bloque de vídeo dentro de una trama o imagen de vídeo actual a un bloque predictivo dentro de una imagen de referencia.

[0185] Un bloque predictivo es un bloque que se encuentra que coincide estrechamente con el bloque que se va a codificar, en lo que respecta a la diferencia de píxeles, lo cual se puede determinar mediante la suma de la diferencia absoluta (SAD), suma de la diferencia de cuadrados (SSD) u otras métricas de diferencia. 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 64. 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 del movimiento 44 puede realizar una búsqueda de movimiento con respecto a las posiciones de píxeles completas y las posiciones de píxeles fraccionarias, y generar un vector de movimiento con una precisión de píxel fraccionaria.

[0186] La unidad de estimación del movimiento 44 calcula un vector de movimiento para un bloque de vídeo en un segmento intercodificado (codificado por interpredicción) comparando la posición del bloque de vídeo con la posición de un bloque predictivo de una imagen de referencia. La imagen de referencia se puede seleccionar entre una primera lista de imágenes de referencia (RefPicList0) o una segunda lista de imágenes de referencia (RefPicList1), cada una de las cuales identifica una o más imágenes de referencia almacenadas en la memoria de imágenes de referencia 64. La unidad de estimación del movimiento 44 envía el vector de movimiento calculado a la unidad de codificación por entropía 56 y a la unidad de compensación del movimiento 46.

[0187] La compensación del movimiento, realizada por la unidad de compensación del movimiento 46, puede implicar obtener o generar el bloque predictivo en base al vector de movimiento determinado mediante estimación de movimiento, posiblemente realizando interpolaciones hasta la precisión de subpíxel. Tras recibir el vector de movimiento para el bloque de vídeo actual, la unidad de compensación del movimiento 46 puede ubicar el bloque predictivo al que apunta el vector de movimiento en una de las listas de imágenes de referencia. El codificador de vídeo 20 forma un bloque de vídeo residual restando los valores de píxel del bloque predictivo a los valores de píxel del bloque de vídeo actual que se está codificando, formando valores de diferencia de píxel. Los valores de diferencia de píxel forman datos residuales para el bloque, y pueden incluir componentes de diferencia tanto de luma como de croma. El sumador 50 representa el componente o los componentes que realizan esta operación de resta. La unidad de compensación del movimiento 46 también puede generar elementos sintácticos asociados a los bloques de vídeo y al segmento de vídeo para su uso por el decodificador de vídeo 30 en la descodificación de los bloques de vídeo del segmento de vídeo.

[0188] La unidad de intrapredicción 48 puede intrapredecir un bloque actual, como alternativa a la interpredicción realizada por la unidad de estimación del movimiento 44 y la unidad de compensación del movimiento 46, como se describe anteriormente. En particular, la unidad de intrapredicción 48 puede determinar un modo de intrapredicción para usar para codificar un bloque actual. En algunos ejemplos, la unidad de intrapredicción 48 puede codificar un bloque actual usando diversos modos de intrapredicción, por ejemplo, durante pasadas de codificación separadas, y la unidad de intrapredicción 48 (o una unidad de selección de modo, en algunos ejemplos) puede seleccionar un modo de intrapredicción apropiado para usar a partir de los modos probados. Por ejemplo, la unidad de intrapredicción 48 puede calcular valores de velocidad-distorsión usando un análisis de velocidad-distorsión para los diversos modos de intrapredicción probados, y seleccionar el modo de intrapredicción que tenga las mejores características de velocidad-distorsión entre los modos probados. El análisis de velocidad-distorsión determina, en general, una cantidad de distorsión (o error) entre un bloque codificado y un bloque original no codificado que se codificó para producir el bloque codificado, así como una velocidad de bits (es decir, un número de bits) usada para producir el bloque codificado. La unidad de intrapredicción 48 puede calcular proporciones a partir de las distorsiones y velocidades para los diversos bloques codificados para determinar qué modo de intrapredicción presenta el mejor valor de velocidad-distorsión para el bloque.

[0189] En cualquier caso, después de seleccionar un modo de intrapredicción para un bloque, la unidad de intrapredicción 48 puede proporcionar información indicativa del modo de intrapredicción seleccionado para el bloque a la unidad de codificación por entropía 56. La unidad de codificación por entropía 56 puede codificar la información que indica el modo de intrapredicción seleccionado de acuerdo con las técnicas de esta divulgación. El codificador de vídeo 20 puede incluir datos de configuración en el flujo de bits transmitido, que pueden incluir una pluralidad de tablas de índices de modos de intrapredicción y una pluralidad de tablas de índices de modos de intrapredicción modificadas (también denominadas tablas de correlación de palabras de código), definiciones de contextos de codificación para diversos bloques e indicaciones de un modo de intrapredicción más probable, una tabla de índices de modos de intrapredicción y una tabla de índices de modos de intrapredicción modificadas para usar para cada uno de los contextos.

[0190] Después de que la unidad de procesamiento de predicción 42 genera el bloque predictivo para el bloque de vídeo actual, por medio de interpredicción o bien intrapredicción, el codificador de vídeo 20 forma un bloque de vídeo residual restando el bloque predictivo al bloque de vídeo actual. Los datos de vídeo residual en el bloque residual se pueden aplicar a la unidad de procesamiento de transformada 52. La unidad de procesamiento de transformada 52 transforma los datos de vídeo residuales en coeficientes de transformada residuales usando una transformada, tal como una transformada de coseno discreta (DCT) o una transformada conceptualmente similar. La unidad de procesamiento de transformada 52 puede convertir los datos de vídeo residuales de un dominio de píxeles a un dominio de transformada, tal como un dominio de frecuencia.

[0191] La unidad de procesamiento de transformada 52 puede enviar los coeficientes de transformada resultantes a la unidad de procesamiento de cuantificación 54. La unidad de procesamiento de cuantificación 54 cuantifica los coeficientes de transformada para reducir adicionalmente la velocidad de bits. El proceso de cuantificación puede reducir la profundidad de bits asociada a algunos, o a la totalidad, de los coeficientes. El grado de cuantificación se puede modificar ajustando un parámetro de cuantificación. En algunos ejemplos, la unidad de procesamiento de cuantificación 54 luego puede realizar un escaneado de la matriz, incluyendo los coeficientes de transformada cuantificados. De forma alternativa, la unidad de codificación por entropía 56 puede realizar el escaneado.

[0192] Tras la cuantificación, la unidad de codificación por entropía 56 codifica por entropía los coeficientes de transformada cuantificados. Por ejemplo, la unidad de codificación por entropía 56 puede realizar una codificación de longitud variable adaptable al contexto (CAVLC), una codificación aritmética binaria adaptable al contexto (CABAC), una codificación aritmética binaria adaptable al contexto basada en la sintaxis (SBAC), una codificación por entropía por división de intervalos de probabilidad (PIPE) u otros procedimientos o técnicas de codificación por entropía. Tras la codificación por entropía por la unidad de codificación por entropía 56, el flujo de bits codificado se puede transmitir al decodificador de vídeo 30, o archivarse para su posterior transmisión o recuperación por el decodificador de vídeo 30. La unidad de codificación por entropía 56 también puede codificar por entropía los vectores de movimiento y los otros elementos sintácticos para el segmento de vídeo actual que se está codificando.

[0193] La unidad de cuantificación inversa 58 y la unidad de procesamiento de transformada inversa 60 aplican una cuantificación inversa y una transformación inversa, respectivamente, para reconstruir el bloque residual en el dominio de los píxeles, para su posterior uso como bloque de referencia de una imagen de referencia. La unidad de compensación del movimiento 46 puede calcular un bloque de referencia añadiendo el bloque residual a un bloque predictivo de una de las imágenes de referencia dentro de una de las listas de imágenes de referencia. La unidad de compensación del movimiento 46 también puede aplicar uno o más filtros de interpolación al bloque residual reconstruido para calcular valores de píxeles de subentero para su uso en la estimación de movimiento. El sumador 62 añade el bloque residual reconstruido al bloque de predicción compensado por movimiento producido por la unidad de compensación del movimiento 46 para producir un bloque de referencia para su almacenamiento en la memoria de imágenes de referencia 64. La unidad de estimación del movimiento 44 y la unidad de compensación del movimiento 46 pueden usar el bloque de referencia como un bloque de referencia para interpredecir un bloque en una imagen o trama de vídeo subsiguiente.

[0194] De esta manera, el codificador de vídeo 20 es un ejemplo de un codificador de vídeo que se configura para implementar una o más técnicas de ejemplo descritas en esta divulgación. Por ejemplo, la memoria de datos de vídeo 40 almacena datos de vídeo. Los datos de vídeo pueden incluir un componente de vídeo de textura de una vista dependiente y un componente de vista de profundidad que corresponde al componente de vista de textura, cada uno de los cuales debe codificar el codificador de vídeo 20 en un proceso de codificación de vídeo compatible con 3D-AVC.

[0195] En las técnicas descritas en esta divulgación, el codificador de vídeo 20 puede incluir uno o más procesadores que se configuran para codificar, en un proceso de codificación de vídeo compatible con 3D-AVC, un componente de vista de textura de una vista dependiente de los datos de vídeo. Como se describe anteriormente, cada vista en un 3D-AVC incluye un componente de vista de textura y un componente de vista de profundidad. Existe una vista básica y una o más vistas de potenciación o dependientes en 3D-AVC, donde los componentes de vista de textura de la una o más vistas de potenciación o dependientes se pueden predecir entre vistas.

[0196] Para codificar el componente de vista de textura, el codificador de vídeo 20 se puede configurar para evaluar la información de movimiento de uno o más bloques vecinos de un bloque actual en el componente de vista de textura para determinar si al menos un bloque vecino se predice entre vistas con un vector de movimiento de disparidad que se refiere a una imagen de referencia entre vistas en una vista distinta de la vista dependiente. El codificador de vídeo 20 puede derivar un vector de disparidad para el bloque actual en base al vector de movimiento de disparidad para uno de los bloques vecinos. Para la codificación de textura primero, el codificador de vídeo 20 puede codificar un componente de vista de profundidad, de los datos de vídeo, que corresponde al componente de vista de textura subsiguiente a la codificación del componente de vista de textura.

[0197] En algunos ejemplos, la unidad de procesamiento de predicción 42 del codificador de vídeo 20 puede ser un ejemplo de un procesador configurado para implementar los ejemplos descritos en esta divulgación para la derivación de NBDV y la codificación de BVSP. En algunos ejemplos, una unidad (por ejemplo, uno o más procesadores) distintos de la unidad de procesamiento de predicción 42 puede implementar los ejemplos descritos anteriormente. En algunos ejemplos, la unidad de procesamiento de predicción 42 en conjunto con una o más de otras unidades del codificador de vídeo 20 puede implementar los ejemplos descritos anteriormente. En algunos ejemplos, un procesador de codificador de vídeo 20 (no mostrado en la FIG. 6) puede, solo o en conjunto con otros procesadores de codificador de vídeo 20, implementar los ejemplos descritos anteriormente.

[0198] La FIG. 7 es un diagrama de bloques que ilustra un ejemplo de un decodificador de vídeo que puede implementar las técnicas descritas en esta divulgación. Por ejemplo, el decodificador de vídeo 30 puede realizar una descodificación interpredicción o una descodificación intrapredicción. La FIG. 7 ilustra el decodificador de vídeo 30. En el ejemplo de la FIG. 7, el decodificador de vídeo 30 incluye una memoria de datos de vídeo 69, una unidad de descodificación por entropía 70, una unidad de procesamiento de predicción 71, una unidad de procesamiento de cuantificación inversa 76, una unidad de procesamiento de transformada inversa 78, un sumador 80 y la memoria de imágenes de referencia 82. La unidad de procesamiento de predicción 71 incluye una unidad de compensación del movimiento 72 y una unidad de intrapredicción 74. En algunos ejemplos, el decodificador de vídeo 30 puede realizar una pasada de descodificación que en general es recíproca a la pasada de codificación descrita con respecto al codificador de vídeo 20 de la FIG. 6.

[0199] La memoria de datos de vídeo 69 puede almacenar datos de vídeo, tales como un flujo de bits de vídeo codificado, que se va a decodificar por los componentes del decodificador de vídeo 30. Los datos de vídeo almacenados en la memoria de datos de vídeo 69 se pueden obtener, por ejemplo, a partir de un dispositivo de almacenamiento 34, a partir de una fuente de vídeo local, tal como una cámara, por medio de comunicación de red alámbrica o inalámbrica de datos de vídeo, o accediendo a medios físicos de almacenamiento de datos. La memoria de datos de vídeo 69 puede formar una memoria intermedia de imágenes codificadas (CPB) que almacene datos de vídeo codificados a partir de un flujo de bits de vídeo codificado.

[0200] La memoria de imágenes de referencia 82 es un ejemplo de una memoria intermedia de imágenes decodificadas (DPB) que almacena datos de vídeo de referencia para su uso en la descodificación de datos de vídeo por el decodificador de vídeo 30 (por ejemplo, en los modos de intra o intercodificación). La memoria de datos de vídeo 69 y la memoria imágenes de referencia 82 se pueden formar por cualquiera de una variedad de dispositivos de memoria, tales como memoria dinámica de acceso aleatorio (DRAM), incluyendo DRAM síncrona (SDRAM), RAM magnetorresistiva (MRAM), RAM resistiva (RRAM) u otros tipos de dispositivos de memoria. La memoria de datos de vídeo 69 y la memoria de imágenes de referencia 82 se pueden proporcionar por el mismo dispositivo de memoria o por dispositivos de memoria separados. En diversos ejemplos, la memoria de datos de vídeo 69 puede estar en un chip con otros componentes del decodificador de vídeo 30, o fuera de chip con respecto a esos componentes.

[0201] Durante el proceso de descodificación, el decodificador de vídeo 30 recibe un flujo de bits de vídeo codificado, que representa los bloques de vídeo de un segmento de vídeo codificado y elementos sintácticos asociados, desde el codificador de vídeo 20. La unidad de descodificación por entropía 70 del decodificador de vídeo 30 descodifica por entropía el flujo de bits para generar coeficientes cuantificados, vectores de movimiento y otros elementos sintácticos. La unidad de descodificación por entropía 70 remite los vectores de movimiento y otros elementos sintácticos a la unidad de procesamiento de predicción 71. El decodificador de vídeo 30 puede recibir los elementos sintácticos en el nivel del segmento de vídeo y/o el nivel del bloque de vídeo.

[0202] Cuando el segmento de vídeo se codifica como un segmento intracodificado (I), la unidad de intrapredicción 74 de la unidad de procesamiento de predicción 71 puede generar datos de predicción para un bloque de vídeo del segmento de vídeo actual, en base a un modo de intrapredicción señalizada y datos de los bloques previamente decodificados de la trama o imagen actual. Cuando la trama de vídeo se codifica como un segmento intercodificado (es decir, B o P), la unidad de compensación de movimiento 72 de la unidad de procesamiento de predicción 71 produce bloques predictivos para un bloque de vídeo del segmento de vídeo actual, en base a los vectores de movimiento y otros elementos sintácticos recibidos desde la unidad de descodificación por entropía 70. Los bloques predictivos se pueden producir a partir de una de las imágenes de referencia dentro de una de las listas de imágenes de referencia. El decodificador de vídeo 30 puede construir las listas de imágenes de referencia (RefPicList0 y RefPicList1) usando técnicas de construcción predeterminadas en base a imágenes de referencia almacenadas en la memoria de imágenes de referencia 82.

[0203] La unidad de compensación de movimiento 72 determina la información de predicción para un bloque de vídeo del segmento de vídeo actual, analizando los vectores de movimiento y otros elementos sintácticos, y usa la información de predicción para producir los bloques predictivos del bloque de vídeo actual que se está descodificando. Por ejemplo, la unidad de compensación de movimiento 72 usa algunos de los elementos sintácticos recibidos para determinar un modo de predicción (por ejemplo, intra o interpredicción), usado para codificar los bloques de vídeo del segmento de vídeo, un tipo de segmento de interpredicción (por ejemplo, segmento B o segmento P), información de construcción para una o más de las listas de imágenes de referencia del segmento, vectores de movimiento para cada bloque de vídeo intercodificado del segmento, el estado de interpredicción para cada bloque de vídeo intercodificado del segmento y otra información, para decodificar los bloques de vídeo en el segmento de vídeo actual.

[0204] La unidad de compensación de movimiento 72 también puede realizar la interpolación en base a filtros de interpolación. La unidad de compensación de movimiento 72 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 este caso, la unidad de compensación de movimiento 72 puede determinar los filtros de interpolación usados por el codificador de vídeo 20 a partir de los elementos sintácticos recibidos y usar los filtros de interpolación para producir bloques predictivos.

[0205] La unidad de cuantificación inversa 76 cuantifica de manera inversa (es decir, descuantifica), los coeficientes de transformada cuantificados, proporcionados en el flujo de bits y descodificados por la unidad de descodificación por entropía 70. 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 del segmento de vídeo para determinar un grado de cuantificación y, asimismo, un grado de cuantificación inversa que se debería aplicar. La unidad de procesamiento de transformada inversa 78 aplica una transformada inversa (por ejemplo, una DCT inversa, una transformada inversa entera o un proceso de transformada inversa conceptualmente similar), a los coeficientes de transformada, a fin de producir bloques residuales en el dominio de píxeles.

[0206] Después de que la unidad de compensación de movimiento 72 genera el bloque predictivo para el bloque de vídeo actual, en base a los vectores de movimiento y otros elementos sintácticos, el decodificador de vídeo 30 forma un bloque de vídeo decodificado sumando los bloques residuales de la unidad de procesamiento de transformada inversa 78 a los correspondientes bloques predictivos generados por la unidad de compensación de movimiento 72. El sumador 80 representa el componente o los componentes que realizan esta operación de suma. Si se desea, también se puede aplicar un filtro de eliminación de bloques para filtrar los bloques descodificados a fin de eliminar distorsiones de efecto pixelado. También se pueden usar otros filtros de bucle (en el bucle de codificación o bien después del bucle de codificación) para allanar las transiciones de píxeles o mejorar de otro modo la calidad del vídeo. Los bloques de vídeo decodificados en una imagen dada se almacenan luego en la memoria de imágenes de referencia 82, que almacena imágenes de referencia usadas para la subsiguiente compensación de movimiento. La memoria de imágenes de referencia 82 almacena también vídeo descodificado para su presentación posterior en un dispositivo de visualización, tal como el dispositivo de visualización 32 de la FIG. 1.

[0207] De esta manera, el decodificador de vídeo 30 es un ejemplo de un decodificador de vídeo que se configura para implementar una o más técnicas de ejemplo descritas en esta divulgación. Por ejemplo, la memoria de datos de vídeo 69 almacena datos de vídeo. Los datos de vídeo pueden incluir información a partir de la cual el decodificador de vídeo 30 puede decodificar un componente de vídeo de textura de una vista dependiente y un componente de vista de profundidad que corresponde al componente de vista de textura, cada uno de cuyos codificadores de vídeo 20 se codifica en un proceso de codificación de vídeo compatible con AVC 3D.

[0208] En las técnicas descritas en esta divulgación, el decodificador de vídeo 30 puede incluir uno o más procesadores que se configuran para decodificar, en un proceso de codificación de vídeo compatible con 3D-AVC, un componente de vista de textura de una vista dependiente de los datos de vídeo. Para decodificar el componente de vista de textura, el decodificador de vídeo 30 se puede configurar para evaluar la información de movimiento de uno o más bloques vecinos de un bloque actual en el componente de vista de textura para determinar si al menos un bloque vecino se predice entre vistas con un vector de movimiento de disparidad que se refiere a una imagen de referencia entre vistas en una vista distinta de la vista dependiente. El decodificador de vídeo 30 puede derivar un vector de disparidad para el bloque actual en base al vector de movimiento de disparidad para uno de los bloques vecinos. Para la codificación de textura primero, el decodificador de vídeo 30 puede decodificar un componente de vista de profundidad, de los datos de vídeo, que corresponde al componente de vista de textura subsiguiente a la descodificación del componente de vista de textura.

[0209] En algunos ejemplos, la unidad de procesamiento de predicción 71 del decodificador de vídeo 30 puede ser un ejemplo de un procesador configurado para implementar los ejemplos descritos en esta divulgación para la derivación de NBDV y la codificación de BVSP. En algunos ejemplos, una unidad (por ejemplo, uno o más procesadores) distintos de la unidad de procesamiento de predicción 71 puede implementar los ejemplos descritos anteriormente. En algunos ejemplos, la unidad de procesamiento de predicción 71 en conjunto con una o más de otras unidades del decodificador de vídeo 30 pueden implementar los ejemplos descritos anteriormente. Aún en algunos otros ejemplos, un procesador

de decodificador de vídeo 30 (no mostrado en la FIG. 7) puede, solo o en conjunto con otros procesadores de decodificador de vídeo 30, implementar los ejemplos descritos anteriormente.

[0210] La FIG. 9 es un diagrama de flujo que ilustra un procedimiento de ejemplo de la divulgación. Las técnicas descritas con referencia a la FIG. 9 se pueden realizar por cualquier elemento estructural o funcional del codificador de vídeo 20 y del decodificador de vídeo 30, incluyendo la unidad de procesamiento de predicción 42 y la unidad de procesamiento de predicción 71. Los siguientes ejemplos se describirán con referencia a un "codificador de vídeo", que como se describe anteriormente, es un término genérico para un codificador de vídeo o bien un decodificador de vídeo (por ejemplo, el codificador de vídeo 20 y el decodificador de vídeo 30).

[0211] Como ejemplo, un codificador de vídeo se puede configurar para codificar datos de vídeo usando una codificación de textura primero (900). Además, el codificador de vídeo se puede configurar para realizar un proceso de derivación de NBDV para un bloque de los datos de vídeo usando una pluralidad de bloques vecinos, en el que el proceso de derivación de NBDV deriva un vector de disparidad. El decodificador de vídeo se puede configurar para realizar el proceso de derivación de NBDV designando un vector de movimiento asociado con un bloque vecino de la pluralidad de bloques vecinos codificados con un modo de predicción de síntesis de vista basada en bloques (BVSP) como un vector de movimiento de disparidad disponible de uno o más vectores de movimiento de disparidad disponibles (910), y designando un vector de movimiento asociado con un bloque vecino de la pluralidad de bloques vecinos codificados usando el modo de predicción entre vistas como un vector de movimiento de disparidad disponible del uno o más vectores de movimiento de disparidad disponibles (920). El codificador de vídeo derivaría el vector de disparidad del uno o más vectores de movimiento de disparidad disponibles (930).

[0212] En otro ejemplo de la divulgación, el codificador de vídeo se configura para realizar el proceso de derivación de NBDV comprobando la pluralidad de bloques vecinos en un orden, y derivando el vector de disparidad en el caso de que un bloque vecino particular se codificara en modo de BVSP, o en el caso de que un bloque vecino particular se codificara usando la predicción entre vistas.

[0213] En otro ejemplo de la divulgación, el codificador de vídeo se configura para realizar el proceso de derivación de NBDV comprobando la pluralidad de bloques vecinos para determinar si alguno de la pluralidad de bloques vecinos se codifica usando la predicción entre vistas, derivando el vector de disparidad a partir del bloque vecino codificado usando la predicción entre vistas en el caso de que el uno de la pluralidad de bloques vecinos se codifique usando la predicción entre vistas en base a la comprobación, comprobando la pluralidad de bloques vecinos para determinar si alguno de la pluralidad de bloques vecinos se codifica usando el modo de BVSP en el caso de que ninguno de la pluralidad de bloques vecinos se codifique usando la predicción de entre vistas, y derivando el vector de disparidad a partir del bloque vecino codificado usando el modo de BVSP en el caso de que uno de la pluralidad de bloques vecinos se codifique usando el modo de BVSP y ninguno de la pluralidad de bloques vecinos se codifiquen usando la predicción de entre vistas en base a la comprobación.

[0214] En otro ejemplo de la divulgación, el codificador de vídeo se configura para realizar el proceso de derivación de NBDV comprobando la pluralidad de bloques vecinos para determinar si alguno de la pluralidad de bloques vecinos se codifica usando el modo de BVSP, derivando el vector de disparidad a partir del bloque vecino codificado usando el modo de BVSP en el caso de que uno de la pluralidad de bloques vecinos se codifique usando el modo de BVSP en base a la comprobación, comprobando la pluralidad de bloques vecinos para determinar si alguno de la pluralidad de bloques vecinos se codifica usando la predicción entre vistas en el caso de que ninguno de la pluralidad de bloques vecinos se codifique usando el modo de BVSP, y derivando el vector de disparidad del bloque vecino codificado usando la predicción de entre vistas en el caso de que uno de la pluralidad de bloques vecinos se codifique usando la predicción de entre vistas y ninguno de la pluralidad de bloques vecinos se codifique usando el modo de BVSP en base a la comprobación.

[0215] En otro ejemplo de la divulgación, el codificador de vídeo se configura para realizar el proceso de derivación de NBDV seleccionando un valor de profundidad a partir de un bloque de profundidad en una vista de referencia de profundidad y convirtiendo el valor de profundidad a un vector de disparidad actualizado, y aplicando el vector de disparidad actualizado al bloque de los datos de vídeo.

[0216] En otro ejemplo de la divulgación, el codificador de vídeo se configura para almacenar el vector de disparidad actualizado como un vector de movimiento para el bloque de datos de vídeo después de que el bloque de los datos de vídeo se codifica. En otro ejemplo de la divulgación, el codificador de vídeo se configura para asignar memoria adicional para almacenar el vector de disparidad derivado.

[0217] En otro ejemplo de la divulgación, el bloque de los datos de vídeo es un macrobloque. En otro ejemplo de la divulgación, el bloque de los datos de vídeo es una subdivisión o división de un macrobloque. En otro ejemplo de la divulgación, el bloque de los datos de vídeo es una unidad de codificación o una unidad de predicción.

[0218] En otro ejemplo de la divulgación, el codificador de vídeo se configura para codificar el bloque de los datos de vídeo usando el modo de BVSP y el vector de disparidad derivado. En otro ejemplo de la divulgación, el codificador

de vídeo se configura para codificar el bloque de los datos de vídeo usando la predicción de vectores de movimiento basada en profundidad (D-MVP) y el vector de disparidad derivado.

[0219] La FIG. 10 es un diagrama de flujo que ilustra otro procedimiento de ejemplo de la divulgación. Las técnicas descritas con referencia a la FIG. 10 se pueden realizar por cualquier elemento estructural o funcional del codificador de vídeo 20 y del decodificador de vídeo 30, incluyendo la unidad de procesamiento de predicción 42 y la unidad de procesamiento de predicción 71. Los siguientes ejemplos se describirán con referencia a un "codificador de vídeo", que como se describe anteriormente, es un término genérico para un codificador de vídeo o bien un decodificador de vídeo (por ejemplo, el codificador de vídeo 20 y el decodificador de vídeo 30).

[0220] En un ejemplo, el codificador de vídeo se configura para realizar un proceso de BVSP en un bloque de datos de vídeo, comprendiendo el proceso de BVSP la realización de un proceso de derivación de NBDV para derivar un vector disparidad. A este respecto, el codificador de vídeo se configura para realizar el proceso de derivación de NBDV designando un vector de movimiento asociado con un bloque vecino de la pluralidad de bloques vecinos codificados con un modo de predicción de síntesis de vista basada en bloques (BVSP) como un vector de movimiento de disparidad disponible de uno o más vectores de movimiento de disparidad disponibles (1010), designando un vector de movimiento asociado con un bloque vecino de la pluralidad de bloques vecinos codificados usando el modo de predicción de entre vistas como un vector de movimiento de disparidad disponible del uno o más vectores de movimiento de disparidad disponibles (1020), y derivando el vector de disparidad a partir del uno o más vectores de movimiento de disparidad disponibles (1030). El codificador de vídeo se configura además para refinar el vector de disparidad derivado para una subregión del bloque de datos de vídeo (1040), y codificar el bloque de datos de vídeo usando BVSP que usa el vector de disparidad refinado (1050).

[0221] En otro ejemplo de la divulgación, al menos de la pluralidad de bloques vecinos es un bloque vecino espacial dentro de una imagen actual o un bloque vecino temporal en una imagen diferente.

[0222] En otro ejemplo de la divulgación, la subregión es una subregión 8x8. En otro ejemplo de la divulgación, la subregión es una de las subregiones 16x16, 4x4, 2x2 y 1x1.

[0223] En otro ejemplo de la divulgación, el codificador de vídeo se configura además para seleccionar un valor de profundidad a partir de uno o más píxeles de profundidad de un bloque de referencia identificada por un vector disparidad producido por el proceso de derivación NBDV para una subregión del bloque de los datos de vídeo, uno o más píxeles de profundidad están en un componente de vista de profundidad del bloque de referencia, pero no píxeles completos dentro de una proximidad a un centro de un bloque de profundidad en una vista de profundidad de referencia que se encuentra en el vector de disparidad.

[0224] En otro ejemplo de la divulgación, el codificador de vídeo se configura además para heredar una componente vertical del vector de disparidad refinado a partir de una componente vertical de un vector de disparidad producido por el proceso de derivación de NBDV. En otro ejemplo de la divulgación, el codificador de vídeo se configura además para establecer un componente vertical del vector de disparidad refinado para que sea cero.

[0225] En otro ejemplo de la divulgación, el codificador de vídeo se configura además para almacenar un vector de disparidad refinado para cada subregión del bloque de datos de vídeo, en el que el vector de disparidad refinado almacenado se usa para el proceso de derivación de NBDV para otro bloque. En otro ejemplo de la divulgación, el codificador de vídeo se configura además para almacenar el vector de disparidad refinado como un vector de movimiento para el bloque de datos de vídeo en el caso de que el bloque de datos de vídeo sea más grande que algún tamaño predeterminado y el vector de disparidad refinado almacenado se usará en un proceso de derivación de NBDV.

[0226] En otro ejemplo de la divulgación, el bloque de datos de vídeo es un macrobloque. En otro ejemplo de la divulgación, el bloque de datos de vídeo es una subdivisión o división de un macrobloque. En otro ejemplo de la divulgación, el bloque de datos de vídeo es una unidad de codificación o una unidad de predicción.

[0227] En uno o más ejemplos, las funciones descritas se pueden implementar en hardware, software, firmware o cualquier combinación de los mismos. Si se implementan en software, las funciones, como una o más instrucciones o código, se pueden almacenar en, o transmitir por, un medio legible por ordenador y ejecutarse mediante una unidad de procesamiento basada en hardware. Los medios legibles por ordenador pueden incluir medios de almacenamiento legibles por ordenador, que correspondan a un medio tangible tal como medios de almacenamiento de datos o medios de comunicación que incluyan cualquier medio que facilite la transferencia de un programa informático desde 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) medios de almacenamiento tangibles legibles por ordenador que sean 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 pueda acceder desde uno o más ordenadores o uno o más procesadores para recuperar instrucciones, código y/o estructuras de datos para la implementación de las técnicas descritas en esta divulgación. Un producto de programa informático puede incluir un medio legible por ordenador.

[0228] 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 se pueda usar para almacenar un código de programa deseado en forma de instrucciones o estructuras de datos y al que se pueda acceder mediante un ordenador. Además, cualquier conexión recibe debidamente la denominación de medio legible por ordenador. Por ejemplo, si las instrucciones se transmiten desde una sede de la Red, un servidor u otro origen remoto usando un cable coaxial, un cable de fibra óptica, un par trenzado, una línea de abonado digital (DSL) o 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, se debería entender 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, como se usa 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, donde algunos discos normalmente emiten datos magnéticamente, mientras que otros discos emiten datos ópticamente con láseres. Las combinaciones de lo anterior también se deberían incluir dentro del alcance de los medios legibles por ordenador.

[0229] Las instrucciones se pueden ejecutar 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* (FPGA) u otros circuitos lógicos integrados o discretos equivalentes. En consecuencia, el término "procesador", como se usa en el presente documento, se puede referir 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 se puede proporcionar dentro de módulos de hardware y/o software dedicados, configurados para la codificación y la descodificación, o incorporados en un códec combinado. También, las técnicas se podrían implementar totalmente en uno o más circuitos o elementos lógicos.

[0230] Las técnicas de esta divulgación se pueden implementar en una amplia variedad de dispositivos o aparatos, incluyendo un teléfono inalámbrico, un circuito integrado (IC) o un conjunto de IC (por ejemplo, un conjunto de chips). Diversos componentes, módulos o unidades se describen en esta divulgación para enfatizar aspectos funcionales de dispositivos configurados para realizar las técnicas divulgadas, pero no requieren necesariamente su realización mediante diferentes unidades de hardware. En cambio, como se describe anteriormente, diversas unidades se pueden combinar en una unidad de hardware de códec o proporcionar por un grupo de unidades de hardware interoperativas, incluyendo uno o más procesadores, como se describe anteriormente, conjuntamente con software y/o firmware adecuados.

[0231] Se han descrito diversos ejemplos.

REIVINDICACIONES

1. Un procedimiento para descodificar datos de vídeo 3D-AVC que incluye una vista básica y una o más vistas dependientes, cada vista que incluye un componente de vista de textura y un componente de vista de profundidad, el procedimiento que comprende:

realizar un proceso de derivación de un vector de disparidad basado en bloques vecinos, NBDV, para derivar (1020) un vector de disparidad para una división de macrobloque; y

dependiendo de que se determine que la división de macrobloques se va a codificar usando la predicción de síntesis de vista basada en bloques, BVSP, el procedimiento que comprende además, para cada subregión 8x8 de la división de macrobloques:

refinar (1040) el vector de disparidad para la subregión 8x8 de la división de macrobloques para crear un vector de disparidad refinado;

descodificar (1050) la subregión 8x8 de la división de macrobloques usando el vector de disparidad refinado para la subregión 8x8 para obtener un bloque de predicción en una vista de textura de referencia; y

almacenar, para la división de macrobloques codificada usando BVSP, el vector de disparidad refinado para la subregión 8x8, en el que el vector de disparidad refinada almacenado se usa para el proceso de derivación de NBDV para otra división de macrobloques,

en el que refinar el vector de disparidad para la subregión 8x8 comprende:

(a) seleccionar un valor de profundidad de uno o más píxeles de profundidad de un bloque de referencia identificado por el vector de disparidad producido por el proceso de derivación de NBDV para la subregión 8x8 de la división de macrobloques, el uno o más píxeles de profundidad que está o están en un componente de vista de profundidad del bloque de referencia, el valor de profundidad seleccionado que es el valor de profundidad máximo de los cuatro píxeles de la esquina del bloque de referencia,

(b) establecer una componente horizontal del vector de disparidad refinado en base al valor de profundidad seleccionado, y

(c) establecer una componente vertical del vector de disparidad refinado para que sea cero,

en el que, en cada unidad de acceso, cada representación codificada de la profundidad de una vista no básica se codifica después de una representación codificada de la textura de la vista no básica correspondiente,

en el que las representaciones codificadas de la textura de la vista básica se codifican sin usar la predicción entre vistas y son decodificables por separado por un decodificador H.264/AVC sin requerir representaciones codificadas correspondientes de la profundidad de la vista básica.

2. Un procedimiento para codificar datos de vídeo 3D-AVC que incluye una vista básica y una o más vistas dependientes, cada vista que incluye un componente de vista de textura y un componente de vista de profundidad, el procedimiento que comprende:

realizar un proceso de derivación de un vector de disparidad basado en bloques vecinos, NBDV, para derivar (1020) un vector de disparidad para una división de macrobloque; y

dependiendo de que se determine que la división de macrobloques se va a codificar usando la predicción de síntesis de vista basada en bloques, BVSP, el procedimiento que comprende además, para cada subregión 8x8 de la división de macrobloques:

refinar (1040), basándose en una vista de profundidad de referencia, el vector de disparidad para la subregión 8x8 de la división de macrobloques para crear un vector de disparidad refinado;

codificar (1050) la subregión 8x8 de la división de macrobloques usando el vector de disparidad refinado para obtener un bloque de predicción en una vista de textura de referencia; y

almacenar el vector de disparidad refinado para la subregión 8x8 de la división de macrobloques, en el que el vector de disparidad refinado almacenado se usa para el proceso de derivación de NBDV para otra división de macrobloques,

en el que refinar el vector de disparidad para la subregión 8x8 comprende:

(a) seleccionar un valor de profundidad de uno o más píxeles de profundidad de un bloque de referencia identificado por el vector de disparidad producido por el proceso de derivación de NBDV para la subregión 8x8 de la división de macrobloque, el uno o más píxeles de profundidad que está o están en un componente de vista de profundidad del bloque de referencia, el valor de profundidad seleccionado que es el valor de profundidad máximo de los cuatro píxeles de la esquina de la referencia,

(b) establecer una componente horizontal del vector de disparidad refinado basándose en el valor de profundidad seleccionado, y

(c) establecer una componente vertical del vector de disparidad refinado para que sea cero,

en el que, en cada unidad de acceso, cada representación codificada de la profundidad de una vista no básica se codifica después de una representación codificada de la textura de la vista no básica correspondiente,

en el que las representaciones codificadas de la textura de la vista básica se codifican sin usar la predicción entre vistas y son decodificables por separado por un decodificador H.264/AVC sin requerir representaciones codificadas correspondientes de la profundidad de la vista básica.

3. Un aparato para codificar datos de vídeo 3D-AVC que incluye una vista básica y una o más vistas dependientes, incluyendo cada vista un componente de vista de textura y un componente de vista de profundidad, el aparato que comprende:

medios para realizar un proceso de derivación de un vector de disparidad basado en bloques vecinos, NBDV, para derivar un vector de disparidad para una división de macrobloque; y

medios para, dependiendo de que se determine que la división de macrobloques se va a codificar usando la predicción de síntesis de vista basada en bloques, BVSP, para cada subregión 8x8 de la división de macrobloques:

refinar, en base a una vista de profundidad de referencia, el vector de disparidad para la subregión 8s8 de la división de macrobloques para crear un vector de disparidad refinado;

codificar la subregión 8xs8 de la división de macrobloques usando el vector de disparidad refinado para obtener un bloque de predicción en una vista de textura de referencia; y

almacenar el vector de disparidad refinado para la subregión 8x8 de la división de macrobloques, en el que el vector de disparidad refinado almacenado se usa para el proceso de derivación de NBDV para otra división de macrobloques,

en el que refinar el vector de disparidad para la subregión 8x8 comprende:

(a) seleccionar un valor de profundidad de uno o más píxeles de profundidad de un bloque de referencia identificado por el vector de disparidad producido por el proceso de derivación de NBDV para la subregión 8x8 de la división de macrobloques, el uno o más píxeles de profundidad que está o están en un componente de vista de profundidad del bloque de referencia, el valor de profundidad seleccionado que es el valor de profundidad máximo de los cuatro píxeles de la esquina del bloque de referencia,

(b) establecer una componente horizontal del vector de disparidad refinado basándose en el valor de profundidad seleccionado, y

(c) establecer una componente vertical del vector de disparidad refinado para que sea cero,

en el que, en cada unidad de acceso, cada representación codificada de la profundidad de una vista no básica se codifica después de una representación codificada de la textura de la vista no básica correspondiente,

en el que las representaciones codificadas de la textura de la vista básica se codifican sin usar la predicción entre vistas y son decodificables por separado por un decodificador H.264/AVC sin requerir representaciones codificadas correspondientes de la profundidad de la vista básica.

4. Un aparato para decodificar datos de vídeo 3D-AVC que incluye una vista básica y una o más vistas dependientes, cada vista que incluye un componente de vista de textura y un componente de vista de profundidad, el aparato que comprende:

medios para realizar un proceso de derivación de un vector de disparidad basado en bloques vecinos, NBDV, para derivar un vector de disparidad para una división de macrobloque; y

medios para, dependiendo de que se determine que la división de macrobloques se va a codificar usando la predicción de síntesis de vista basada en bloques, BVSP, para cada subregión 8x8 de la división de macrobloques:

refinar, basándose en una vista de profundidad de referencia, el vector de disparidad para la subregión 8x8 de la división de macrobloques para crear un vector de disparidad refinado;

descodificar la subregión 8x8 de la división de macrobloques usando el vector de disparidad refinado para obtener un bloque de predicción en una vista de textura de referencia; y

almacenar el vector de disparidad refinado para la subregión 8x8 de la división de macrobloques, en el que el vector de disparidad refinado almacenado se usa para el proceso de derivación de NBDV para otra división de macrobloques,

en el que refinar el vector de disparidad para la subregión 8x8 comprende:

(a) seleccionar un valor de profundidad de uno o más píxeles de profundidad de un bloque de referencia identificado por el vector de disparidad producido por el proceso de derivación de NBDV para la subregión 8x8 de la división de macrobloques, el uno o más píxeles de profundidad que está o están en un componente de vista de profundidad del bloque de referencia, el valor de profundidad seleccionado que es el valor de profundidad máximo de los cuatro píxeles de la esquina del bloque de referencia,

(b) establecer una componente horizontal del vector de disparidad refinado en base al valor de profundidad seleccionado, y

(c) establecer una componente vertical del vector de disparidad refinado para que sea cero,

en el que, en cada unidad de acceso, cada representación codificada de la profundidad de una vista no básica se codifica después de una representación codificada de la textura de la vista no básica correspondiente,

en el que las representaciones codificadas de la textura de la vista básica se codifican sin usar la predicción entre vistas y son descodificables por separado por un decodificador H.264/AVC sin requerir representaciones codificadas correspondientes de la profundidad de la vista básica.

5. Un medio de almacenamiento legible por ordenador que almacena instrucciones que, cuando se ejecutan, hacen que uno o más procesadores lleven a cabo un procedimiento de acuerdo con la reivindicación 1 o la reivindicación 2.

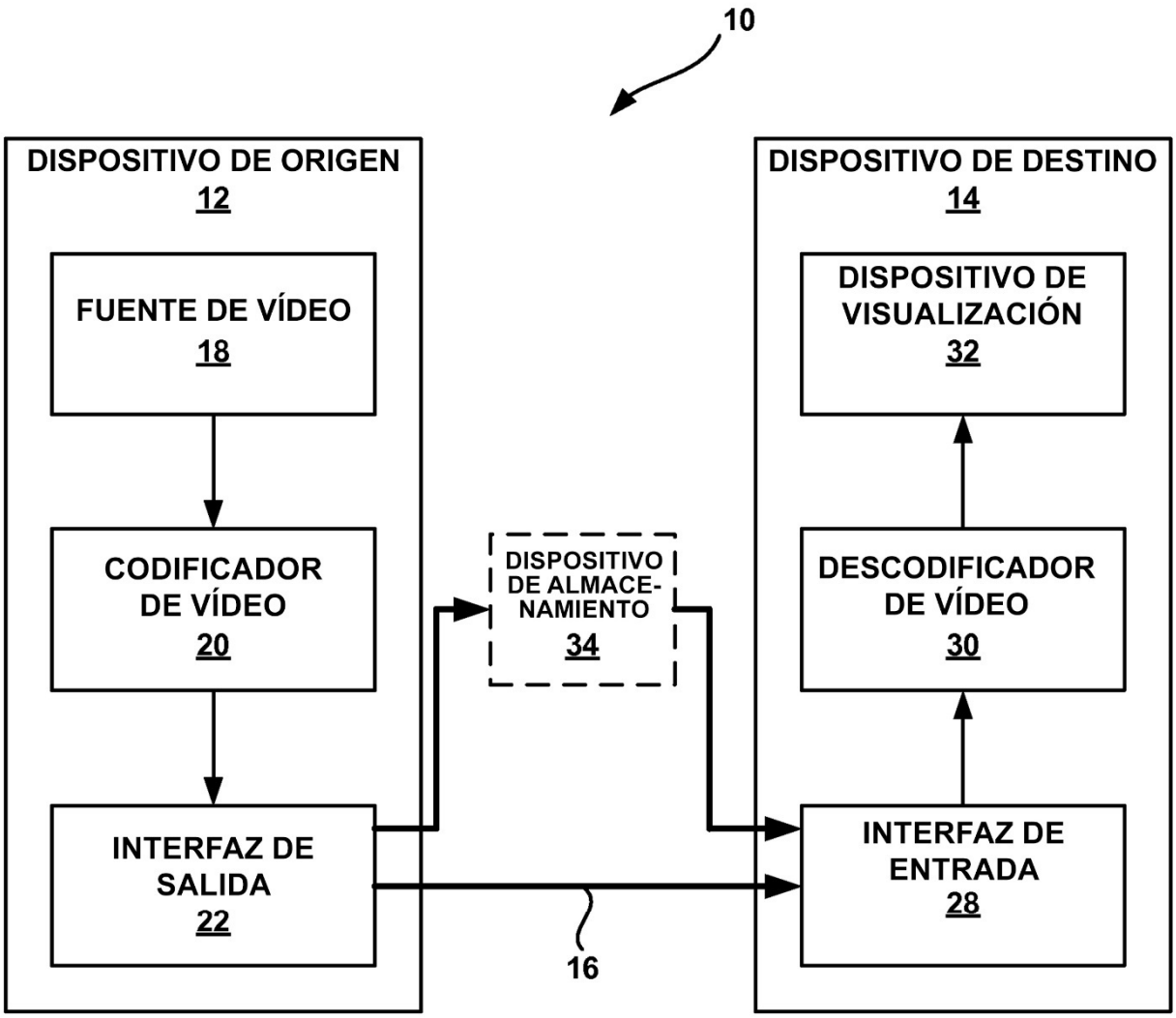
10
DISPOSITIVO DE ORIGEN
12
FUENTE DE VÍDEO
18
CODIFICADOR DE VÍDEO
20
INTERFAZ DE SALIDA
22
DISPOSITIVO DE ALMACE-NAMIENTO
34
16
DISPOSITIVO DE DESTINO
14
DISPOSITIVO DE VISUALIZACIÓN
32
DESCODIFICADOR DE VÍDEO
30
INTERFAZ DE ENTRADA
28

FIG. 1

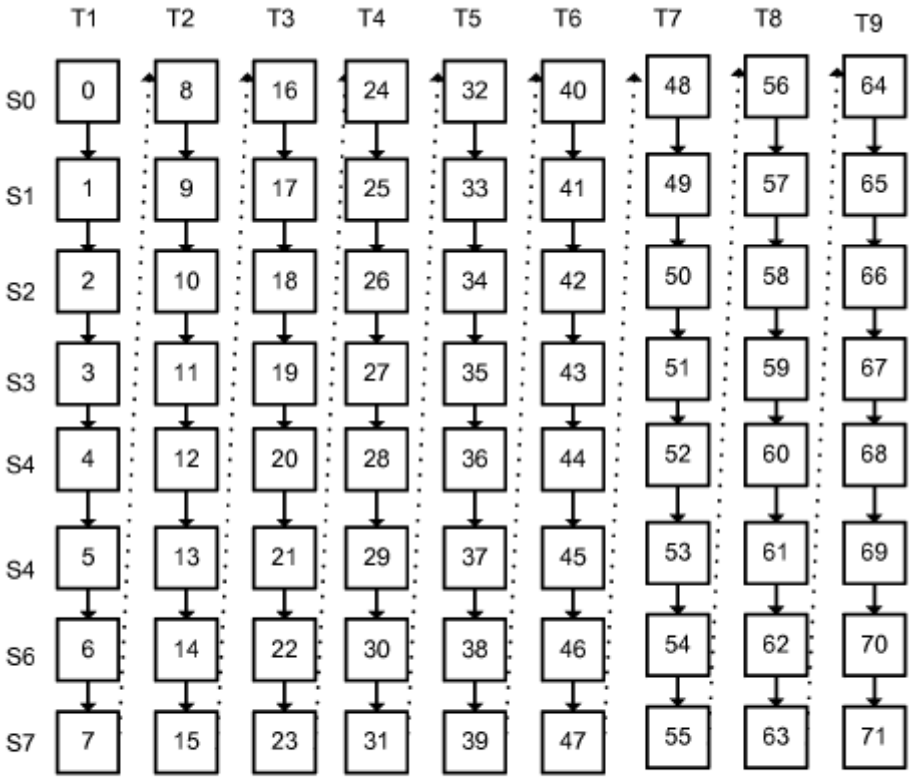
T1
T2
T3
T4
T5
T6
T7
T8
T9
S0
S1
S2
S3
S4
S4
S6
S7
0
8
16
24
32
40
48
56
64
1
9
17
25
33
41
49
57
65
2
10
18
26
34
42
50
58
66
3
11
19
27
35
43
51
59
67
4
12
20
28
36
44
52
60
68
5
13
21
29
37
45
53
61
69
6
14
22
30
38
46
54
62
70
7
15
23
31
39
47
55
63
71

FIG. 2

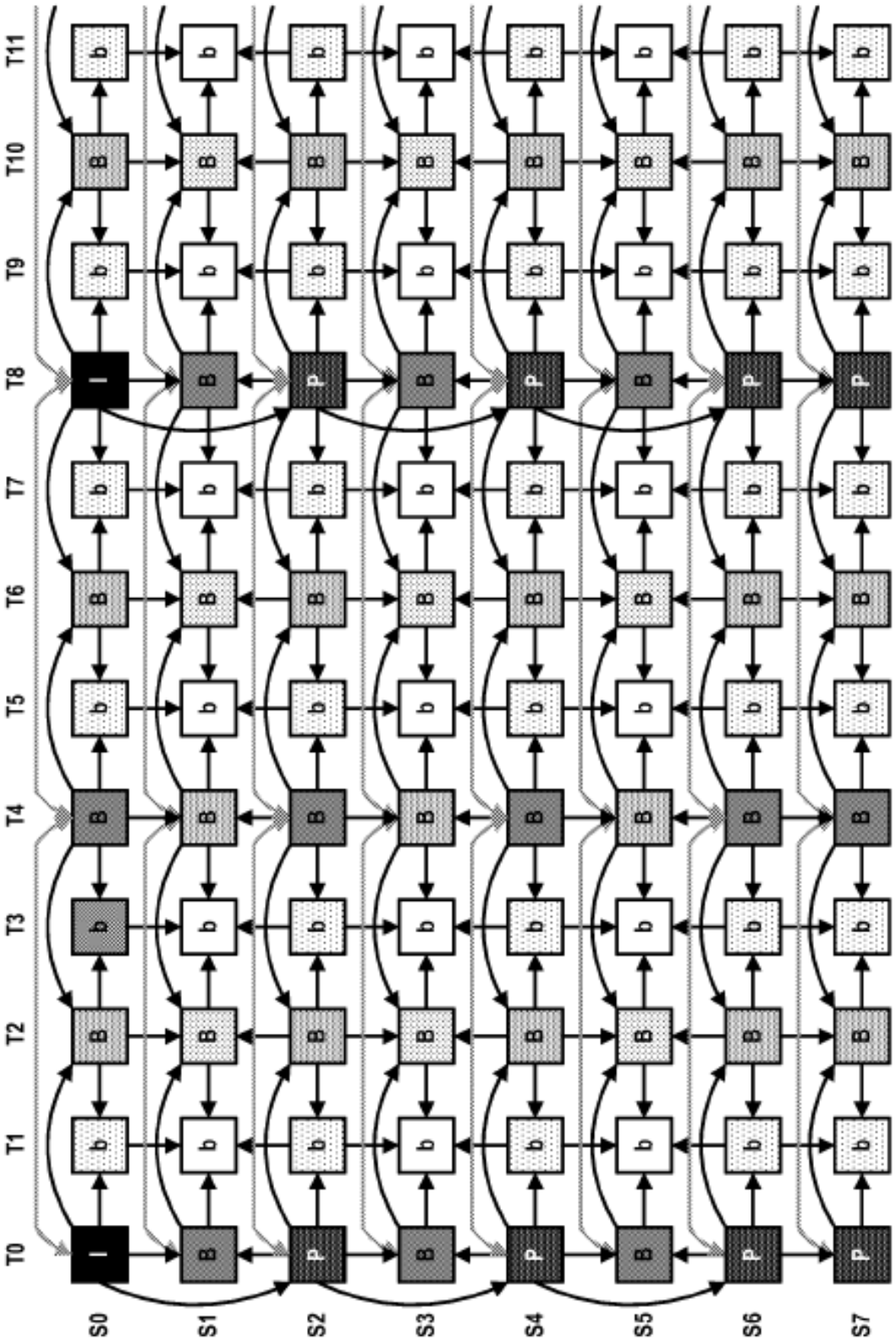

FIG. 3

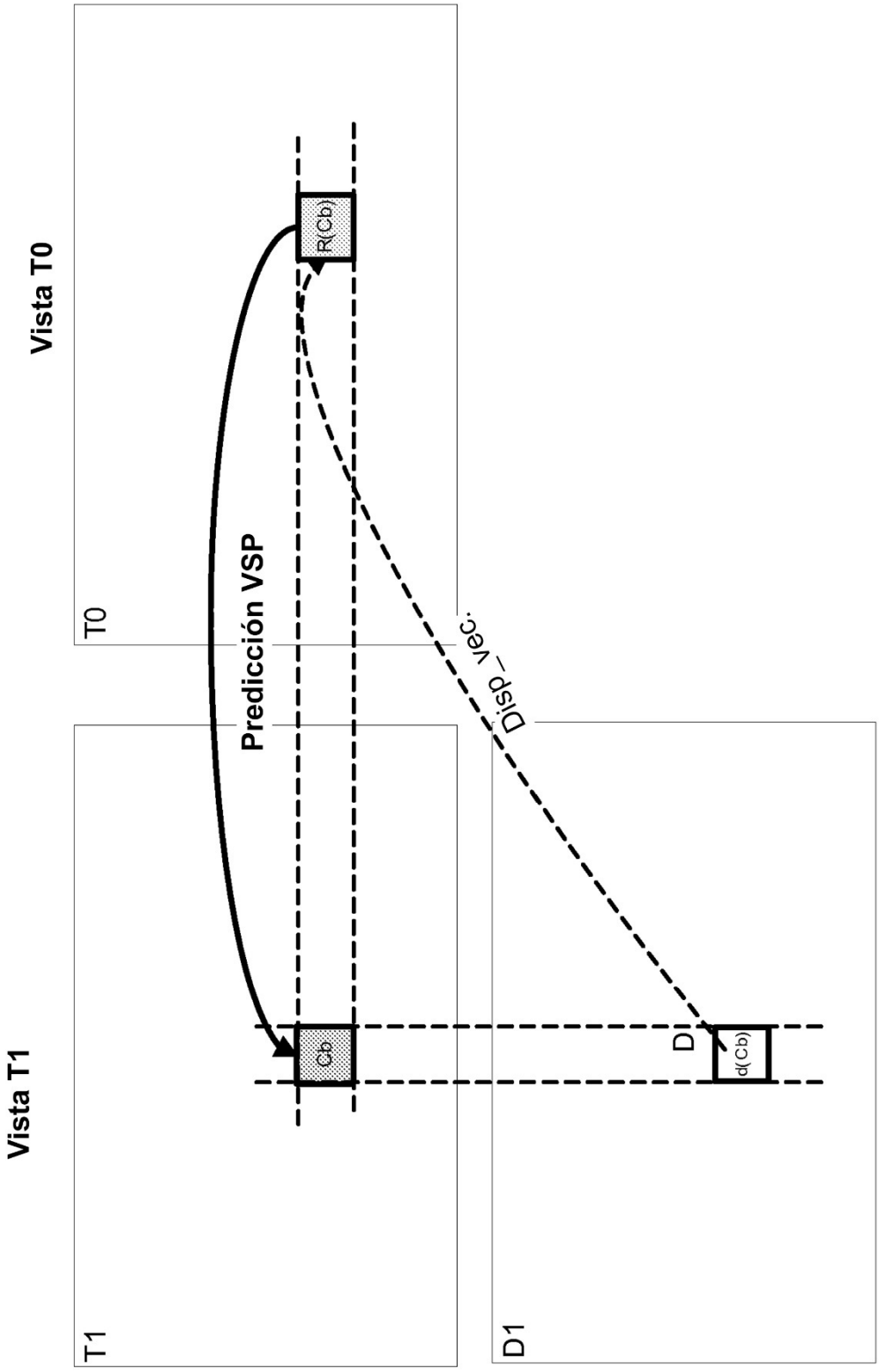

FIG. 4

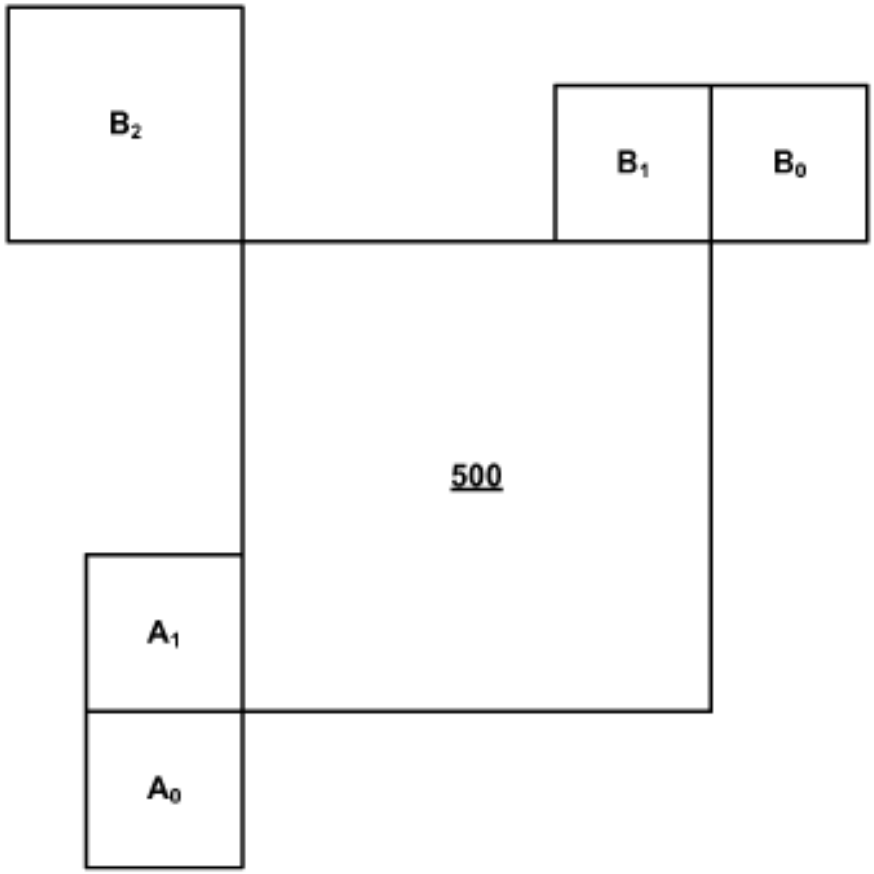

FIG. 5

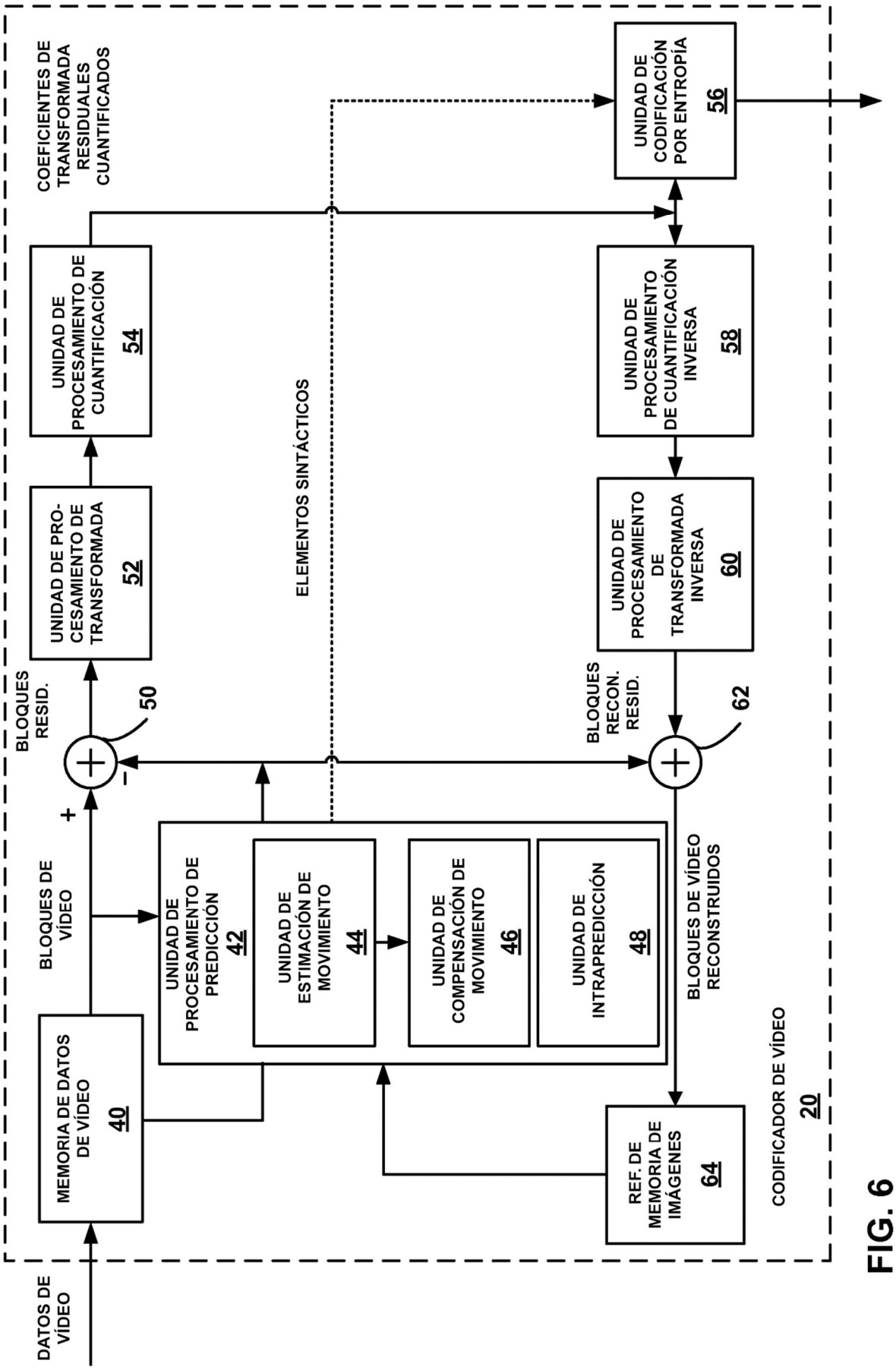
DATOS DE VÍDEO
MEMORIA DE DATOS DE VÍDEO 40
BLOQUES DE VÍDEO
UNIDAD DE PROCESAMIENTO DE PREDICCIÓN 42
UNIDAD DE ESTIMACIÓN DE MOVIMIENTO 44
UNIDAD DE COMPENSACIÓN DE MOVIMIENTO 46
UNIDAD DE INTRAPREDICCIÓN 48
BLOQUES RESID.
50
UNIDAD DE PRO-CESAMIENTO DE TRANSFORMADA 52
UNIDAD DE PROCESAMIENTO DE CUANTIFICACIÓN 54
COEFICIENTES DE TRANSFORMADA RESIDUALES CUANTIFICADOS
ELEMENTOS SINTÁCTICOS
UNIDAD DE CODIFICACIÓN POR ENTROPÍA 56
UNIDAD DE PROCESAMIENTO DE CUANTIFICACIÓN INVERSA 58
UNIDAD DE PROCESAMIENTO DE TRANSFORMADA INVERSA 60
BLOQUES RECON. RESID.
62
BLOQUES DE VÍDEO RECONSTRUIDOS
REF. DE MEMORIA DE IMÁGENES 64
CODIFICADOR DE VÍDEO 20

FIG. 6

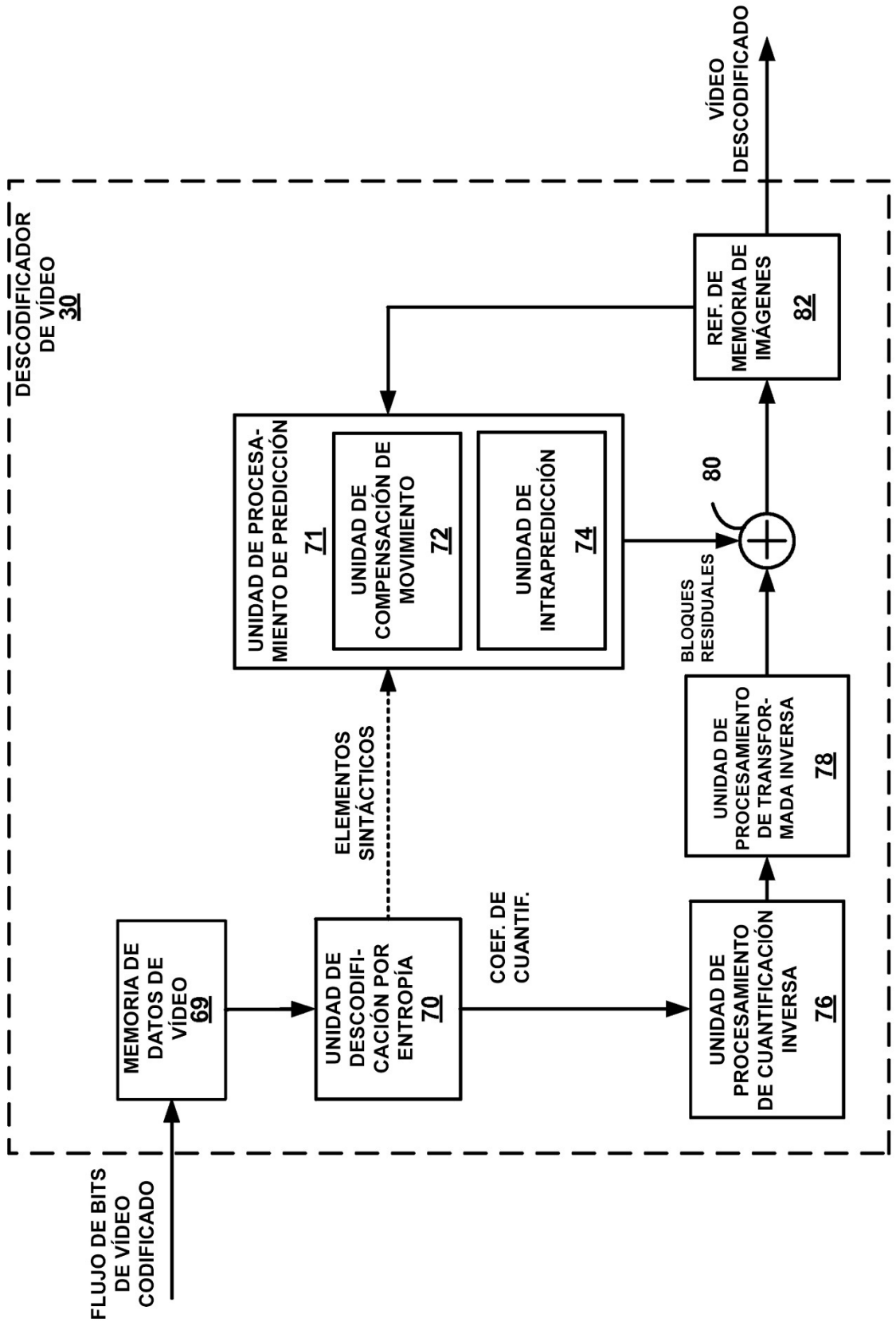
DESCODIFICADOR DE VÍDEO 30
FLUJO DE BITS DE VÍDEO CODIFICADO
MEMORIA DE DATOS DE VÍDEO 69
UNIDAD DE DESCODIFI-CACIÓN POR ENTROPÍA 70
ELEMENTOS SINTÁCTICOS
COEF. DE CUANTIF.
UNIDAD DE PROCESAMIENTO DE CUANTIFICACIÓN INVERSA 76
UNIDAD DE PROCESAMIENTO DE TRANSFOR-MADA INVERSA 78
BLOQUES RESIDUALES
80
UNIDAD DE PROCESA-MIENTO DE PREDICCIÓN 71
UNIDAD DE COMPENSACIÓN DE MOVIMIENTO 72
UNIDAD DE INTRAPREDICCIÓN 74
REF. DE MEMORIA DE IMÁGENES 82
VÍDEO DESCODIFICADO

FIG. 7

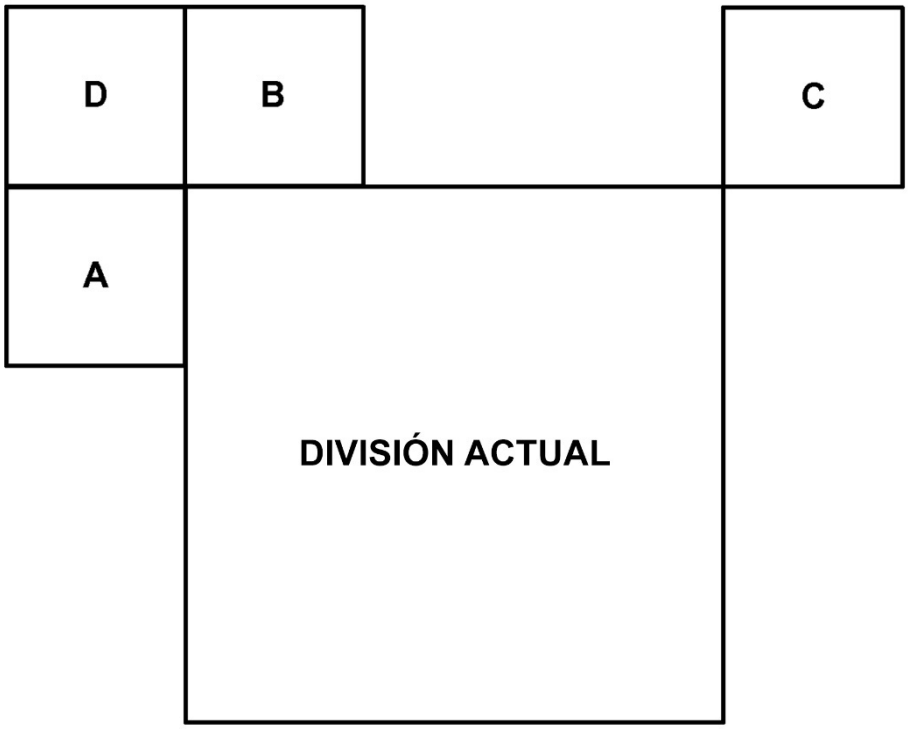

FIG. 8

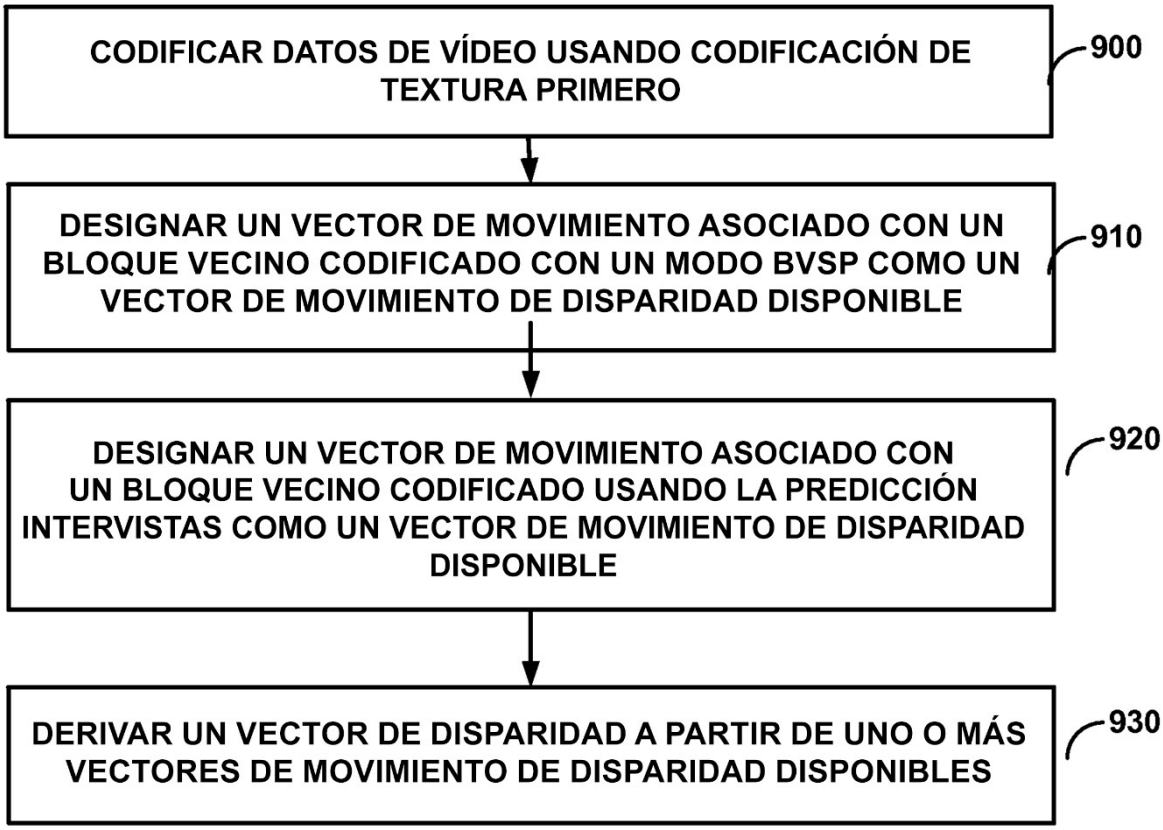
CODIFICAR DATOS DE VÍDEO USANDO CODIFICACIÓN DE TEXTURA PRIMERO
900
DESIGNAR UN VECTOR DE MOVIMIENTO ASOCIADO CON UN BLOQUE VECINO CODIFICADO CON UN MODO BVSP COMO UN VECTOR DE MOVIMIENTO DE DISPARIDAD DISPONIBLE
910
DESIGNAR UN VECTOR DE MOVIMIENTO ASOCIADO CON UN BLOQUE VECINO CODIFICADO USANDO LA PREDICCIÓN INTERVISTAS COMO UN VECTOR DE MOVIMIENTO DE DISPARIDAD DISPONIBLE
920
DERIVAR UN VECTOR DE DISPARIDAD A PARTIR DE UNO O MÁS VECTORES DE MOVIMIENTO DE DISPARIDAD DISPONIBLES
930

FIG. 9

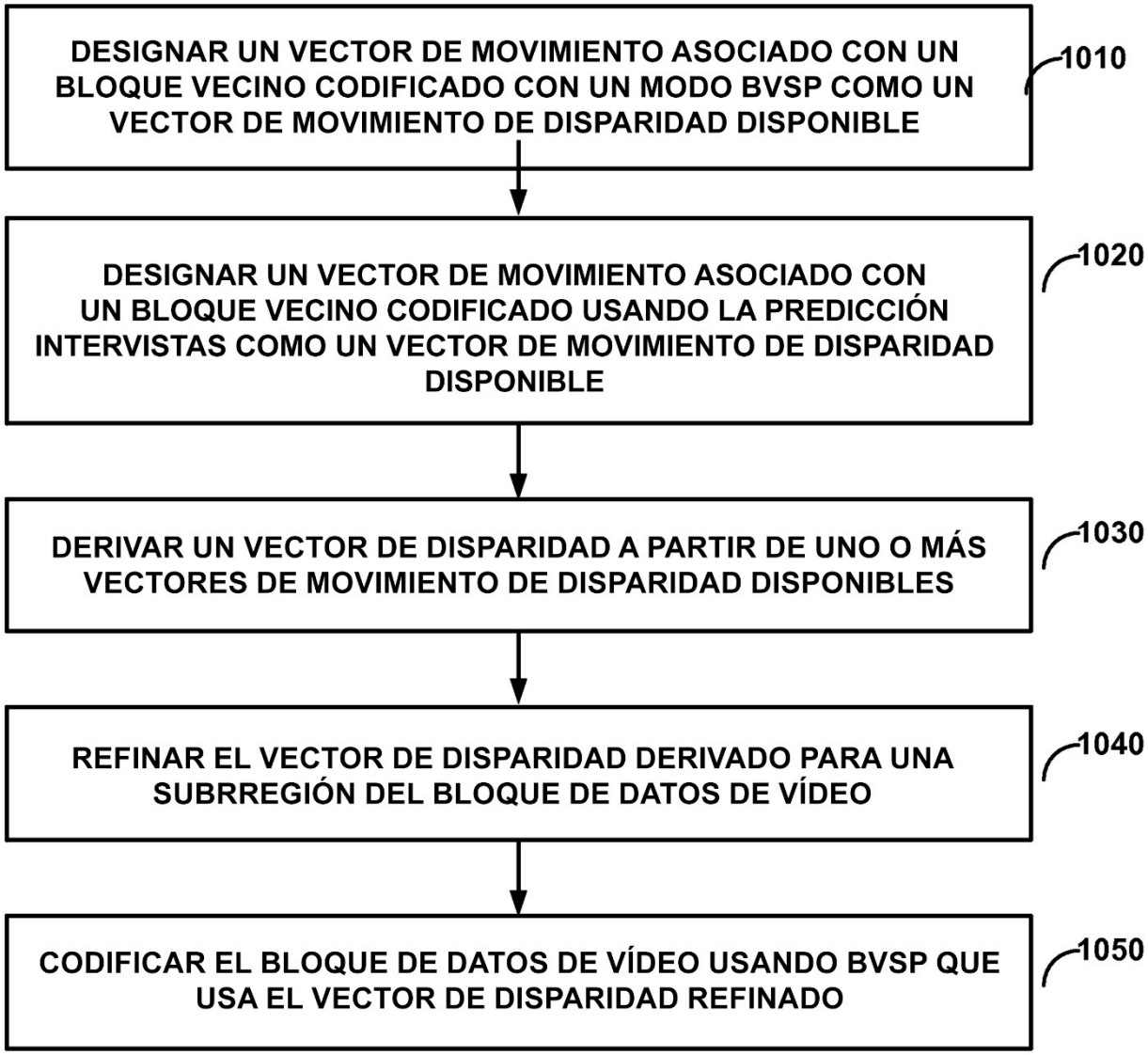
DESIGNAR UN VECTOR DE MOVIMIENTO ASOCIADO CON UN BLOQUE VECINO CODIFICADO CON UN MODO BVSP COMO UN VECTOR DE MOVIMIENTO DE DISPARIDAD DISPONIBLE
1010
DESIGNAR UN VECTOR DE MOVIMIENTO ASOCIADO CON UN BLOQUE VECINO CODIFICADO USANDO LA PREDICCIÓN INTERVISTAS COMO UN VECTOR DE MOVIMIENTO DE DISPARIDAD DISPONIBLE
1020
DERIVAR UN VECTOR DE DISPARIDAD A PARTIR DE UNO O MÁS VECTORES DE MOVIMIENTO DE DISPARIDAD DISPONIBLES
1030
REFINAR EL VECTOR DE DISPARIDAD DERIVADO PARA UNA SUBRREGIÓN DEL BLOQUE DE DATOS DE VÍDEO
1040
CODIFICAR EL BLOQUE DE DATOS DE VÍDEO USANDO BVSP QUE USA EL VECTOR DE DISPARIDAD REFINADO
1050

FIG. 10