

19



OFICINA ESPAÑOLA DE  
PATENTES Y MARCAS

ESPAÑA



11 Número de publicación: **2 988 476**

51 Int. Cl.:

**H04N 19/105** (2014.01)

**H04N 19/52** (2014.01)

**H04N 19/176** (2014.01)

**H04N 19/137** (2014.01)

**H04N 19/54** (2014.01)

12

## TRADUCCIÓN DE PATENTE EUROPEA

T3

86 Fecha de presentación y número de la solicitud internacional: **02.10.2018** **PCT/US2018/053936**

87 Fecha y número de publicación internacional: **11.04.2019** **WO19070683**

96 Fecha de presentación y número de la solicitud europea: **02.10.2018** **E 18793097 (9)**

97 Fecha y número de publicación de la concesión europea: **11.09.2024** **EP 3692715**

54 Título: **Codificación de información de movimiento de predicción afín para codificación de vídeo**

30 Prioridad:

**03.10.2017 US 201762567598 P**

**01.10.2018 US 201816148738**

45 Fecha de publicación y mención en BOPI de la  
traducción de la patente:  
**20.11.2024**

73 Titular/es:

**QUALCOMM INCORPORATED (100.0%)**

**5775 Morehouse Drive**

**San Diego, CA 92121-1714, US**

72 Inventor/es:

**ZHANG, KAI;**

**CHEN, JIANLE;**

**LI, XIANG;**

**CHIEN, WEI-JUNG;**

**CHEN, YI-WEN;**

**ZHANG, LI y**

**KARCZEWICZ, MARTA**

74 Agente/Representante:

**ISERN JARA, Jorge**

ES 2 988 476 T3

Aviso: En el plazo de nueve meses a contar desde la fecha de publicación en el Boletín Europeo de Patentes, de la mención de concesión de la patente europea, cualquier persona podrá oponerse ante la Oficina Europea de Patentes a la patente concedida. La oposición deberá formularse por escrito y estar motivada; sólo se considerará como formulada una vez que se haya realizado el pago de la tasa de oposición (art. 99.1 del Convenio sobre Concesión de Patentes Europeas).

## DESCRIPCIÓN

Codificación de información de movimiento de predicción afín para codificación de vídeo

## 5 Campo técnico

Esta divulgación se refiere a la codificación de vídeo, y más particularmente, a la codificación de información de movimiento de datos de vídeo.

## 10 Antecedentes

Las capacidades de vídeo digital pueden incorporarse en una amplia gama de dispositivos, que incluye televisores digitales, sistemas de difusión digital directa, sistemas de difusión inalámbrica, asistentes digitales personales (PDA), ordenadores portátiles o de escritorio, tabletas, lectores de libros electrónicos, cámaras digitales, dispositivos de grabación digital, reproductores de medios digitales, dispositivos de videojuegos, consolas de videojuegos, celular o teléfonos por radio satélite, los denominados "teléfonos inteligentes", dispositivos de videoconferencia, dispositivos de transmisión de vídeo en directo y similares. Los dispositivos de vídeo digital implementan técnicas de codificación de vídeo, tales como las que se describen en los estándares que se definen por MPEG-2, MPEG-4, ITU-T H.263, ITU-T H.264/MPEG-4, Parte 10, Codificación de Vídeo Avanzada (AVC), estándar de Codificación de Vídeo de Alta Eficiencia (HEVC), ITU-T H.265/Codificación de Vídeo de Alta Eficiencia (HEVC) y extensiones de tales estándares. Los dispositivos de vídeo pueden transmitir, recibir, codificar, decodificar y/o almacenar información de vídeo digital más eficientemente mediante la implementación de tales técnicas de codificación de vídeo.

Las técnicas de codificación de vídeo incluyen la predicción espacial (intraimagen) y/o la predicción temporal (interimagen) para reducir o eliminar la redundancia inherente a las secuencias de vídeo. Para la codificación de vídeo basada en bloques, un segmento de vídeo (por ejemplo, una imagen de vídeo o una porción de una imagen de vídeo) se puede dividir en bloques de vídeo, que también pueden denominarse unidades de árbol de codificación (CTU), unidades de codificación (CU) y/o nodos de codificación. Los bloques de vídeo en un segmento intracodificado (I) de una imagen se codifican mediante el uso de la predicción espacial con respecto a las muestras de referencia de los bloques vecinos de la misma imagen. Los bloques de vídeo en un segmento intercodificado (P o B) de una imagen pueden usar predicción espacial con respecto a las muestras de referencia en bloques vecinos en la misma imagen o predicción temporal con respecto a las muestras de referencia en otras imágenes de referencia. Las imágenes pueden referirse a tramas y las imágenes de referencia pueden referirse como tramas de referencia.

La predicción espacial o temporal da como resultado un bloque predictivo para un bloque a codificar. Los datos residuales representan las diferencias de píxeles entre el bloque original a codificar y el bloque predictivo. Un bloque intercodificado se codifica de acuerdo con un vector de movimiento que apunta a un bloque de muestras de referencia que forma el bloque predictivo y los datos residuales indican la diferencia entre el bloque codificado y el bloque predictivo. Un bloque intracodificado se codifica de acuerdo con un modo de intracodificación y los datos residuales. Para una compresión adicional, los datos residuales pueden transformarse del dominio de píxeles a un dominio de transformación, y dar como resultado coeficientes de transformación residuales, que luego pueden cuantificarse. Los coeficientes de transformación cuantificados, dispuestos inicialmente en una matriz bidimensional, pueden escanearse con el fin de producir un vector unidimensional de coeficientes de transformación, y puede aplicarse codificación de entropía para lograr una compresión aún mayor.

En el documento US 2016/0134886 A1 se describe un procedimiento para obtener información de movimiento en la codificación/decodificación de vídeo, el procedimiento que incluye: calcular un primer predictor de un vector de movimiento de un bloque/subbloque actual de acuerdo con un vector de movimiento de cada bloque de referencia en un primer conjunto de bloques de referencia del bloque/subbloque actual; determinar una primera diferencia de vectores de movimiento entre un vector de movimiento de cada bloque de referencia en un segundo conjunto de bloques de referencia del bloque/subbloque actual y un primer predictor del vector de movimiento del bloque de referencia en el segundo conjunto de bloques de referencia del bloque/subbloque actual; predecir una primera diferencia de vectores de movimiento entre el vector de movimiento del bloque/subbloque actual y el primer predictor del vector de movimiento del bloque/subbloque actual de acuerdo con la primera diferencia de vectores de movimiento de cada bloque de referencia en el segundo conjunto de bloques de referencia para obtener un predictor de la primera diferencia de vectores de movimiento del bloque/subbloque actual; y determinar un segundo predictor del vector de movimiento del bloque/subbloque actual de acuerdo con el predictor de la diferencia del primer vector de movimiento del bloque/subbloque actual y el primer predictor del vector de movimiento del bloque/subbloque actual.

En el documento "Control-Point Representation and Differential Coding Affine-Motion Compensation" de Huang y otros (IEEE Transactions on Circuits and Systems for Video Technology, vol. 23, núm. 10, págs. 1651-1660, octubre de 2013, doi: 10.1109/TCSVT.2013.2254977), se informa que el modelo de movimiento afín es capaz de capturar la rotación, el zoom y la deformación de objetos en movimiento, proporcionando, por lo tanto, una mejor predicción compensada por movimiento. Se informa además que no se usa ampliamente debido a la dificultad tanto en la estimación como en la codificación eficiente de sus parámetros de movimiento. Para aliviar este problema, los

autores proponen una representación de puntos de control que favorece la codificación diferencial para la compresión de parámetros afines.

En el documento "Affine transform prediction for next generation video coding" de Huawei Technologies Corp. (ITU-T SG16 MEETING; 12-10-2015 - 23-10-2015; Ginebra; documento núm. T13-SG16-C-1016) se presenta una herramienta de predicción de transformada afin que, según se informa, muestra una buena eficiencia de codificación, especialmente para acercar/alejar o rotar secuencias de prueba en movimiento.

## Sumario

En general, esta divulgación describe técnicas relacionadas con la predicción entre imágenes. Por ejemplo, las técnicas de esta divulgación incluyen codificación de vectores de movimiento (codificación y/o decodificación) para compensación de movimiento afin en codificación de vídeo basada en bloques. Estas técnicas pueden aplicarse a los estándares de codificación de vídeo existentes y/o futuros.

La invención se define por las reivindicaciones independientes que se adjuntan a esta divulgación. Facilitando la divulgación de la invención tal como se define en las reivindicaciones se encuentran las realizaciones descritas en relación con las Figuras 9, 10, 11 y 12. Cualquier ejemplo o realizaciones de la descripción que no entre dentro del ámbito de las reivindicaciones no forma parte de la invención y se proporcionan sólo con fines ilustrativos.

Las reivindicaciones dependientes incluyen características de realizaciones preferentes.

Los detalles de uno o más ejemplos se establecen en las figuras adjuntas y la descripción más abajo. Otras características, objetivos y ventajas serán evidentes a partir de la descripción y las figuras y a partir de las reivindicaciones.

## Breve descripción de las figuras

Las Figuras 1A y 1B son diagramas conceptuales que ilustran ejemplos de candidatos vecinos espaciales para los modos de fusión y predicción avanzada de vectores de movimiento (AMVP) de Codificación de Vídeo de Alta Eficiencia (HEVC).

La Figura 2 es un diagrama conceptual que ilustra un ejemplo de vector de movimiento de dos puntos afin con cuatro parámetros afines para un bloque actual.

La Figura 3 es un diagrama conceptual que ilustra un ejemplo de modo de interpredicción afin para un bloque actual.

La Figura 4A y 4B son diagramas conceptuales que ilustran un ejemplo de modo de fusión afin para un bloque actual.

La Figura 5 es un diagrama conceptual que ilustra un modelo afin de ejemplo con seis parámetros (tres vectores de movimiento).

La Figura 6 es un diagrama de bloques que ilustra un sistema de codificación y decodificación de vídeo de ejemplo que puede utilizar técnicas de esta divulgación para codificar eficientemente la información de movimiento para la predicción afin.

La Figura 7 es un diagrama de bloques que ilustra un ejemplo de codificador de vídeo que puede implementar técnicas de esta divulgación para codificar información de movimiento de predicción afin.

La Figura 8 es un diagrama de bloques que ilustra un ejemplo de decodificador de vídeo 30 que puede implementar técnicas de esta divulgación para decodificar información de movimiento de predicción afin.

La Figura 9 es un diagrama conceptual que ilustra un ejemplo de predicción de diferencia de vectores de movimiento (MVD) para la predicción de información de movimiento afin.

La Figura 10 es un diagrama conceptual que ilustra un ejemplo de predicción de MVD para la predicción afin con tres vectores de movimiento (predicción afin de seis parámetros).

La Figura 11 es un diagrama de flujo que ilustra un procedimiento de ejemplo para codificar un bloque actual de datos de vídeo de acuerdo con las técnicas de esta divulgación.

La Figura 12 es un diagrama de flujo que ilustra un procedimiento de ejemplo para decodificar un bloque actual de datos de vídeo de acuerdo con las técnicas de esta divulgación.

## Descripción detallada

Los estándares de codificación de vídeo incluyen ITU-T H.261, ISO/IEC MPEG-1 Visual, ITU-T H.262 o ISO/IEC MPEG-2 Visual, ITU-T H.263, ISO/IEC MPEG-4 Visual e ITU-T H.264 (también conocido como ISO/IEC MPEG-4 AVC), que incluye sus extensiones Codificación de Vídeo Escalable (SVC) y Codificación de Vídeo Multivista (MVC).

Recientemente, el diseño de un nuevo estándar de codificación de vídeo, específicamente, ITU-T H.265/Codificación de Vídeo de Alta Eficiencia (HEVC), ha finalizado por el Equipo de Colaboración Conjunta sobre Codificación de Vídeo (JCT-VC) del Grupo de Expertos en Codificación de Vídeo (VCEG) del UIT-T y el Grupo de Expertos en Imágenes en Movimiento (MPEG) de la ISO/CEI. El último borrador de especificación HEVC, denominado en adelante HEVC WD, está disponible en [phenix.int-evry.fr/jct/doc\\_end\\_user/documents/15\\_Geneva/wg11/JCTVC-](http://phenix.int-evry.fr/jct/doc_end_user/documents/15_Geneva/wg11/JCTVC-)

O1003-v2.zip. El JCT-VC también está desarrollando las extensiones de la gama HEVC, llamadas HEVC-Rext. Un borrador de trabajo (WD) de extensiones de gama, denominado RExt WD6 en adelante, está disponible en [phenix.int-evry.fr/jct/doc\\_end\\_user/documents/16\\_San%20Jose/wg11/JCTVC-P1005-v1.zip](http://phenix.int-evry.fr/jct/doc_end_user/documents/16_San%20Jose/wg11/JCTVC-P1005-v1.zip).

Se están realizando investigaciones sobre nuevas herramientas de codificación para la codificación de vídeo futura (por ejemplo, como se estudió en el Equipo de Exploración de Vídeo Conjunto JVET) y se han propuesto tecnologías que mejoran la eficiencia de la codificación para la codificación de vídeo. Hay evidencia de que pueden obtenerse mejoras significativas en la eficiencia de la codificación al explotar las características del contenido de vídeo, especialmente para contenido de alta resolución como 4K, con nuevas herramientas de codificación dedicadas más allá de H.265/HEVC. Se ha invitado a empresas y organizaciones a presentar propuestas en la etapa de exploración para una posible mejora adicional de la eficiencia de la codificación de vídeo.

En H.265/HEVC, para cada bloque, puede estar disponible un conjunto de información de movimiento. Un conjunto de información de movimiento puede contener información de movimiento para direcciones de predicción hacia adelante y hacia atrás. Las direcciones de predicción hacia adelante y hacia atrás son dos direcciones de predicción de un modo de predicción bidireccional y los términos "hacia adelante" y "hacia atrás" no necesariamente tienen un significado geométrico, sino que corresponden a la lista de imágenes de referencia 0 (RefPicList0) y a la lista de imágenes de referencia 1 (RefPicList1) de una imagen actual. Cuando solo hay una lista de imágenes de referencia disponible para una imagen o segmento, solo RefPicList0 está disponible y la información de movimiento de cada bloque de un segmento siempre es hacia adelante.

Para cada dirección de predicción, de acuerdo con H.265/HEVC, la información de movimiento debe contener un índice de referencia y un vector de movimiento. En algunos casos, para simplificar, se puede hacer referencia a un vector de movimiento de manera tal que se suponga que el vector de movimiento tiene un índice de referencia asociado. Un índice de referencia se usa para identificar una imagen de referencia en la lista de imágenes de referencia actual (RefPicList0 o RefPicList1). Un vector de movimiento tiene un componente horizontal y uno vertical.

El recuento de orden de imagen (POC) se usa ampliamente en los estándares de codificación de vídeo para identificar el orden de visualización de una imagen. Aunque hay casos en los que dos imágenes dentro de una secuencia de vídeo codificada pueden tener el mismo valor POC, esto típicamente no sucede dentro de una secuencia de vídeo codificada. Cuando están presentes varias secuencias de vídeo codificadas en un flujo de bits, las imágenes con el mismo valor de POC pueden estar más cerca entre sí en términos de orden de decodificación. Los valores POC de las imágenes se usan típicamente para la construcción de listas de imágenes de referencia, la derivación del conjunto de imágenes de referencia como en HEVC y el escalado del vector de movimiento.

En HEVC, la unidad de codificación más grande en un segmento se denomina bloque de árbol de codificación (CTB). Un CTB contiene un árbol cuádruple, cuyos nodos son unidades de codificación. El tamaño de un CTB puede variar desde 16x16 píxeles a 64x64 píxeles en el perfil principal HEVC (aunque técnicamente pueden soportarse tamaños de CTB de 8x8). Una unidad de codificación (CU) podría tener el mismo tamaño que un CTB y ser tan pequeña como 8x8 píxeles. Cada unidad de codificación puede codificarse con un modo. Cuando una CU se intercodifica, puede dividirse además en dos o más unidades de predicción (PU), o convertirse en una sola PU cuando no corresponde otra partición. Cuando dos PU están presentes en una CU, pueden ser rectángulos de medio tamaño o de dos rectángulos con  $\frac{1}{4}$  o  $\frac{3}{4}$  del tamaño de la CU. Cuando la CU se intercodifica, un conjunto de información de movimiento está presente para cada PU. Además, cada PU está codificada con un modo de interpredicción único para derivar el conjunto de información de movimiento. En HEVC, los tamaños de PU más pequeños son 8x4 y 4x8.

En HEVC, hay dos modos de interpredicción, llamados fusión (la omisión se considera como un caso especial de fusión) y modos de predicción avanzada de vectores de movimiento (AMVP), para una unidad de predicción (PU). En modo de fusión o AMVP, se mantiene una lista de candidatos del vector de movimiento (MV) para múltiples predictores de vectores de movimiento. Los vectores de movimiento, así como también los índices de referencia en el modo de fusión, de la PU actual se generan tomando un candidato de la lista de candidatos del MV.

La lista de candidatos de MV, por HEVC, contiene hasta 5 candidatos para el modo de fusión y solo dos candidatos para el modo de AMVP. Un candidato de fusión puede contener un conjunto de información de movimiento, por ejemplo, vectores de movimiento correspondientes tanto a las listas de imágenes de referencia (lista 0 y lista 1) como a los índices de referencia. Si se identifica un candidato de fusión mediante un índice de fusión, se usan las imágenes de referencia para la predicción de los bloques actuales, así como también se determinan los vectores de movimiento asociados. Sin embargo, en el modo de AMVP, para cada dirección de predicción potencial de la lista 0 o de la lista 1, es necesario señalar explícitamente un índice de referencia, junto con un índice predictor de vector de movimiento MVP a la lista de candidatos MV, ya que el candidato de AMVP sólo contiene un vector de movimiento. En el modo de AMVP, los vectores de movimiento previstos pueden refinarse aún más.

Como puede verse arriba, un candidato de fusión corresponde a un conjunto completo de información de movimiento, mientras que un candidato de AMVP contiene solo un vector de movimiento para una dirección de predicción y un índice de referencia específicos. Los candidatos para ambos modos se derivan de manera similar de

los mismos bloques vecinos espaciales y temporales.

Las Figuras 1A y 1B son diagramas conceptuales que ilustran ejemplos de candidatos vecinos espaciales para los modos de fusión y AMVP de HEVC. En particular, la Figura 1A ilustra candidatos de vectores de movimiento (MV) vecinos espaciales para el modo de fusión, mientras que la Figura 1B ilustra candidatos de MV vecinos espaciales para el modo de AMVP. De acuerdo con HEVC, los candidatos de MV espaciales se derivan de los bloques vecinos que se muestran en las Figuras 1A y 1B, para una PU específica (PU<sub>0</sub>), aunque las técnicas para generar los candidatos a partir de los bloques difieren para los modos de fusión y AMVP.

En el modo de fusión de HEVC, pueden derivarse hasta cuatro candidatos de MV espaciales con los órdenes que se muestran en la Figura 1A con números, y el orden es el siguiente: izquierda (0), arriba (1), arriba a la derecha (2), abajo a la izquierda (3), y arriba a la izquierda (4), como se muestra en la Figura 1A.

En el modo de AMVP de HEVC, los bloques vecinos se dividen en dos grupos: un grupo izquierdo que incluye los bloques 0 y 1, y un grupo superior que incluye los bloques 2, 3 y 4, como se muestra en la Figura 1B. Para cada grupo, el candidato potencial en un bloque vecino que hace referencia a la misma imagen de referencia que la indicada por el índice de referencia señalado tiene la mayor prioridad para ser elegido para formar un candidato final del grupo. Es posible que ningún bloque vecino contenga un vector de movimiento que apunte a la misma imagen de referencia. Por lo tanto, si no se puede encontrar dicho candidato, se escalará el primer candidato disponible para formar el candidato final. Por tanto, pueden compensarse las diferencias de distancia temporal.

La compensación de movimiento en H.265/HEVC se usa para generar un predictor para un bloque intercodificado actual. Puede usarse un vector de movimiento con una precisión de un cuarto de píxel, y los valores de píxeles en posiciones fraccionarias pueden interpolarse mediante el uso de valores de píxeles enteros vecinos para los componentes de luminancia y croma.

La Figura 2 es un diagrama conceptual que ilustra un ejemplo de vector de movimiento de dos puntos afín con cuatro parámetros afines para un bloque actual. En los estándares de códec de vídeo existentes actualmente, solo se aplica un modelo de movimiento traslacional para la predicción de compensación de movimiento (MCP). Sin embargo, en el mundo real, hay muchos tipos de movimientos, por ejemplo, acercar/alejar, rotar, hacer movimientos de perspectiva y otros movimientos irregulares. Si solo se aplica un modelo de movimiento de traslación para la MCP en dichas secuencias de prueba con movimientos irregulares, afectará la precisión de la predicción y dará como resultado una baja eficiencia de codificación. Durante muchos años, muchos expertos en vídeo han intentado diseñar muchos algoritmos para mejorar la MCP y lograr una mayor eficiencia de codificación. Se han propuesto modos de fusión afín e inter afín (AMVP) para tratar modelos de movimiento afín con 4 parámetros como:

$$\begin{cases} V_x = ax + by + e \\ V_y = cx + dy + f \end{cases} \quad (1)$$

En la ecuación (1) anterior, (v<sub>x0</sub>, v<sub>y0</sub>) es el vector de movimiento del punto de control en la esquina superior izquierda del bloque actual de la Figura 2, y (v<sub>x1</sub>, v<sub>y1</sub>) es otro vector de movimiento del punto de control en la esquina superior derecha del bloque actual de la Figura 2. El modelo afín se reduce a:

$$\begin{cases} V_x = \frac{(v_{1x}-v_{0x})}{w}x + \frac{(v_{2x}-v_{0x})}{h}y + v_{0x} \\ V_y = \frac{(v_{1y}-v_{0y})}{w}x + \frac{(v_{2y}-v_{0y})}{h}y + v_{0y} \end{cases} \quad (2)$$

En el software JEM actual, la predicción de movimiento afín solo se aplica a bloques cuadrados. Como extensión natural, la predicción de movimiento afín puede aplicarse a bloques no cuadrados.

La Figura 3 es un diagrama conceptual que ilustra un ejemplo de modo de interpredicción afín para un bloque actual. El bloque actual puede ser una CU actual o una PU actual. En este ejemplo, el bloque actual incluye dos bloques etiquetados como "V0" en una esquina superior izquierda y "V1" en una esquina superior derecha, y bloques vecinos etiquetados como A, B, C, D y E. En particular, el bloque "V0" es vecino de los bloques A, B y C, mientras que el bloque "V1" es vecino de los bloques D y E.

Para cada CU/PU cuyo tamaño sea igual o mayor a 16x16, puede aplicarse el modo de interpredicción afín (modo AF\_INTER) de la siguiente manera. Si la CU/PU actual está en modo AF\_INTER, puede señalarse un indicador afín en el nivel de CU/PU en el flujo de bits. Una lista de candidatos {(v<sub>0</sub>, v<sub>1</sub>)} v<sub>0</sub> = {v<sub>A</sub>, v<sub>B</sub>, v<sub>C</sub>}, v<sub>1</sub> = {v<sub>D</sub>, v<sub>E</sub>}} se construye mediante el uso de los bloques reconstruidos válidos vecinos.

Como se muestra en la Figura 3, la información de movimiento v<sub>0</sub> se selecciona de los vectores de movimiento de los bloques A, B y/o C. El vector de movimiento del bloque vecino se escala de acuerdo con la lista de referencia y la relación entre el POC de la referencia para el bloque vecino, el POC de la referencia para la CU/PU actual y el POC de la CU/PU actual. Y el enfoque para seleccionar v<sub>1</sub> de los bloques vecinos D y E son similares. Si el número de

candidatos en la lista es menor a 2, los candidatos de AMVP se asignan a  $v_0$  y  $v_1$ . Se usa un costo de optimización de distorsión de tasa (RDO) del CU/PU actual para determinar cuál ( $v_0$ ,  $v_1$ ) se selecciona como la predicción del vector de movimiento del punto de control (CPMVP) de la CU/PU actual. Y el índice para indicar la posición del CPMVP en la lista de candidatos se señala en el flujo de bits.

Una vez determinado el CPMVP de la CU/PU afin actual, se aplica la estimación de movimiento afin y se encuentra el CPMV. Luego, la diferencia entre el CPMV y el CPMVP se codifica en el flujo de bits. La predicción de compensación de movimiento afin mencionada anteriormente se aplica para generar los residuos de la CU/PU actual. Finalmente, los residuos de la CU/PU actual se transforman, cuantifican y codifican en el flujo de bits, de acuerdo con procedimientos convencionales.

La Figura 4A y 4B son diagramas conceptuales que ilustra un ejemplo de modo de fusión afin para un bloque actual. El bloque actual puede ser una CU actual o una PU actual. En este ejemplo, el bloque actual tiene cinco bloques vecinos, etiquetados A, B, C, D y E, como se muestra en la Figura 4A.

Cuando la CU/PU actual se aplica en modo de fusión afin (modo AF\_MERGE), obtiene el primer bloque codificado con modo afin de los bloques reconstruidos vecinos válidos de A, B, C, D y E. El orden de selección para el bloque candidato es de izquierda, arriba, arriba a la derecha, abajo a la izquierda, arriba a la izquierda, como se muestra en la Figura 4A. Por ejemplo, si el bloque inferior izquierdo vecino A está codificado en modo afin como se muestra en la Figura 4B, los vectores de movimiento  $v_2$ ,  $v_3$  y  $v_4$  se derivan de la esquina superior izquierda, la esquina superior derecha y la esquina inferior izquierda de la CU/PU que contiene el bloque A. El vector de movimiento  $v_0$  de la esquina superior izquierda de la CU/PU actual se calcula de acuerdo con  $v_2$ ,  $v_3$  y  $v_4$ . De manera similar, el vector de movimiento  $v_1$  del derecho anterior del CU/PU actual se calcula en base a  $v_2$ ,  $v_3$ , y  $v_4$ .

Después del CPMV de la actual CU/PU  $v_0$  y  $v_1$  se calculan, de acuerdo con el modelo de movimiento afin simplificado definido en la ecuación (2) anterior, se genera el MVF de la CU/PU actual. Luego se aplica el MCP afin. Para identificar si la CU/PU actual está codificada con el modo AF\_MERGE, se señala un indicador afin en el flujo de bits cuando hay al menos un bloque vecino codificado en modo afin. Si no existe ningún bloque afin vecino al bloque actual, como se muestra en la Figura 4A, no se escribe ningún indicador afin en el flujo de bits.

En HEVC, la codificación aritmética binaria adaptativa al contexto (CABAC) incluye un proceso de binarización que se usa para convertir un símbolo en un valor binarizado. La binarización permite una codificación aritmética binaria eficiente a través de un mapeo único de elementos de sintaxis no binarios a una secuencia de bits, que se denominan bins. En el software de referencia JEM2.0, para el modo de fusión afin, solo se codifica el indicador afin y se infiere que el índice de fusión es el primer modelo afin vecino disponible en el orden de verificación predefinido A → B → C → D → E. Para el modo inter afin, se codifican dos sintaxis MVD para cada lista de predicción, lo que indica la diferencia de vectores de movimiento entre el vector de movimiento afin derivado y el vector de movimiento predicho.

La Figura 5 es un diagrama conceptual que ilustra un ejemplo de modelo afin con seis parámetros (tres vectores de movimiento). En el documento de Zou y otros, "AFFINE MOTION PREDICTION FOR VIDEO CODING", Solicitante de Estados Unidos núm. 15/587,044, presentada el 4 de mayo de 2017, se describió un esquema de predicción de movimiento afin conmutable. Un bloque con predicción afin puede usar un modelo afin de cuatro parámetros o un modelo afin de seis parámetros de forma adaptativa. Un modelo afin con seis parámetros puede definirse como:

$$\begin{cases} mv_x = ax + by + e \\ mv_y = cx + dy + f \end{cases} \quad (3)$$

Un modelo afin con 6 parámetros tiene tres puntos de control. En otras palabras, un modelo afin con seis parámetros está determinado por tres vectores de movimiento (MV0, MV1 y MV2), por ejemplo, como se muestra en la Figura 5. MV0 es el primer vector de movimiento del punto de control en la esquina superior izquierda de un bloque actual, MV1 es el segundo vector de movimiento del punto de control en la esquina superior derecha del bloque actual y MV2 es el tercer vector de movimiento del punto de control en la esquina inferior izquierda del bloque actual, como se muestra en la Figura 5. El modelo afin construido con los tres vectores de movimiento se calcula como:

$$\begin{cases} mv_x = \frac{(mv_{1x} - mv_{0x})}{w}x + \frac{(mv_{2x} - mv_{0x})}{w}y + mv_{0x} \\ mv_y = \frac{(mv_{1y} - mv_{0y})}{w}x + \frac{(mv_{2y} - mv_{0y})}{w}y + mv_{0y} \end{cases} \quad (4)$$

La ecuación (4) anterior es para un bloque cuadrado que tiene lados iguales a  $w$ . Para un bloque no cuadrado (por ejemplo, un bloque rectangular) que tiene un ancho de  $w$  y una altura de  $h$ , puede usarse el siguiente modelo afin:

$$\begin{cases} mv_x = \frac{(mv_{1x} - mv_{0x})}{w}x + \frac{(mv_{2x} - mv_{0x})}{h}y + mv_{0x} \\ mv_y = \frac{(mv_{1y} - mv_{0y})}{w}x + \frac{(mv_{2y} - mv_{0y})}{h}y + mv_{0y} \end{cases} \quad (5)$$

Un manera similar a la fusión afin para derivar los vectores de movimiento de la esquina superior izquierda y la esquina superior derecha como se describe con respecto a la Figura 4 anterior también puede usarse para derivar los MVP para la esquina superior izquierda, la esquina superior derecha y la esquina inferior izquierda. Se describen ejemplos adicionales en el documento de Chen y otros, "MOTION VECTOR PREDICTION FOR AFFINE MOTION MODEL", Solicitud Provisional de Estados Unidos núm. 62/404,719, presentada el 5 de octubre de 2016.

La Figura 6 es un diagrama de bloques que ilustra un sistema de codificación y decodificación de vídeo de ejemplo 10 que puede utilizar técnicas de esta divulgación para codificar eficientemente la información de movimiento para la predicción afin. Como se muestra en la Figura 6, el sistema 10 incluye un dispositivo fuente 12 que proporciona datos de vídeo codificados para ser decodificados posteriormente por un dispositivo de destino 14. En particular, el dispositivo fuente 12 proporciona los datos de vídeo al dispositivo de destino 14 a través de un medio legible por ordenador 16. El dispositivo fuente 12 y el dispositivo de destino 14 pueden comprender cualquiera de una amplia gama de dispositivos, que incluyen ordenadores de escritorio, ordenadores portátiles (es decir, laptop), tabletas, módulos de conexión, teléfonos móviles tal como los denominados teléfonos "inteligentes", las denominadas almohadillas "inteligentes", televisores, cámaras, dispositivos de visualización, reproductores de medios digitales, consolas de videojuegos, dispositivos de transmisión en tiempo real de vídeos o similares. En algunos casos, el dispositivo fuente 12 y el dispositivo de destino 14 se pueden equipar para la comunicación inalámbrica.

El dispositivo de destino 14 puede recibir los datos de vídeo codificados para decodificarlos a través de un medio legible por ordenador 16. El medio legible por ordenador 16 puede comprender cualquier tipo de medio o dispositivo capaz de mover los datos de vídeo codificados desde el dispositivo fuente 12 al dispositivo de destino 14. En un ejemplo, el medio legible por ordenador 16 puede comprender un medio de comunicación para permitir que el dispositivo fuente 12 transmita datos de vídeo codificados directamente al dispositivo de destino 14 en tiempo real. Los datos de vídeo codificados se pueden modular de acuerdo con un estándar 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 en base a 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 fuente 12 al dispositivo de destino 14.

En algunos ejemplos, los datos codificados pueden salir desde la interfaz de salida 22 hacia un dispositivo de almacenamiento. De manera similar, puede accederse a los datos codificados desde el dispositivo de almacenamiento mediante la interfaz de entrada. El dispositivo de almacenamiento puede incluir cualquiera de una variedad de medios de almacenamiento de datos distribuidos o de acceso local, tales como un disco duro, discos Blu-ray, DVD, CD-ROM, memoria flash, memoria volátil o no volátil, o cualquier otro medio de almacenamiento digital que pueda adecuarse para almacenar datos de vídeo codificados. En un ejemplo adicional, el dispositivo de almacenamiento puede corresponder a un servidor de archivos u otro dispositivo de almacenamiento intermedio que puede almacenar el vídeo codificado que se genera mediante el dispositivo fuente 12. El dispositivo de destino 14 puede acceder a los datos de vídeo que se almacenan desde el dispositivo de almacenamiento a través de transmisión o descarga. El servidor de archivos puede ser cualquier tipo de servidor capaz de almacenar los 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 conectados a la 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, que incluye una conexión a Internet. Esto puede incluir un canal inalámbrico (por ejemplo, una conexión Wi-Fi), una conexión alámbrica (por ejemplo, DSL, módem por cable, etc.) o una combinación de ambos que sea adecuada para acceder a los datos de vídeo codificados almacenados en un servidor de archivos. La transmisión de datos de vídeo codificados desde el dispositivo de almacenamiento puede ser una transmisión continua, una transmisión de descarga o una combinación de las mismas.

Las técnicas de esta divulgación no se limitan necesariamente a las aplicaciones o configuraciones inalámbricas. Las técnicas pueden aplicarse a la codificación de vídeo en apoyo de cualquiera de una variedad de aplicaciones multimedia, tales como difusión de televisión por aire, transmisiones de televisión por cable, transmisiones de televisión por satélite, transmisiones de vídeo por Internet, tales como transmisión dinámica adaptativa a través de HTTP (DASH), vídeo digital que se codifica en un medio de almacenamiento de datos, decodificación de vídeo digital que se almacenan en un medio de almacenamiento de datos u otras aplicaciones. En algunos ejemplos, el sistema 10 se puede configurar para soportar la transmisión de vídeo unidireccional o bidireccional para soportar aplicaciones tal como el vídeo en directo, la reproducción de vídeo, la difusión de vídeo, y/o la telefonía de vídeo.

En el ejemplo de la Figura 6, el dispositivo fuente 12 incluye la fuente de vídeo 18, el codificador de vídeo 20 y la

interfaz de salida 22. El dispositivo de destino 14 incluye la interfaz de entrada 28, el decodificador de vídeo 30, y el dispositivo de visualización 32. De acuerdo con esta divulgación, el codificador de vídeo 20 del dispositivo fuente 12 puede configurarse para aplicar las técnicas para codificar eficientemente la información de movimiento para la predicción afín. En otros ejemplos, un dispositivo fuente y un dispositivo de destino pueden incluir otros componentes o disposiciones. Por ejemplo, el dispositivo fuente 12 puede recibir datos de vídeo de una fuente de vídeo externa 18, tal como una cámara externa. Asimismo, el dispositivo de destino 14 puede interactuar con un dispositivo de visualización externo, en lugar de incluir un dispositivo de visualización integrado.

El sistema 10 ilustrado de la Figura 6 es simplemente un ejemplo. Las técnicas para codificar eficientemente la información de movimiento para la predicción afín pueden ser realizadas por cualquier dispositivo de codificación y/o decodificación de vídeo digital. Aunque generalmente las técnicas de esta divulgación se realizan mediante un dispositivo de codificación de vídeo, las técnicas también pueden realizarse mediante un codificador de vídeo/decodificador, típicamente denominado "CODEC". Además, las técnicas de esta divulgación pueden además, realizarse mediante un preprocesador de vídeo. El dispositivo fuente 12 y el dispositivo de destino 14 son simplemente ejemplos de tales dispositivos de codificación en los que el dispositivo fuente 12 genera datos de vídeo codificados para su transmisión al dispositivo de destino 14. En algunos ejemplos, los dispositivos 12, 14 pueden funcionar de una manera sustancialmente simétrica de manera que cada uno de los dispositivos 12, 14 incluya los componentes de codificación y decodificación de vídeo. De ahí que, el sistema 10 puede admitir la transmisión de vídeo unidireccional o bidireccional entre dispositivos de vídeo 12, 14, por ejemplo, para transmisión de vídeo, reproducción de vídeo, difusión de vídeo o videotelefonía.

La fuente de vídeo 18 del dispositivo fuente 12 puede incluir un dispositivo de captura de vídeo, tales como una cámara de vídeo, un archivo de vídeo que contiene vídeo que se captura previamente y/o una interfaz de alimentación de vídeo para recibir vídeo de un proveedor de contenido de vídeo. Como una alternativa adicional, la fuente de vídeo 18 puede generar datos basados en gráficos de ordenador como la fuente de vídeo, o una combinación de vídeo en vivo, vídeo que se archiva y vídeo que se genera por ordenador. En algunos casos, si la fuente de vídeo 18 es una cámara de vídeo, el dispositivo fuente 12 y el dispositivo de destino 14 pueden formar los llamados teléfonos con cámara o videoteléfonos. Sin embargo, como se mencionó anteriormente, las técnicas que se describen en esta divulgación pueden aplicarse a la codificación de vídeo en general, y pueden aplicarse a las aplicaciones inalámbricas y/o por cable. En cada caso, el vídeo que se captura, se precaptura o se genera por ordenador puede codificarse mediante el codificador de vídeo 20. La información de vídeo codificada puede luego salir mediante la interfaz de salida 22 a un medio legible por ordenador 16.

El medio legible por ordenador 16 puede incluir medios transitorios, tal como una difusión inalámbrica o una transmisión de red por cable, o medios de almacenamiento (es decir, medios de almacenamiento no transitorios), tal como un disco duro, una unidad flash, un disco compacto, un disco de vídeo digital, disco Blu-ray, u otro medio legible por ordenador. En algunos ejemplos, un servidor de red (que no se muestra) puede recibir datos de vídeo codificados desde el dispositivo fuente 12 y proporcionar los datos de vídeo codificados al dispositivo de destino 14, por ejemplo, a través de transmisión de red. De manera similar, un dispositivo informático de una instalación de producción media, tal como una instalación de estampado de discos, puede recibir datos de vídeo codificados desde el dispositivo fuente 12 y producir un disco que contenga los datos de vídeo codificados. Por lo tanto, puede entenderse que el medio legible por ordenador 16 incluye uno o más medios legibles por ordenador de varias formas, en diversos ejemplos.

La interfaz de entrada 28 del dispositivo de destino 14 recibe información del medio legible por ordenador 16. La información del medio legible por ordenador 16 puede incluir la información de sintaxis que se define mediante el codificador de vídeo 20, que se usa también por el decodificador de vídeo 30, que incluye los elementos de sintaxis que describen las características y/o procesamiento de bloques y otras unidades codificadas. El dispositivo de visualización 32 muestra los datos de vídeo decodificados a un usuario, y puede comprender cualquiera de una variedad de dispositivos de visualización tales como un tubo de rayos catódicos (CRT), una pantalla de cristal líquido (LCD), una pantalla de plasma, una pantalla de diodo orgánico de emisión de luz (OLED), u otro tipo de dispositivo de visualización.

El codificador de vídeo 20 y el decodificador de vídeo 30 pueden operar de acuerdo con un estándar de codificación de vídeo, tal como el estándar de Codificación de Vídeo de Alta Eficiencia (HEVC), también conocido como ITU-T H.265. Alternativamente, el codificador de vídeo 20 y el decodificador de vídeo 30 pueden operar de acuerdo con otros estándares de propiedad o de la industria, tal como el estándar ITU-T H.264, también conocido como MPEG-4, Parte 10, Codificación de Vídeo Avanzada (AVC) o con las extensiones de tales estándares. Las técnicas de esta divulgación, sin embargo, no se limitan a ningún estándar de codificación particular. Otros ejemplos de los estándares de codificación de vídeo incluyen MPEG-2 e ITU-T H.263. Aunque no se muestra en la Figura 6, en algunos aspectos, el codificador de vídeo 20 y el decodificador de vídeo 30 pueden integrarse cada uno con un codificador y decodificador de audio, y pueden incluir unidades MUX-DEMUX apropiadas, u otro hardware y software, para manejar la codificación tanto de audio como de vídeo en un flujo de datos común o en flujos de datos separados. Si corresponde, las unidades MUX-DEMUX se pueden ajustar al protocolo multiplexor ITU H.223, o a otros protocolos tales como el protocolo de datagramas de usuario (UDP).



El codificador de vídeo 20 y el decodificador de vídeo 30 pueden implementarse cada uno como cualquiera de una variedad de circuitos codificadores adecuados, como uno o más microprocesadores, procesadores de señales digitales (DSP), circuitos integrados para aplicaciones específicas (ASIC), matrices de puertas lógicas programables en campo (FPGA), lógica discreta, software, hardware, microprograma o cualquiera de sus combinaciones. Cuando las técnicas se implementan de manera parcial en el software, un dispositivo puede almacenar instrucciones para el software en un medio legible por ordenador no transitorio adecuado y ejecutar las instrucciones en el hardware mediante el uso de uno o más procesadores para realizar las técnicas de esta divulgación. Cada uno de los codificadores de vídeo 20 y el decodificador de vídeo 30 puede incluirse en uno o más codificadores o decodificadores, cualquiera de los cuales puede integrarse como parte de un codificador/decodificador combinado (CODEC) en un dispositivo respectivo.

En general, de acuerdo con ITU-T H.265, una imagen de vídeo puede dividirse en una secuencia de unidades de árbol de codificación (CTU) (o unidades de codificación más grandes (LCU)) que pueden incluir muestras de luminancia y croma. Alternativamente, las CTU pueden incluir datos monocromáticos (es decir, solo muestras de luminancia). Los datos de sintaxis dentro de un flujo de bits pueden definir un tamaño para la CTU, que es una unidad de codificación más grande en términos de número de píxeles. Un segmento incluye una serie de CTU consecutivas en el orden de codificación. Una imagen de vídeo puede dividirse en una o más segmentos. Cada CTU puede dividirse en unidades de codificación (CU) de acuerdo con un árbol cuaternario. En general, una estructura de datos de árbol cuaternario incluye un nodo por CU, con un nodo raíz correspondiente a la CTU. Si una CU se divide en cuatro sub-CU, el nodo correspondiente a la CU incluye cuatro nodos hoja, cada uno de los cuales corresponde a una de las sub-CU.

Cada nodo de la estructura de datos de árbol cuaternario puede proporcionar los datos de sintaxis para la correspondiente CU. Por ejemplo, un nodo en el árbol cuaternario puede incluir un indicador de división, que indica si la CU correspondiente al nodo se divide en sub-CU. Los elementos de sintaxis para una CU pueden definirse recursivamente y pueden depender de si la CU se divide en sub-CU. Si una CU no se divide más, se refiere a ella como una CU hoja. En esta divulgación, cuatro sub-CU de una CU hoja también se denominarán CU hojas incluso si no hay una división explícita de la CU hoja original. Por ejemplo, si una CU de tamaño de 16x16 no se divide más, las cuatro sub-CU de 8x8 también se denomina como CU hoja, aunque la CU de 16x16 nunca se dividió.

Una CU tiene un propósito similar a un macrobloque del estándar H.264, excepto que una CU no tiene una distinción de tamaño. Por ejemplo, una CTU puede dividirse en cuatro nodos hijo (también denominados sub-CU), y cada nodo hijo puede ser a su vez un nodo padre y dividirse en otros cuatro nodos hijo. Un nodo hijo final no dividido, denominado nodo hoja del árbol cuaternario, comprende un nodo de codificación, también denominado CU hoja. Los datos de sintaxis asociados con un flujo de bits codificado pueden definir un número máximo de veces que una CTU puede dividirse, denominado profundidad máxima de CU, y también pueden definir un tamaño mínimo de los nodos de codificación. En consecuencia, un flujo de bits también puede definir una unidad de codificación más pequeña (SCU). Esta divulgación usa el término "bloque" para referirse a cualquier CU, unidad de predicción (PU) o unidad de transformación (TU), en el contexto de HEVC, o a estructuras de datos similares en el contexto de otros estándares (por ejemplo, macrobloques y subbloques de los mismos en H.264/AVC).

Una CU incluye un nodo de codificación y las unidades de predicción (PU) y las unidades de transformación (TU) asociadas con el nodo de codificación. El tamaño de la CU corresponde al tamaño del nodo de codificación y generalmente tiene forma cuadrada. El tamaño de la CU puede variar desde 8x8 píxeles hasta el tamaño de la CTU con un tamaño máximo, por ejemplo, de 64x64 píxeles o superior. Cada CU puede contener una o más PU y una o más TU. Los datos de sintaxis asociados con una CU pueden describir, por ejemplo, la partición de la CU en una o más PU. Los modos de particionamiento pueden diferir entre si la CU es codificada en modo directo o en modo de salto, codificada en modo de intrapredicción o codificada en modo de interpretación. Las PU se pueden dividir para que sean de forma no cuadrada. Los datos de sintaxis asociados con una CU también pueden describir, por ejemplo, la división de la CU en una o más TU de acuerdo con un árbol cuaternario. Una TU puede ser de forma cuadrada o no cuadrada (por ejemplo, rectangular).

El estándar HEVC permite transformaciones de acuerdo con las TU, que pueden ser diferentes para diferentes CU. Las TU suelen tener un tamaño en base al tamaño de las PU (o particiones de una CU) dentro de una CU determinada definida para una CTU particionada, aunque no siempre es el caso. Las TU suelen ser del mismo tamaño o más pequeñas que las PU (o particiones de una CU, por ejemplo, en el caso de intrapredicción). En algunos ejemplos, las muestras residuales correspondientes a una CU pueden subdividirse en unidades más pequeñas mediante el uso de una estructura de árbol cuaternario conocida como "árbol cuaternario residual" (RQT). Los nodos hoja del RQT pueden denominarse como unidades de transformación (TU). Los valores de diferencia de píxeles asociados con las TU se pueden transformar para producir coeficientes de transformación, que se pueden cuantificar.

Una CU hoja puede incluir una o más unidades de predicción (PU) cuando se predice mediante el uso de interpretación. En general, una PU representa un área espacial correspondiente a toda o una porción de la CU correspondiente y puede incluir los datos para recuperar y/o generar una muestra de referencia para la PU. Además, una PU incluye los datos relacionados con la predicción. Cuando la CU está codificada entre modos, una o más PU

de la CU pueden incluir datos que definen información de movimiento, tal como uno o más vectores de movimiento, o las PU pueden estar codificadas en modo de omisión. Los datos que definen el vector de movimiento para una PU pueden describir, por ejemplo, un componente horizontal del vector de movimiento, un componente vertical del vector de movimiento, una resolución para el vector de movimiento (por ejemplo, precisión de un cuarto de píxel o precisión de un octavo de píxel), una imagen de referencia a la que apunta el vector de movimiento, y/o una lista de imágenes de referencia (por ejemplo, Lista 0 o Lista 1) para el vector de movimiento.

Las CU hojas también pueden predecirse intramodo. En general, la intrapredicción implica predecir una CU hoja (o particiones de la misma) mediante el uso de un intramodo. Un codificador de vídeo puede seleccionar un conjunto de píxeles vecinos, previamente codificados, para la UC hoja a fin de usarlos para predecir la UC hoja (o particiones de la misma).

Una CU hoja también puede incluir una o más unidades de transformación (TU). Las unidades de transformación se pueden especificar mediante el uso de un RQT (también denominado estructura de árbol cuaternario TU), como se analizó anteriormente. Por ejemplo, un indicador de división puede indicar si una CU hoja se divide en cuatro unidades de transformación. Entonces, cada TU puede dividirse además en más sub-TU. Cuando una TU no se divide más, puede denominarse TU hoja. Generalmente, para la intracodificación, todas las TU hoja que pertenecen a una CU hoja comparten el mismo modo de intrapredicción. Es decir, el mismo modo de intrapredicción se aplica generalmente para calcular los valores pronosticados para todas las TU de una CU hoja. Para la intracodificación, un codificador de vídeo puede calcular un valor residual para cada TU hoja mediante el uso del modo de intrapredicción, como una diferencia entre la porción de la CU correspondiente a la TU y al bloque original. Una TU no se limita necesariamente al tamaño de una PU. Por lo tanto, las TU pueden ser más grandes o más pequeñas que una PU. Para la codificación interna, las particiones de una CU, o la CU misma, pueden colocarse junto a una TU hoja correspondiente para la CU. En algunos ejemplos, el tamaño máximo de una TU hoja puede corresponder al tamaño de la CU hoja correspondiente.

Además, las TU de las CU hoja también se pueden asociar con las respectivas estructuras de datos de árbol cuaternario, denominadas árboles cuaternarios residuales (RQT). Es decir, una CU hoja puede incluir un árbol cuaternario que indica cómo se particiona la CU hoja en las TU. El nodo raíz de un árbol cuaternario TU corresponde generalmente a una CU hoja, mientras que el nodo raíz de un árbol cuaternario CU corresponde generalmente a un CTU (o LCU). Las TU del RQT que no están divididas se denominan TU hoja. En general, esta divulgación usa los términos CU y TU para referirse a CU hoja y TU hoja, respectivamente, a menos que se indique lo contrario.

Una secuencia de vídeo típicamente incluye una serie de tramas o imágenes de vídeo que inician con una imagen de punto de acceso aleatorio (RAP). Una secuencia de vídeo puede incluir datos de sintaxis en un conjunto de parámetros de secuencia (SPS) que caracterizan la secuencia de vídeo. Cada segmento de una imagen puede incluir los datos de sintaxis del segmento que describen un modo de codificación para el segmento respectivo. El codificador de vídeo típicamente opera en los bloques de vídeo dentro de los segmentos de vídeo individuales con el fin de codificar los datos de vídeo. Un bloque de vídeo puede corresponder a un nodo de codificación dentro de una CU. Los bloques de vídeo pueden tener tamaños fijos o variables y pueden diferir en tamaño de acuerdo con un estándar de codificación específico.

A modo de ejemplo, puede realizarse una predicción para PU de distintos tamaños. Suponiendo que el tamaño de una CU particular es  $2N \times 2N$ , la intrapredicción puede realizarse en tamaños de PU de  $2N \times 2N$  o  $N \times N$ , y la interpredicción puede realizarse en tamaños de PU simétricos de  $2N \times 2N$ ,  $2N \times N$ ,  $N \times 2N$  o  $N \times N$ . La partición asimétrica para la interpredicción también puede realizarse para tamaños de PU de  $2N \times nU$ ,  $2N \times nD$ ,  $nL \times 2N$  y  $nR \times 2N$ . En la partición asimétrica, una dirección de una CU no se particiona, mientras que la otra dirección se particiona en 25 % y 75 %. La porción de la CU correspondiente a la partición del 25 % se indica mediante una "n" seguida de una indicación de "Arriba", "Abajo", "Izquierda" o "Derecha". Por tanto, por ejemplo, " $2N \times nU$ " se refiere a una CU de  $2N \times 2N$  que se particiona de manera horizontal con una PU de  $2N \times 0,5N$  en la parte superior y una PU de  $2N \times 1,5N$  en la parte inferior.

En esta divulgación, " $N \times N$ " y "N por N" se pueden utilizar indistintamente para referirse a las dimensiones de los píxeles de un bloque de vídeo en términos de dimensiones verticales y horizontales, por ejemplo,  $16 \times 16$  píxeles o 16 por 16 píxeles. En general, un bloque de  $16 \times 16$  tendrá 16 píxeles en una dirección vertical ( $y = 16$ ) y 16 píxeles en una dirección horizontal ( $x = 16$ ). Del mismo modo, un bloque  $N \times N$  de manera general tiene N píxeles en una dirección vertical y N píxeles en una dirección horizontal, donde N representa un valor entero no negativo. Los píxeles de un bloque pueden organizarse en filas y columnas. Además, los bloques no necesitan tener necesariamente el mismo número de píxeles en la dirección horizontal que en la dirección vertical. Por ejemplo, los bloques pueden comprender  $N \times M$  píxeles, donde M no es necesariamente igual a N.

Después de la codificación intrapredictiva o interpredictiva mediante el uso de las PU de una CU, el codificador de vídeo puede calcular los datos residuales para las TU de la CU. Las PU pueden comprender los datos de sintaxis que describen un procedimiento o modo de generar los datos de píxeles predictivos en el dominio espacial (también denominado dominio de píxeles) y las TU pueden comprender los coeficientes en el dominio de transformación después la aplicación de una transformada, por ejemplo, una transformada de coseno discreta (DCT), una

transformada de entero, una transformada de ondícula, o una transformada conceptualmente similar a los datos de vídeo residuales. Los datos residuales pueden corresponder a las diferencias de píxeles entre los píxeles de la imagen no codificada y los valores de predicción correspondientes a las PU. El codificador de vídeo 20 puede formar las TU para incluir coeficientes de transformación cuantificados representativos de los datos residuales para la CU. Es decir, el codificador de vídeo 20 puede calcular los datos residuales (en forma de un bloque residual), transformar el bloque residual para producir un bloque de coeficientes de transformación y luego cuantificar los coeficientes de transformación para formar coeficientes de transformación cuantificados. El codificador de vídeo 20 puede formar una TU que incluya los coeficientes de transformación cuantificados, así como también otra información de sintaxis (por ejemplo, información de división para la TU).

Como se ha indicado anteriormente, después de cualquier transformación para producir coeficientes de transformación, el codificador de vídeo 20 puede realizar la cuantización de los coeficientes de transformación. La cuantificación de manera general se refiere a un procedimiento en el que los coeficientes de transformación se cuantifican para de manera posible reducir la cantidad de datos utilizados para representar los coeficientes, proporcionando compresión adicional. El procedimiento de cuantificación puede reducir la profundidad de bits asociada con algunos o todos los coeficientes. Por ejemplo, un valor de bits  $n$  se puede redondear a un valor de bits  $m$  durante la cuantificación, donde  $n$  es mayor que  $m$ .

Después de la cuantificación, el codificador de vídeo puede escanear los coeficientes de transformación, al producir un vector unidimensional a partir de la matriz bidimensional que incluye los coeficientes de transformación cuantificados. El escaneo puede diseñarse para colocar los coeficientes de energía más alta (y por lo tanto de frecuencia más baja) en la parte delantera de la matriz y para colocar los coeficientes de energía más baja (y por lo tanto de frecuencia más alta) en la parte posterior de la matriz. En algunos ejemplos, el codificador de vídeo 20 puede utilizar un orden de exploración predefinido para explorar los coeficientes de transformación cuantificados para producir un vector serializado que puede codificarse por entropía. En otros ejemplos, el codificador de vídeo 20 puede realizar un escaneo adaptativo. Después de escanear los coeficientes de transformación cuantificados para formar un vector unidimensional, el codificador de vídeo 20 puede codificar por entropía el vector unidimensional, por ejemplo, de acuerdo con la codificación de longitud variable adaptativa al contexto (CAVLC), la codificación aritmética binaria adaptativa al contexto (CABAC), la codificación aritmética binaria adaptativa al contexto basada en la sintaxis (SBAC), la codificación por Entropía de Particiones de Intervalos de Probabilidad (PIPE) u otra metodología de codificación por entropía. El codificador de vídeo 20 también puede codificar por entropía los elementos de sintaxis asociados con los datos de vídeo codificados para su uso por el decodificador de vídeo 30 al decodificar los datos de vídeo.

Para realizar la CABAC, el codificador de vídeo 20 puede asignar un contexto dentro de un modelo de contexto a un símbolo a transmitir. El contexto se puede relacionar a, por ejemplo, si los valores vecinos del símbolo son o no de valor cero. Para realizar la CAVLC, el codificador de vídeo 20 puede seleccionar un código de longitud variable para transmitir un símbolo a transmitir. Las palabras de código en VLC se pueden construir de manera que los códigos relativamente más cortos correspondan a los símbolos más probables, mientras que los códigos más largos correspondan a los símbolos menos probables. De esta manera, el uso de VLC puede lograr un ahorro de bits con respecto a, por ejemplo, mediante el uso de palabras de código de igual longitud para cada símbolo a transmitir. La determinación de la probabilidad puede ser en base a un contexto asignado al símbolo.

En general, el decodificador de vídeo 30 realiza un proceso sustancialmente similar, aunque recíproco, al realizado por el codificador de vídeo 20 para decodificar datos codificados. Por ejemplo, el decodificador de vídeo 30 cuantiza inversamente y transforma inversamente los coeficientes de una TU recibida para reproducir un bloque residual. El decodificador de vídeo 30 usa un modo de predicción señalado (intra o interpredicción) para formar un bloque predicho. Luego, el decodificador de vídeo 30 combina el bloque predicho y el bloque residual (píxel por píxel) para reproducir el bloque original. Puede realizarse un procesamiento adicional, tal como un proceso de desbloqueo para reducir los artefactos visuales a lo largo de los límites de los bloques. Además, el decodificador de vídeo 30 puede decodificar elementos de sintaxis mediante el uso de CABAC de una manera sustancialmente similar, aunque recíproca, al proceso de codificación CABAC del codificador de vídeo 20.

En general, el codificador de vídeo 20 y el decodificador de vídeo 30 pueden configurarse para codificar de manera más eficiente (codificar o decodificar, respectivamente) información de movimiento para la predicción afín, de acuerdo con las técnicas de esta divulgación. El codificador de vídeo 20 y/o el decodificador de vídeo 30 pueden configurarse para aplicar cualquiera de las diversas técnicas analizadas más abajo, solas o en cualquier combinación.

En algunos ejemplos, el codificador de vídeo 20 y/o el decodificador de vídeo 30 pueden usar la diferencia de vectores de movimiento (MVD) de un vector de movimiento (MV) para predecir la MVD de otro MV en un bloque predicho con predicción afín. La MVD puede definirse como la diferencia entre la MV y la predicción de vector de movimiento (MVP):  $MVD = MV - MVP$ . Más específicamente, si un vector de movimiento ( $MV_x$ ,  $MV_y$ ) se denota por su componente horizontal ( $MV_x$ ) y componente vertical ( $MV_y$ ), y el predictor de vector de movimiento tiene componentes ( $MVP_x$ ,  $MVP_y$ ) el componente horizontal (vertical) de MVD se define como la diferencia de los componentes horizontales (verticales) de MV y MVP, respectivamente. Por tanto, el MVD puede definirse como ( $MVD_x$ ,  $MVD_y$ ),

donde  $MVD_x = MV_x - MVP_x$ , y  $MVD_y = MV_y - MVP_y$ .

Adicional o alternativamente, en algunos ejemplos, el codificador de vídeo 20 y/o el decodificador de vídeo 30 pueden configurarse para usar el MVD del primer MV para predecir el(los) MVD de uno o más otros MV en la predicción afín. La Figura 9 es un diagrama conceptual que ilustra un ejemplo de dicha predicción de MVD. En algunos ejemplos, el codificador de vídeo 20 y/o el decodificador de vídeo 30 pueden configurarse para usar el MVD del primer MV para predecir el MVD de un segundo MV en la predicción afín (por ejemplo, afín de 4 parámetros). La Figura 9 más abajo muestra un ejemplo de predicción de MVD para la predicción afín con dos vectores de movimiento, donde MVD1 es predicho por MVD0.

La Figura 10 es un diagrama conceptual que ilustra un ejemplo de predicción de MVD para la predicción afín con tres vectores de movimiento (predicción afín de seis parámetros). Para la predicción afín de seis parámetros, el codificador de vídeo 20 y/o el decodificador de vídeo 30 pueden usar el MVD del primer MV para predecir el MVD del segundo MV. Además, el codificador de vídeo 20 y/o el decodificador de vídeo 30 pueden usar el MVD del primer MV para predecir el MVD del tercer MV y la predicción afín con tres vectores de movimiento. La Figura 10 muestra un ejemplo de predicción de MVD para la predicción afín con tres vectores de movimiento, donde MVD1 es predice por MVD0 y MVD2 también es predice por MVD0.

Haciendo referencia nuevamente a la Figura 6, en algunos ejemplos, el codificador de vídeo 20 y el decodificador de vídeo 30 pueden configurarse de manera que el primer MV en los ejemplos anteriores se defina como el MV asociado con el punto de control superior izquierdo denominado "MV0" en las Figuras 3, 9 y 10. Alternativamente, el codificador de vídeo 20 y el decodificador de vídeo 30 pueden configurarse para seleccionar de forma adaptativa el primer punto de control asociado con el primer MV. Por ejemplo, el primer punto de control puede depender de información codificada, tal como la forma del bloque. Alternativamente, el codificador de vídeo 20 y el decodificador de vídeo 30 pueden derivar implícitamente el primer punto de control asociado con el primer MV.

Adicional o alternativamente, en algunos ejemplos, el codificador de vídeo 20 y/o el decodificador de vídeo 30 pueden configurarse para aplicar la predicción de MVD entre dos MVD cualesquiera para la predicción afín. Por ejemplo, el codificador de vídeo 20 y el decodificador de vídeo 30 pueden predecir MVD0 a partir de MVD1 para la predicción afín con dos vectores de movimiento. En otro ejemplo, el codificador de vídeo 20 y el decodificador de vídeo 30 pueden predecir MVD1 a partir de MVD0, y predecir MVD2 a partir de MVD1, para la predicción afín con tres vectores de movimiento.

Adicional o alternativamente, en algunos ejemplos, el codificador de vídeo 20 y/o el decodificador de vídeo 30 pueden configurarse para predecir MVDa a partir de MVDb. El codificador de vídeo 20 puede calcular  $MVDa = MVDa - MVDb$  y codificar información que representa MVDa' como parte del flujo de bits, de manera que el decodificador de vídeo 30 puede decodificar esta información para determinar MVDa'. El decodificador de vídeo 30 puede entonces calcular  $MVDa = MVDa' + MVDb$ . En un ejemplo,  $a = 1$  y  $b = 0$  para la predicción afín con cuatro parámetros.

Adicional o alternativamente, en algunos ejemplos, el codificador de vídeo 20 y/o el decodificador de vídeo 30 pueden configurarse para predecir MVDa a partir de MVDb. El codificador de vídeo 20 puede luego calcular  $MVDa' = MVDa - w * MVDb$  y codificar información que representa MVDa' como parte del flujo de bits, de manera que el decodificador de vídeo 30 puede decodificar esta información para determinar MVDa'. El decodificador de vídeo 30 puede entonces calcular  $MVDa = MVDa' + w * MVDb$ . En este ejemplo,  $w$  es un valor de ponderación, tal como 0,5. En un ejemplo,  $a = 1$  y  $b = 0$  para la predicción afín con dos parámetros. Este ejemplo puede implementarse en forma de entero, como  $MVD1' = MVD1 - ((MVD0 + 1) >> 1)$  cuando  $w = 0,5$ , o  $MVD1' = MVD1 - ((MVD0 + 2) >> 2)$  cuando  $w = 0,25$ . En un ejemplo, el codificador de vídeo 20 codifica datos que representan  $w$  como parte del flujo de bits, por ejemplo, a nivel de secuencia (tal como un conjunto de parámetros de secuencia (SPS)), a nivel de imagen (tal como un conjunto de parámetros de imagen (PPS)), a nivel de segmento (tal como en un encabezado de segmento) o a nivel de bloque (tal como en un encabezado de bloque). El decodificador de vídeo 30 puede extraer además esta información señalizada del nivel de información correspondiente.

Adicional o alternativamente, en algunos ejemplos, el codificador de vídeo 20 y/o el decodificador de vídeo 30 pueden configurarse para predecir la MVD de un punto de control a partir de las MVD de más de un punto de control. Por ejemplo, el codificador de vídeo 20 y el decodificador de vídeo 30 pueden predecir MVD2 a partir de MVD0 y MVD1 en el modelo afín de seis parámetros. Por ejemplo, el codificador de vídeo 20 puede calcular  $MVD2' = MVD2 - ((MVD0 + MVD1) >> 1)$ , y codificar la información que representa MVD2' como parte del flujo de bits, de manera que el decodificador de vídeo 30 pueda decodificar esta información para determinar MVD2'. El decodificador de vídeo 30 puede entonces usar esta información para calcular  $MVD2 = MVD2' + ((MVD0 + MVD1) >> 1)$ .

Adicional o alternativamente, en algunos ejemplos, el codificador de vídeo 20 y/o el decodificador de vídeo 30 pueden configurarse para determinar si predecir o no una MVD en base a un valor de MVD. Por ejemplo, el codificador de vídeo 20 puede determinar predecir MVDa a partir de MVDb si  $|MVDb^x + MVDb^y| < T$ ; de otro modo, el codificador de vídeo 20 no predice MVDa a partir de MVDb. En otro ejemplo, el codificador de vídeo 20 puede determinar predecir MVDa a partir de MVDb si  $\max(|MVDb^x + MVDb^y|) < T$ ; de otro modo, el codificador de vídeo 20 no

predice MVDA a partir de MVDB. En otro ejemplo más, el codificador de vídeo 20 puede predecir MVDA a partir de MVDB si  $|MVDB^x + MVDB^y| > T$ ; de otro modo, el codificador de vídeo 20 no predice MVDA a partir de MVDB. En otro ejemplo más, el codificador de vídeo 20 puede predecir MVDA a partir de MVDB si  $\min(|MVDB^x + MVDB^y|) > T$ ; de otro modo, el codificador de vídeo 20 no predice MVDA a partir de MVDB. En los ejemplos anteriores, T representa un umbral, que puede ser un número fijo o señalado por el codificador de vídeo 20 y decodificado por el decodificador de vídeo 30. El codificador de vídeo 20 puede codificar datos que representan si se debe o no predecir MVDA a partir de MVDB en base a cualquiera de las determinaciones de ejemplo anteriores, y el decodificador de vídeo 30 puede decodificar estos datos codificados para determinar si se debe predecir MVDA a partir de MVDB.

Adicional o alternativamente, en algunos ejemplos, el codificador de vídeo 20 y/o el decodificador de vídeo 30 pueden configurarse para predecir los componentes horizontales (x) y/o verticales (y) de un MVD de diferentes maneras. Por ejemplo, el codificador de vídeo 20 y el decodificador de vídeo 30 pueden predecir solo el componente x de MVDB a partir del componente x de MVDA, pero no predecir el componente y de MVDB a partir de otro MVD (por ejemplo, MVDA).

Adicional o alternativamente, en algunos ejemplos, el codificador de vídeo 20 y/o el decodificador de vídeo 30 pueden configurarse para determinar si predecir la MVD en base al valor de MVP. Por ejemplo, el codificador de vídeo 20 puede determinar predecir MVDA a partir de MVDB si  $|MVPa^x - MVPb^x| + |MVPa^y - MVPb^y| < S$ ; de otro modo, el codificador de vídeo 20 no predice MVDA a partir de MVDB. En otro ejemplo, el codificador de vídeo 20 puede determinar predecir MVDA a partir de MVDB si  $\max(|MVPa^x - MVPb^x|, |MVPa^y - MVPb^y|) < S$ ; de otro modo, el codificador de vídeo 20 no predice MVDA a partir de MVDB. En los ejemplos anteriores, S representa un umbral, que puede ser un número fijo o señalado por el codificador de vídeo 20 y decodificado por el decodificador de vídeo 30. El codificador de vídeo 20 puede codificar datos que representan si se debe o no predecir MVDA a partir de MVDB en base a cualquiera de las determinaciones de ejemplo anteriores, y el decodificador de vídeo 30 puede decodificar estos datos codificados para determinar si se debe predecir MVDA a partir de MVDB.

Adicional o alternativamente, en algunos ejemplos, el codificador de vídeo 20 y/o el decodificador de vídeo 30 pueden configurarse para determinar si predecir un MVD para un bloque predicho mediante el uso de la predicción afín en base a un procedimiento de predicción de movimiento. Por ejemplo, si el MVP proviene del procedimiento de derivación de MVP en JEM como se describió anteriormente con respecto a la Figura 3, el codificador de vídeo 20 y el decodificador de vídeo 30 pueden determinar no usar la predicción de MVD. Como otro ejemplo, si MVP proviene del procedimiento de derivación de MVP similar a la fusión afín como se describió anteriormente con respecto a la Solicitud Provisional de Estados Unidos núm. 62/404,719, el codificador de vídeo 20 y el decodificador de vídeo 30 pueden determinar usar la predicción de MVD. Adicional o alternativamente, el codificador de vídeo 20 y el decodificador de vídeo 30 pueden determinar si usar la predicción de MVD en base a si se usa compensación de iluminación para el bloque fuente del MVP.

Adicional o alternativamente, en algunos ejemplos, el codificador de vídeo 20 y/o el decodificador de vídeo 30 pueden configurarse para determinar si predecir una MVD en base a un tamaño y/o una forma de un bloque actual. Por ejemplo, el codificador de vídeo 20 y el decodificador de vídeo 30 pueden determinar usar la predicción de MVD cuando  $W \cdot H > T$ , donde W representa un ancho del bloque actual, H representa una altura del bloque actual y T representa un valor de umbral. T puede ser un número fijo o señalado desde el codificador de vídeo 20 al decodificador de vídeo 30 en el flujo de bits. En otro ejemplo, el codificador de vídeo 20 y el decodificador de vídeo 30 pueden determinar usar la predicción de MVD cuando  $W \cdot H < T$ .

Adicional o alternativamente, en algunos ejemplos, el codificador de vídeo 20 y/o el decodificador de vídeo 30 pueden configurarse para codificar datos que representen si predecir o no un MVD (o un componente del mismo) en un flujo de bits. Es decir, el codificador de vídeo 20 puede codificar datos que representan si se debe predecir uno o ambos componentes (horizontal y vertical) de un MVD en el flujo de bits, y el decodificador de vídeo 30 puede determinar si se debe predecir uno o ambos componentes del MVD a partir de los datos codificados del flujo de bits (decodificando los datos).

Adicional o alternativamente, en algunos ejemplos, el codificador de vídeo 20 y/o el decodificador de vídeo 30 pueden configurarse para codificar datos que representen que punto(s) de control usar como referencia para la predicción de MVD un flujo de bits. Es decir, el codificador de vídeo 20 puede codificar estos datos, y el decodificador de vídeo 30 puede decodificar estos datos para determinar qué punto(s) de control usar como referencia para la predicción de MVD.

Adicional o alternativamente, en algunos ejemplos, el codificador de vídeo 20 y/o el decodificador de vídeo 30 pueden configurarse para generar la MVP de un MV a partir del MVD de otro MV en una predicción de bloques mediante el uso de la predicción afín.

Adicional o alternativamente, en algunos ejemplos, el codificador de vídeo 20 y/o el decodificador de vídeo 30 pueden configurarse para usar el MVD de un primer MV para generar el(los) MVP de uno o más otros MV en la predicción afín. En algunos ejemplos, el codificador de vídeo 20 y el decodificador de vídeo 30 pueden usar el MVD del primer MV para generar el MVP de un segundo MV en la predicción afín (por ejemplo, afín de cuatro

parámetros). En otro ejemplo, para la predicción afín de seis parámetros, el codificador de vídeo 20 y el decodificador de vídeo 30 pueden usar el MVD del primer MV para generar el MVP de un segundo MV en la predicción afín con tres vectores de movimiento. Adicionalmente, el codificador de vídeo 20 y el decodificador de vídeo 30 pueden usar el MVD del primer MV para generar el MVP del tercer MV en la predicción afín con tres vectores de movimiento. Alternativamente, el codificador de vídeo 20 y el decodificador de vídeo 30 pueden usar el MVD del segundo MV para generar el MVP del tercer MV en la predicción afín con tres vectores de movimiento.

Adicional o alternativamente, en algunos ejemplos, el codificador de vídeo 20 y/o el decodificador de vídeo 30 pueden configurarse para generar MVP0 a partir de MVD1 para la predicción afín con dos vectores de movimiento. En otro ejemplo, el codificador de vídeo 20 y/o el decodificador de vídeo 30 pueden configurarse para generar MVP1 a partir de MVD0, y para generar MVP2 a partir de MVD1, para la predicción afín con tres vectores de movimiento.

Adicional o alternativamente, en algunos ejemplos, el codificador de vídeo 20 y/o el decodificador de vídeo 30 pueden configurarse para calcular MVPa de acuerdo con  $MVPa = MVP'a + MVD_b$ . MVP'a representa el MVP generado de la forma original sin tener en cuenta MVDb como se describió anteriormente para AMVP y el modo de fusión para la predicción afín. En un ejemplo,  $a = 1$  y  $b = 0$  para la predicción afín con cuatro parámetros (dos vectores de movimiento).

Adicional o alternativamente, en algunos ejemplos, el codificador de vídeo 20 y/o el decodificador de vídeo 30 pueden configurarse para calcular MVPa de acuerdo con  $MVPa = MVP'a + w * MVD_b$ , donde  $w$  es un valor de ponderación, tal como 0,5. En este ejemplo, MVP'a es el MVP generado de la forma original sin tener en cuenta MVDb como se describió anteriormente para AMVP y el modo de fusión para la predicción afín. En un ejemplo,  $a = 1$  y  $b = 0$  para la predicción afín con dos parámetros. Este ejemplo puede implementarse en forma de entero, como  $MVP1 = MVP'1 + ((MVD0+1) >> 1)$  cuando  $w = 0,5$ , o  $MVP1 = MVP'1 + ((MVD0+2) >> 2)$  cuando  $w = 0,25$ . En un ejemplo, el codificador de vídeo 20 determina  $w$  y señala el valor de  $w$  en el flujo de bits a nivel de secuencia, nivel de imagen, nivel de segmento o nivel de bloque. El decodificador de vídeo 30, en consecuencia, decodificaría el valor de  $w$  desde el nivel apropiado de información.

Adicional o alternativamente, en algunos ejemplos, el codificador de vídeo 20 y/o el decodificador de vídeo 30 pueden configurarse para generar el MVP de un punto de control a partir de las MVD de varios otros puntos de control. Por ejemplo, el codificador de vídeo 20 y/o el decodificador de vídeo 30 pueden generar MVP2 a partir de MVD0 y MVD1 en el modelo afín de seis parámetros. El codificador de vídeo 20 y/o el decodificador de vídeo 30 pueden calcular MVP2 como  $MVP2 = MVP'2 + ((MVD0 + MVD1) >> 1)$ .

Adicional o alternativamente, en algunos ejemplos, el codificador de vídeo 20 y/o el decodificador de vídeo 30 pueden configurarse para usar el MV de un punto de control para generar el(los) MVP del(de los) MV de uno o más otros punto(s) de control en una predicción de bloques mediante el uso de la predicción afín. En algunos ejemplos, el codificador de vídeo 20 y el decodificador de vídeo 30 pueden usar el primer MV para generar el MVP del segundo MV en la predicción afín (por ejemplo, afín de cuatro parámetros). En otro ejemplo, para la predicción afín de seis parámetros, el codificador de vídeo 20 y/o el decodificador de vídeo 30 pueden usar el primer MV para generar el MVP del segundo MV en la predicción afín con tres vectores de movimiento, y usar el primer MV para generar el MVP del tercer MV en la predicción afín con tres vectores de movimiento. Alternativamente, para la predicción afín de seis parámetros, el codificador de vídeo 20 y el decodificador de vídeo 30 pueden usar el segundo MV para generar el MVP del tercer MV en la predicción afín con tres vectores de movimiento.

Adicional o alternativamente, en algunos ejemplos, el codificador de vídeo 20 y/o el decodificador de vídeo 30 pueden configurarse para calcular MVPa como  $MVPa = (MVP'a + MVb) >> 1$ . MVP'a es el MVP generado de la forma original sin tener en cuenta MVDb como se describió anteriormente en la sección que analiza AMVP y la fusión para la predicción afín. En un ejemplo,  $a = 1$  y  $b = 0$  para la predicción afín con cuatro parámetros.

Adicional o alternativamente, en algunos ejemplos, el codificador de vídeo 20 y/o el decodificador de vídeo 30 pueden configurarse para calcular MVPa como  $MVPa = w1 * MVP'a + w2 * MVb$ . En este ejemplo,  $w1$  y  $w2$  son valores de ponderación que pueden tener valores iguales o diferentes, por ejemplo,  $w1 = w2 = 0,5$ . MVP'a es el MVP generado de la forma original sin tener en cuenta MVDb como se describió en la sección anterior que analiza AMVP y el modo de fusión para la predicción afín. En un ejemplo,  $a = 1$  y  $b = 0$  para la predicción afín con cuatro parámetros. Este ejemplo puede implementarse en forma de entero, como  $MVP1 = (3 * MVP'1 + MV0 + 2) >> 2$  cuando  $w1 = 0,75$  y  $w2 = 0,25$ . En un ejemplo, el codificador de vídeo 20 codifica datos para  $w1$  y  $w2$  en el flujo de bits a nivel de secuencia, nivel de imagen, nivel de segmento o nivel de bloque. De la misma forma, el decodificador de vídeo 30 determinaría  $w1$  y  $w2$  al decodificar estos datos en el nivel apropiado.

El codificador de vídeo 20 puede enviar además datos de sintaxis, tales como datos de sintaxis basados en bloques, datos de sintaxis basados en imágenes y datos de sintaxis basados en secuencias, al decodificador de vídeo 30, por ejemplo, en un encabezado de imagen, un encabezado de bloque, un encabezado de segmento u otros datos de sintaxis, tales como un conjunto de parámetros de secuencia (SPS), un conjunto de parámetros de imagen (PPS) o un conjunto de parámetros de vídeo (VPS).

El codificador de vídeo 20 y el decodificador de vídeo 30 pueden implementarse cada uno como cualquiera de una variedad de circuitos de codificador o decodificador adecuados, según corresponda, tal como uno o más microprocesadores, procesadores de señales digitales (DSP), circuitos integrados para aplicaciones específicas (ASIC), matrices de puertas lógicas programables en campo (FPGA), circuitos lógicos discretos, software, hardware, microprograma o cualquier combinación de los mismos. Cada uno del codificador de vídeo 20 y 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 combinado (CODEC). Un dispositivo que incluye un codificador de vídeo 20 y/o un decodificador de vídeo 30 puede comprender un circuito integrado, un microprocesador, y/o un dispositivo de comunicación inalámbrica, tal como un teléfono celular.

La Figura 7 es un diagrama de bloques que ilustra un ejemplo de codificador de vídeo 20 que puede implementar técnicas de esta divulgación para codificar información de movimiento de predicción afín. El codificador de vídeo 20 puede realizar la intracodificación y la intercodificación de los bloques de vídeo dentro de los segmentos de vídeo. 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 determinado. La intercodificación se basa en la predicción temporal para reducir o eliminar la redundancia temporal en el vídeo dentro de tramas o imágenes adyacentes de una secuencia de vídeo. El intramodo (modo I) puede referirse a cualquiera de diversos modos de codificación que se basan en el espacio. Los intermodos, tales como la predicción unidireccional (modo P) o bipredicción (modo B), pueden referirse a cualquiera de diversos modos de codificación que se basan en el tiempo.

Como se muestra en la Figura 7, el codificador de vídeo 20 recibe un bloque de vídeo actual dentro de una trama de vídeo a codificar. En el ejemplo de la Figura 7, el codificador de vídeo 20 incluye la unidad de selección de modo 40, la memoria de imágenes de referencia 64 (la cual puede también denominarse búfer de imagen decodificada (DPB)), el sumador 50, la unidad de procesamiento de transformación 52, la unidad de cuantificación 54 y la unidad de codificación de entropía 56. La unidad de selección de modo 40, a su vez, incluye la unidad de compensación de movimiento 44, la unidad de estimación de movimiento 42, la unidad de intrapredicción 46, y la unidad de partición 48. Para la reconstrucción de bloques de vídeo, el codificador de vídeo 20 también incluye la unidad de cuantificación inversa 58, la unidad de transformación inversa 60, y el sumador 62. También puede incluirse un filtro de desbloqueo (no mostrado en la Figura 7) para filtrar los límites de bloque y eliminar los artefactos de bloqueo del vídeo reconstruido. Si se desea, el filtro de desbloqueo típicamente filtraría la salida del sumador 62. Pueden además, usarse filtros adicionales (en bucle o bucle posterior) además del filtro de desbloqueo. Tales filtros no se muestran por brevedad, pero si se desea, pueden filtrar la salida del sumador 50 (como un filtro en bucle).

Durante el proceso de codificación, el codificador de vídeo 20 recibe una trama o segmento de vídeo a codificarse. El segmento o trama puede dividirse en múltiples bloques de vídeo. La unidad de estimación de movimiento 42 y la unidad de compensación de movimiento 44 realizan la codificación interpredictiva del bloque de vídeo recibido en relación con uno o más bloques en uno o más tramas de referencia para proporcionar predicción temporal. La unidad de intrapredicción 46 puede realizar alternativamente una codificación intrapredictiva del bloque de vídeo recibido en relación con uno o más bloques vecinos en la misma trama o segmento como el bloque a codificarse para proporcionar predicción espacial. El codificador de vídeo 20 puede realizar múltiples pasadas de codificación, por ejemplo, para seleccionar un modo de codificación apropiado para cada bloque de datos de vídeo.

Además, la unidad de partición 48 puede particionar bloques de datos de vídeo en subbloques, en base a la evaluación de esquemas de partición previos en pasadas de codificación previas. Por ejemplo, la unidad de partición 48 puede particionar inicialmente una trama o segmento en CTU y particionar cada una de las CTU en sub-CU en base al análisis de la tasa de distorsión (por ejemplo, optimización de la tasa de distorsión). La unidad de selección de modo 40 puede producir además, una estructura de datos de árboles cuádruples indicativa de la partición de una CTU en sub-CU. Las CU de los nodos de hoja del árbol cuaternario pueden incluir una o más PU y una o más TU.

La unidad de selección de modo 40 puede seleccionar uno de los modos de predicción, intra o inter, por ejemplo, en base a resultados de error, y proporciona el bloque predicho resultante al sumador 50 para generar datos residuales y al sumador 62 para reconstruir el bloque codificado para su uso como trama de referencia. La unidad de selección de modo 40 proporciona además, elementos de sintaxis, tales como vectores de movimiento, indicadores intramodo, información de partición y otra información de sintaxis similar, a la unidad de codificación de entropía 56.

La unidad de estimación de movimiento 42 y la unidad de compensación de movimiento 44 pueden estar muy integradas, pero se ilustran por separado para propósitos conceptuales. La estimación de movimiento, realizada por la unidad de estimación de movimiento 42, es el procedimiento de generar los vectores de movimiento, que estiman el movimiento para los bloques de vídeo. Un vector de movimiento, por ejemplo, puede indicar el desplazamiento de una PU de un bloque de vídeo dentro de una trama de vídeo actual o una imagen con relación a un bloque predictivo dentro de una trama de referencia (u otra unidad codificada) en relación con el bloque actual que se codifica dentro de la trama actual (u otra unidad codificada). Un bloque predictivo es un bloque que coincide estrechamente con el bloque que se va a codificar, en términos de diferencia de píxeles, que puede determinarse mediante la suma de la diferencia absoluta (SAD), la suma de la diferencia cuadrada (SSD) u otras métricas de diferencia. En algunos ejemplos, el codificador de vídeo 20 puede calcular valores para posiciones de píxeles subenteros de imágenes de referencia almacenadas en la memoria de imágenes de referencia 64. Por ejemplo, el codificador de vídeo 20 puede

interpolan valores de posiciones de un cuarto de píxel, posiciones de un octavo de píxel u otras posiciones de píxeles fraccionarios de la imagen de referencia. Por lo tanto, la unidad de estimación de movimiento 42 puede realizar una búsqueda de movimiento en relación con las posiciones de píxeles completos y a las posiciones de píxeles fraccionarios y generar un vector de movimiento con precisión de píxeles fraccionarios.

La unidad de estimación de movimiento 42 calcula un vector de movimiento para una PU de un bloque de vídeo en un segmento intercodificado comparando la posición de la PU con la posición de un bloque predictivo de una imagen de referencia. La imagen de referencia puede seleccionarse de una primera lista de imágenes de referencia (Lista 0) o de una segunda lista de imágenes de referencia (Lista 1), cada una de las cuales identifica una o más imágenes de referencia almacenadas en la memoria de imágenes de referencia 64. La unidad de estimación de movimiento 42 envía el vector de movimiento calculado a la unidad de codificación de entropía 56 y a la unidad de compensación de movimiento 44.

La compensación de movimiento, que se realiza mediante la unidad de compensación de movimiento 44, puede implicar buscar o generar el bloque predictivo en base al vector de movimiento que determina mediante la unidad de estimación de movimiento 42. De nuevo, la unidad de estimación de movimiento 42 y la unidad de compensación de movimiento 44 pueden integrarse funcionalmente, en algunos ejemplos. Al recibir el vector de movimiento para la PU del bloque de vídeo actual, la unidad de compensación de movimiento 44 puede localizar el bloque predictivo al que apunta el vector de movimiento en una de las listas de imágenes de referencia. El sumador 50 forma un bloque de vídeo residual mediante la sustracción de los valores de píxeles del bloque predictivo, de los valores de píxeles del bloque de vídeo actual que se está codificando, formando valores de diferencia de píxeles, como se discute más abajo. En general, la unidad de estimación de movimiento 42 realiza una estimación de movimiento en relación con los componentes de luminancia, y la unidad de compensación de movimiento 44 usa vectores de movimiento calculados en base a los componentes de luminancia para ambos componentes de crominancia y componentes de luminancia. La unidad de compensación de movimiento 44 puede generar un bloque de predicción mediante el uso de los vectores de movimiento, lo que puede incluir la interpolación o la manipulación matemática de otro modo de los valores de los bloques predictivos a los que hacen referencia los vectores de movimiento. La unidad de selección de modo 40 puede además, generar elementos de sintaxis que se asocian con los bloques de vídeo y el segmento de vídeo para su uso mediante el decodificador de vídeo 30 al decodificar los bloques de vídeo del segmento de vídeo.

El codificador de vídeo 20 puede configurarse para realizar cualquiera de las diversas técnicas de esta divulgación analizadas anteriormente con respecto a la Figura 6. Por ejemplo, la unidad de compensación de movimiento 44 puede configurarse para codificar información de movimiento para un bloque de datos de vídeo mediante el uso de AMVP o modo de fusión de acuerdo con HEVC, y/o puede configurarse para codificar información de movimiento afín o un bloque de datos de vídeo mediante el uso del modo inter afín o modo de fusión afín de acuerdo con las técnicas de esta divulgación.

La unidad de intrapredicción 46 puede intrapredecir un bloque actual, como una alternativa a la interpredicción realizada por la unidad de estimación de movimiento 42 y la unidad de compensación de movimiento 44, como se describió anteriormente. En particular, la unidad de intrapredicción 46 puede determinar un modo de intrapredicción a usar para codificar un bloque actual. En algunos ejemplos, la unidad de intrapredicción 46 puede codificar un bloque actual mediante el uso de diversos modos de intrapredicción, por ejemplo, durante pasadas de codificación separadas, y la unidad de intrapredicción 46 (o la unidad de selección de modo 40, en algunos ejemplos) puede seleccionar un modo de intrapredicción apropiado para usar entre los modos probados.

Por ejemplo, la unidad de intrapredicción 46 puede calcular los valores de tasa de distorsión mediante el uso de un análisis de la tasa de distorsión para los diversos modos de intrapredicción probados, y seleccionar el modo de intrapredicción que tiene las mejores características de tasa de distorsión entre los modos probados. El análisis de la tasa de distorsión generalmente determina una cantidad de distorsión (o error) entre un bloque codificado y un bloque no codificado original que se codificó para producir el bloque codificado, así como también una tasa de bits (es decir, una cantidad de bits) que se usa para producir el bloque codificado. La unidad de intrapredicción 46 puede calcular las relaciones de las distorsiones y tasas para diversos bloques codificados para determinar qué modo de intrapredicción exhibe el mejor valor de tasa de distorsión para el bloque.

Después de seleccionar un modo de intrapredicción para un bloque, la unidad de intrapredicción 46 puede proporcionar la información indicativa del modo de intrapredicción seleccionado para el bloque a la unidad de codificación de entropía 56. La unidad de codificación de entropía 56 puede codificar la información que indica el modo de intrapredicción que se selecciona. El codificador de vídeo 20 puede incluir en los datos de configuración del flujo de bits transmitido, que pueden incluir una pluralidad de tablas de índice en modo de intrapredicción y una pluralidad de tablas de índice en modo de intrapredicción modificadas (que se refieren además, a tablas de asignación de palabras de código), definiciones de contextos de codificación para diversos bloques e indicaciones de un modo de intrapredicción más probable, una tabla de índice de modo de intrapredicción y una tabla de índice de modo de intrapredicción modificada para usar en cada uno de los contextos.

El codificador de vídeo 20 forma un bloque de vídeo residual mediante la sustracción de los datos de predicción de



la unidad de selección de modo 40 del bloque de vídeo original que se está codificando. El sumador 50 representa el componente o componentes que realizan esta operación de sustracción. La unidad de procesamiento de transformación 52 aplica una transformación, tal como una transformada de coseno discreta (DCT) o una transformada conceptualmente similar, al bloque residual, lo que produce un bloque de vídeo que comprende valores de coeficiente de transformada. Podrían usarse transformadas wavelet, transformadas enteras, transformadas de subbanda, transformadas sinusoidales discretas (DST) u otros tipos de transformadas en lugar de una DCT. En cualquier caso, la unidad de procesamiento de transformación 52 aplica la transformada al bloque residual, lo que produce un bloque de coeficientes de transformada. La transformación puede convertir la información residual de un dominio de píxel en un dominio de transformación, tal como un dominio de frecuencia. La unidad de procesamiento de transformación 52 puede enviar los coeficientes de transformación resultantes a la unidad de cuantificación 54. La unidad de cuantificación 54 cuantifica los coeficientes de transformación para reducir además la tasa de bits. El procedimiento de cuantificación puede reducir la profundidad de bits asociada con algunos o todos los coeficientes. El grado de cuantificación puede modificarse mediante el ajuste de un parámetro de cuantificación.

Después de la cuantificación, la entropía de la unidad de codificación de entropía 56 codifica los coeficientes de transformación cuantificados. Por ejemplo, la unidad de codificación de entropía 56 puede realizar codificación de longitud variable adaptativa al contexto (CAVLC), codificación aritmética binaria adaptativa al contexto (CABAC), codificación aritmética binaria adaptativa al contexto en base a sintaxis (SBAC), codificación de entropía de partición de intervalo de probabilidad (PIPE) u otra técnica de codificación de entropía. En el caso de la codificación de entropía basada en el contexto, el contexto puede ser en base a los bloques vecinos. Después de la codificación de entropía mediante la unidad de codificación de entropía 56, el flujo de bits codificado puede transmitirse a otro dispositivo (por ejemplo, decodificador de vídeo 30) o archivarse para su posterior transmisión o recuperación.

La unidad de cuantificación inversa 58 y la unidad de transformación inversa 60 aplican la cuantificación inversa y la transformación inversa, respectivamente, para reconstruir el bloque residual en el dominio de píxel. En particular, el sumador 62 añade el bloque residual reconstruido al bloque de predicción compensado de movimiento producido anteriormente por la unidad de compensación de movimiento 44 o la unidad de intrapredicción 46 para producir un bloque de vídeo reconstruido para almacenamiento en la memoria de imágenes de referencia 64. El bloque de vídeo reconstruido puede usarse mediante la unidad de estimación de movimiento 42 y la unidad de compensación de movimiento 44 como un bloque de referencia para intercodificar un bloque en una trama de vídeo subsiguiente.

La Figura 8 es un diagrama de bloques que ilustra un ejemplo de decodificador de vídeo 30 que puede implementar técnicas de esta divulgación para decodificar información de movimiento de predicción afín. En el ejemplo de la Figura 8, el decodificador de vídeo 30 incluye una unidad de decodificación de entropía 70, una unidad de compensación de movimiento 72, una unidad de intrapredicción 74, una unidad de cuantificación inversa 76, una unidad de transformación inversa 78, una memoria de imágenes de referencia 82 y un sumador 80. El decodificador de vídeo 30 puede, en algunos ejemplos, realizar un pase de decodificación generalmente recíproco al pase de codificación descrito con respecto al codificador de vídeo 20 (Figura 7). La unidad de compensación de movimiento 72 puede generar datos de predicción en base a vectores de movimiento recibidos de la unidad de decodificación de entropía 70, mientras que la unidad de intrapredicción 74 puede generar datos de predicción en base a indicadores de modo de intrapredicción recibidos de la unidad de decodificación de entropía 70.

Durante el procedimiento de decodificación, 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 los elementos de sintaxis asociados del codificador de vídeo 20. La unidad de decodificación de entropía 70 del decodificador de vídeo 30 decodifica la entropía del flujo de bits para generar coeficientes cuantificados, vectores de movimiento o indicadores de modo de intrapredicción, y otros elementos de sintaxis. La unidad de decodificación de entropía 70 envía los vectores de movimiento y otros elementos de sintaxis a la unidad de compensación de movimiento 72. El decodificador de vídeo 30 puede recibir los elementos de sintaxis a nivel de segmento de vídeo y/o a nivel de bloque de vídeo.

Cuando el segmento de vídeo se codifica como un segmento intracodificado (I), la unidad de intra predicción 74 puede generar datos de predicción para un bloque de vídeo del segmento de vídeo actual en base a un modo de intrapredicción señalado y en datos de bloques decodificados previamente 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 produce bloques predictivos para un bloque de vídeo del segmento de vídeo actual en base a los vectores de movimiento y otros elementos de sintaxis que se reciben de la unidad de decodificación de entropía 70. Los bloques predictivos pueden producirse 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 tramas de referencia, Lista 0 y Lista 1, mediante el uso de técnicas de construcción predeterminadas en base a las imágenes de referencia almacenadas en la memoria de imágenes de referencia 82. La unidad de compensación de movimiento 72 determina la información de predicción para un bloque de vídeo del segmento de vídeo actual mediante el análisis de los vectores de movimiento y otros elementos de sintaxis, y usa la información de predicción para producir los bloques predictivos para el bloque de vídeo actual que se está decodificando. Por ejemplo, la unidad de compensación de movimiento 72 usa algunos de los elementos de sintaxis que se reciben para determinar un modo de predicción (por ejemplo, intra o interpredicción) que se usa para codificar los bloques de vídeo del segmento de

video, un tipo de segmento de interpredicción (por ejemplo, el segmento B o el segmento P), la información de construcción para una o más de las listas de imágenes de referencia para el segmento, vectores de movimiento para cada bloque de video intercodificado del segmento, el estado de interpredicción para cada bloque de video intercodificado del segmento, y otra información para decodificar los bloques de video en el segmento de video actual.

El decodificador de video 30 puede configurarse para realizar cualquiera de las diversas técnicas de esta divulgación analizadas anteriormente con respecto a la Figura 6. Por ejemplo, la unidad de compensación de movimiento 72 puede configurarse para realizar una predicción de vector de movimiento mediante el uso de la AMVP o el modo de fusión de acuerdo con HEVC, y/o puede configurarse para realizar información de movimiento afin o un bloque de datos de video mediante el uso del modo inter afin o el modo de fusión afin de acuerdo con las técnicas de esta divulgación. La unidad de decodificación de entropía 70 puede decodificar uno o más elementos de sintaxis que representan cómo se codifica la información de movimiento (por ejemplo, información de movimiento afin) para el bloque actual.

La unidad de compensación de movimiento 72 puede además, realizar una interpolación en base a filtros de interpolación. La unidad de compensación de movimiento 72 puede usar filtros de interpolación como se usan mediante el codificador de video 20 durante la codificación de los bloques de video para calcular valores interpolados para píxeles subenteros de bloques de referencia. En este caso, la unidad de compensación de movimiento 72 puede determinar los filtros de interpolación que se usan mediante el codificador de video 20 a partir de los elementos de sintaxis que se reciben y usar los filtros de interpolación para producir bloques predictivos.

La unidad de cuantificación inversa 76 cuantifica inversamente, es decir, descuantifica, los coeficientes de transformación cuantificados que se proporcionan en el flujo de bits y decodificados mediante la unidad de decodificación de entropía 70. El proceso de cuantificación inversa puede incluir el uso de un parámetro de cuantificación  $QP_Y$  que se calcula mediante el decodificador de video 30 para cada bloque de video en el segmento de video para determinar un grado de cuantificación y, asimismo, un grado de cuantificación inversa que debería aplicarse.

La unidad de transformación inversa 78 aplica una transformada inversa, por ejemplo, una DCT inversa, una transformada de entero inverso o un procedimiento de transformada inversa conceptualmente similar, a los coeficientes de transformación con el fin de producir bloques residuales en el dominio de píxeles.

Después de que la unidad de compensación de movimiento 72 genera el bloque predictivo para el bloque de video actual en base a los vectores de movimiento y otros elementos de sintaxis, el decodificador de video 30 forma un bloque de video decodificado que suma los bloques residuales de la unidad de transformación inversa 78 con los bloques predictivos correspondientes generado por la unidad de compensación de movimiento 72. El sumador 80 representa el componente o componentes que realizan esta operación de suma. Si se desea, también puede aplicarse un filtro de desbloqueo para filtrar los bloques decodificados con el fin de eliminar los artefactos de bloqueo. Pueden además, usarse otros filtros de bucle (ya sea en el bucle de codificación o después del bucle de codificación) para suavizar las transiciones de píxeles o mejorar de cualquier otro modo la calidad del video. Los bloques de video decodificados en una trama o imagen dadas se almacenan luego en la memoria de imágenes de referencia 82, que almacena las imágenes de referencia usadas para la compensación de movimiento subsiguiente. La memoria de imágenes de referencia 82 también almacena video decodificado para su posterior presentación en un dispositivo de visualización, tal como el dispositivo de visualización 32 de la Figura 6.

La Figura 11 es un diagrama de flujo que ilustra un procedimiento de ejemplo para codificar un bloque actual de datos de video de acuerdo con las técnicas de esta divulgación. A modo de ejemplo y explicación, se explica el procedimiento de la Figura 11 con respecto al codificador de video 20 de las Figuras 6 y 7. Sin embargo, debe entenderse que otros dispositivos pueden configurarse para realizar este procedimiento o uno similar.

Inicialmente, aunque no se muestra en la Figura 11, la unidad de selección de modo 40 puede determinar un modo de predicción que se usará para predecir un bloque actual. En este ejemplo, se supone que la unidad de selección de modo 40 selecciona un modo de predicción afin, que incluye la predicción mediante el uso de al menos dos vectores de movimiento. Por tanto, la unidad de selección de modo 40 hace que la unidad de estimación de movimiento 42 realice una búsqueda de movimiento para determinar un primer vector de movimiento y un segundo vector de movimiento (100). La unidad de selección de modo 40 puede comparar resultados de la tasa de distorsión entre una variedad de procedimientos de predicción, tales como intrapredicción, interpredicción y predicción afin, y determinar que la predicción afin da como resultado los mejores resultados de tasa de distorsión entre los diversos modos de predicción probados.

Después de que la unidad de estimación de movimiento 42 determina el primer y el segundo vector de movimiento, el codificador de video 20 puede calcular un primer predictor de vector de movimiento (MVP) (102) para el primer vector de movimiento. El primer MVP puede corresponder a un vector de movimiento de un bloque vecino. El codificador de video 20 puede entonces calcular una primera diferencia de vectores de movimiento (MVD) como la diferencia entre el primer vector de movimiento y el primer predictor de vector de movimiento (104). En particular, el

codificador de vídeo 20 puede calcular diferencias entre los componentes x e y del vector de movimiento y el MVD, respectivamente.

5 El codificador de vídeo 20 puede entonces determinar un segundo MVP (106) para el segundo vector de movimiento. El codificador de vídeo 20 puede calcular además una segunda MVD (108) como la diferencia entre el segundo vector de movimiento y la segunda MVD.

10 La unidad de compensación de movimiento 44 también puede predecir el bloque actual (110) mediante el uso de la predicción afin para generar un bloque de predicción para el bloque actual. Aunque se analizaron dos vectores de movimiento a modo de ejemplo, se debe entender que pueden usarse tres vectores de movimiento para la predicción afin para generar el bloque de predicción. Asimismo, el codificador de vídeo 20 puede generar una tercera MVD y un tercer MVP de acuerdo con las técnicas de esta divulgación para un tercer vector de movimiento, como se analizó anteriormente.

15 Después de generar el bloque de predicción, el codificador de vídeo 20 puede calcular un bloque residual (112) que representa las diferencias píxel por píxel entre el bloque actual y el bloque de predicción. En particular, el sumador 50 puede calcular las diferencias píxel por píxel entre el bloque actual y el bloque de predicción. El codificador de vídeo 20 puede luego codificar la primera MVD, la segunda MVD y el bloque residual (114) para codificar el bloque actual. Es decir, de acuerdo con las técnicas de esta divulgación, el codificador de vídeo 20 puede codificar la  
20 primera MVD mediante el uso de, por ejemplo, técnicas de predicción de vectores de movimiento, tales como el modo de fusión o el modo de AMVP, y codificar la segunda MVD prediciendo la segunda MVD a partir de la primera MVD. Por tanto, para codificar la segunda MVD, el codificador de vídeo 20 puede codificar datos representativos de una diferencia entre la primera MVD y la segunda MVD, tales como las diferencias entre los componentes x e y de la primera MVD y la segunda MVD. Para codificar el bloque residual, la unidad de procesamiento de transformación 52  
25 puede transformar el bloque residual, la unidad de cuantificación 54 puede cuantificar los coeficientes de transformación del bloque de transformación resultante, y la unidad de codificación de entropía 56 puede codificar por entropía los coeficientes de transformación cuantificados resultantes.

30 De esta manera, el procedimiento de la Figura 11 representa un ejemplo de un procedimiento que incluye codificar (es decir, codificar) una primera diferencia de vectores de movimiento (MVD) que representa una diferencia entre un primer vector de movimiento de un bloque actual de datos de vídeo predicho mediante el uso de la predicción afin y un primer predictor de vector de movimiento (MVP) para el primer vector de movimiento; predecir una segunda MVD a partir de la primera MVD para un segundo vector de movimiento del bloque actual; y codificar (es decir, codificar) el bloque actual mediante el uso de la predicción afin de acuerdo con el primer vector de movimiento y el segundo  
35 vector de movimiento. Al realizar el procedimiento de la Figura 11, el codificador de vídeo 20 puede generar un flujo de bits que es más eficiente en términos de ancho de banda que al realizar técnicas anteriores, porque los datos que representan la segunda MVD pueden ser más pequeños porque la segunda MVD se predice a partir de la primera MVD.

40 La Figura 12 es un diagrama de flujo que ilustra un procedimiento de ejemplo para decodificar un bloque actual de datos de vídeo de acuerdo con las técnicas de esta divulgación. El procedimiento de la Figura 12 se explica con respecto al decodificador de vídeo 30 de las Figuras 6 y 8 con fines de ejemplo. Sin embargo, debe entenderse que otros dispositivos pueden configurarse para realizar las técnicas de este procedimiento o uno similar.

45 El decodificador de vídeo 30 puede decodificar una primera diferencia de vectores de movimiento (MVD), una segunda MVD y un bloque residual (120) de un bloque actual. Es decir, de acuerdo con las técnicas de esta divulgación, el decodificador de vídeo 30 puede decodificar la primera MVD mediante el uso de, por ejemplo, técnicas de predicción de vectores de movimiento, tales como el modo de fusión o el modo de AMVP, y decodificar la segunda MVD prediciendo la segunda MVD a partir de la primera MVD. Por tanto, para decodificar la segunda  
50 MVD, el decodificador de vídeo 30 puede decodificar datos representativos de una diferencia entre la primera MVD y la segunda MVD, tales como las diferencias entre los componentes x e y de la primera MVD y la segunda MVD. Para decodificar el bloque residual, la unidad de decodificación de entropía 70 puede decodificar por entropía los coeficientes de transformación cuantificados, la unidad de cuantificación inversa 76 puede cuantificar inversamente los coeficientes de transformación, y la unidad de transformación inversa 78 puede transformar inversamente los  
55 coeficientes de transformación para reproducir el bloque residual.

La unidad de compensación de movimiento 72 puede entonces determinar un primer predictor de vector de movimiento (MVP) (122) para un primer vector de movimiento del bloque actual y calcular el primer vector de movimiento (124) a partir del primer MVP. En particular, la unidad de compensación de movimiento 72 puede añadir  
60 la primera MVD al primer MVP para calcular el primer vector de movimiento. La unidad de compensación de movimiento 72 puede determinar de manera similar un segundo MVP (126) para un segundo vector de movimiento del bloque actual y calcular el segundo vector de movimiento (128) a partir del segundo MVP. En particular, la unidad de compensación de movimiento 72 puede añadir la segunda MVD al segundo MVP para calcular el segundo vector de movimiento. En algunos ejemplos, puede incluirse un tercer vector de movimiento, en cuyo caso la unidad  
65 de decodificación de entropía 70 puede decodificar por entropía datos que representan una diferencia entre, por ejemplo, la primera MVD y una tercera MVD para el tercer vector de movimiento, y la unidad de compensación de

movimiento 72 puede calcular el tercer vector de movimiento a partir de la tercera MVD y un tercer MVP de manera similar.

La unidad de compensación de movimiento 72 puede entonces predecir el bloque actual (130), por ejemplo, de acuerdo con la predicción de movimiento afín mediante el uso del primer y segundo (y potencialmente tercer) vector de movimiento. El decodificador de vídeo 30 puede entonces decodificar el bloque actual (132), por ejemplo, haciendo que el sumador 80 añada los valores del bloque de predicción a los valores del bloque residual píxel por píxel.

De esta manera, el procedimiento de la Figura 12 representa un ejemplo de un procedimiento que incluye codificar (es decir, decodificar) una primera diferencia de vectores de movimiento (MVD) que representa una diferencia entre un primer vector de movimiento de un bloque actual de datos de vídeo predicho mediante el uso de la predicción afín y un primer predictor de vector de movimiento (MVP) para el primer vector de movimiento; predecir una segunda MVD a partir de la primera MVD para un segundo vector de movimiento del bloque actual; y codificar (es decir, decodificar) el bloque actual mediante el uso de la predicción afín de acuerdo con el primer vector de movimiento y el segundo vector de movimiento. Al realizar el procedimiento de la Figura 12, el decodificador de vídeo 30 puede decodificar un flujo de bits que es más eficiente en términos de ancho de banda que al realizar técnicas anteriores, porque los datos que representan la segunda MVD pueden ser más pequeños porque la segunda MVD se predice a partir de la primera MVD.

Debe reconocerse que, en función del ejemplo, ciertos actos o eventos de cualquiera de las técnicas que se describen en la presente memoria pueden realizarse en una secuencia diferente, pueden adicionarse, fusionarse u omitirse por completo (por ejemplo, no todos los actos o eventos que se describen son necesario para la práctica de las técnicas). Además, en ciertos ejemplos, los actos o eventos pueden realizarse concurrentemente, por ejemplo, a través de procesamiento de multihilos, procesamiento de interrupciones o múltiples procesadores, en lugar de secuencialmente.

En uno o más ejemplos, las funciones descritas pueden implementarse en hardware, software, microprograma o cualquiera de sus combinaciones. Si se implementan en software, las funciones pueden almacenarse en o transmitirse como una o más instrucciones o código en un medio legible por ordenador y que se ejecutan mediante una unidad de procesamiento basada en hardware. Los medios legibles por ordenador pueden incluir medios de almacenamiento legibles por ordenador, que corresponden a un medio tangible, tal como los medios de almacenamiento de datos o los medios de comunicación, que incluyen cualquier medio que facilite transferir un programa informático de un lugar a otro, por ejemplo, de acuerdo con un protocolo de comunicación. De esta manera, los medios legibles por ordenador generalmente pueden corresponder a (1) medios de almacenamiento legibles por ordenador tangibles que no son transitorios o (2) un medio de comunicación tal como una señal u onda portadora. Los medios de almacenamiento de datos pueden ser cualquier medio disponible al que se pueda acceder por 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.

A manera de ejemplo, y no de limitación, tales medios de almacenamiento legibles por ordenador pueden comprender RAM, ROM, EPROM, EEPROM, CD-ROM u otro almacenamiento en disco óptico, almacenamiento en disco magnético u otros dispositivos de almacenamiento magnéticos, memoria flash o cualquier otro medio que pueda utilizarse para almacenar el código del programa deseado en forma de instrucciones o estructuras de datos y al que puede accederse por un ordenador. Además, cualquier conexión se denomina apropiadamente como un medio legible por ordenador. Por ejemplo, si las instrucciones se transmiten desde un sitio web, servidor u otra fuente remota mediante el uso de un cable coaxial, un cable de fibra óptica, un par trenzado, una línea de suscriptor 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, debe entenderse que los medios de almacenamiento legibles por ordenador y los medios de almacenamiento de datos no incluyen conexiones, ondas portadoras, señales u otros medios transitorios, pero en cambio se dirigen a medios de almacenamiento tangibles y no transitorios. Disco magnético y disco óptico, como se usa en la presente memoria, incluye disco compacto (CD), disco láser, disco óptico, disco versátil digital (DVD), disquete y disco Blu-ray, donde los discos suelen reproducir datos magnéticamente, mientras que los discos reproducen datos ópticamente con láser. Las combinaciones de los medios anteriores también pueden incluirse dentro del ámbito de los medios legibles por ordenador.

Las instrucciones pueden ejecutarse mediante uno o más procesadores, tal como uno o más procesadores de señales digitales (DSP), microprocesadores de propósito general, circuitos integrados de aplicación específica (ASIC), matrices de puertas programables en campo (FPGA) u otros circuitos lógicos discretos o integrados equivalentes. En consecuencia, el término "procesador", como se usa en la presente memoria puede referirse a cualquiera de las estructuras anteriores o cualquier otra estructura adecuada para la implementación de las técnicas descritas en la presente memoria. Además, en algunos aspectos, la funcionalidad descrita en la presente memoria puede proporcionarse dentro de módulos de hardware y/o software dedicados configurados para la codificación y decodificación, o incorporarse en un códec combinado. Asimismo, las técnicas se podrían implementar

completamente en uno o más circuitos o elementos lógicos.

Las técnicas de esta divulgación pueden implementarse en una amplia variedad de dispositivos o aparatos, que incluyen un teléfono inalámbrico, un circuito integrado (IC) o un conjunto de IC (por ejemplo, un conjunto de chips). En esta divulgación se describen diversos componentes, módulos o unidades para enfatizar los aspectos funcionales de los dispositivos configurados para realizar las técnicas divulgadas, pero no necesariamente requieren la realización por diferentes unidades de hardware. Más bien, como se describió anteriormente, diversas unidades pueden combinarse en una unidad de hardware de códec o proporcionarse por una colección de unidades de hardware interoperativas, que incluyen uno o más procesadores como se describió anteriormente, junto con software y/o microprograma adecuados.

Se han descrito varios ejemplos.

## REIVINDICACIONES

1. Un procedimiento de decodificación de datos de vídeo, el procedimiento que comprende:

5 decodificar (120) una primera diferencia de vectores de movimiento, MVD, que representa una diferencia entre un primer vector de movimiento de un primer punto de control de un bloque actual de datos de vídeo predicho mediante el uso de la predicción afín y un primer predictor de vector de movimiento, MVP, para el primer vector de movimiento;  
 10 añadir (124) la primera MVD al primer MVP para reconstruir el primer vector de movimiento;  
 determinar (126) un segundo MVP para un segundo vector de movimiento de un segundo punto de control del bloque actual;  
 predecir una segunda MVD a partir de la primera MVD para el segundo vector de movimiento del bloque actual, representando la segunda MVD una diferencia entre el segundo vector de movimiento y el segundo MVP;  
 15 decodificar datos que representan MVD'2 para la segunda MVD, en el que MVD'2 representa un valor residual para la segunda MVD en relación con la primera MVD;  
 reconstruir la segunda MVD mediante el uso de MVD'2 y la primera MVD;  
 añadir (128) la segunda MVD al segundo MVP para reconstruir el segundo vector de movimiento; y  
 decodificar (132) el bloque actual mediante el uso de la predicción afín al: formar un bloque de predicción para el bloque actual mediante el uso del primer vector de movimiento y el segundo vector de movimiento,  
 20 decodificar un bloque residual para el bloque actual y añadir el bloque residual y el bloque de predicción para reconstruir el bloque actual,  
 en el que el primer vector de movimiento se origina en una esquina superior izquierda del bloque actual, y en el que el segundo vector de movimiento se origina en una esquina superior derecha del bloque actual.

2. El procedimiento de la reivindicación 1, en el que la primera MVD incluye un componente horizontal, MVDx1, y un componente vertical, MVDy1, el primer MVP incluye un componente horizontal, MVPx1, y un componente vertical, MVPy1, el primer vector de movimiento incluye un componente horizontal, MVx1, y un componente vertical, MVy1, MVDx1 = MVx1-MVPx1, y MVDy1 = MVy1-MVPy1.

3. El procedimiento de la reivindicación 1, en el que la decodificación del bloque actual comprende decodificar el bloque actual de acuerdo con un modelo afín de cuatro parámetros:

$$\begin{cases} V_x = ax + by + e \\ V_y = cx + dy + f \end{cases}$$

4. El procedimiento de la reivindicación 1, que comprende, además, predecir una tercera MVD a partir de al menos una de la primera MVD o la segunda MVD para un tercer vector de movimiento de un tercer punto de control del bloque actual, en el que la decodificación del bloque actual comprende decodificar el bloque actual mediante el uso de la predicción afín de acuerdo con el primer vector de movimiento, el segundo vector de movimiento y el tercer vector de movimiento, en el que la decodificación del bloque actual comprende decodificar el bloque actual de acuerdo con el siguiente modelo afín de seis parámetros, en el que w es un valor de ponderación:

$$\begin{cases} mv_x = \frac{(mv_{1x}-mv_{0x})}{w}x + \frac{(mv_{2x}-mv_{0x})}{w}y + mv_{0x} \\ mv_y = \frac{(mv_{1y}-mv_{0y})}{w}x + \frac{(mv_{2y}-mv_{0y})}{w}y + mv_{0y} \end{cases}$$

5. El procedimiento de la reivindicación 1, en el que los puntos de control se:

- i) determinan mediante la decodificación de datos que definen los puntos de control para el primer vector de movimiento y el segundo vector de movimiento, o  
 ii) determinan para el primer vector de movimiento y el segundo vector de movimiento en base a una forma del bloque actual, o  
 iii) derivada implícitamente para el primer vector de movimiento y el segundo vector de movimiento.

6. El procedimiento de la reivindicación 1, en el que:

- i) la primera MVD comprende MVD 1, la segunda MVD comprende MVD2, MVD'2 = MVD1-w\*MVD2, y w comprende un valor de ponderación; o  
 ii) la primera MVD comprende MVD1, la segunda MVD comprende MVD2, y MVD'2 = MVD2-((MVD1+1)>>1) para un valor de ponderación de 0,5; o bien  
 iii) la primera MVD comprende MVD1, la segunda MVD comprende MVD2, y MVD'2 = MVD2-((MVD1+2)>>2) para un valor de ponderación de 0,25.

7. El procedimiento de la reivindicación 1, que comprende, además, determinar predecir la segunda MVD a partir de la primera MVD, en el que predecir la segunda MVD a partir de la primera MVD comprende predecir la segunda MVD a partir de la primera MVD en respuesta a determinar predecir la segunda MVD a partir de la primera MVD, en el que determinar predecir la segunda MVD a partir de la primera MVD comprende:

- i) determinar predecir la segunda MVD a partir de la primera MVD en base a una forma del bloque actual, o
- ii) determinar predecir la segunda MVD a partir de la primera MVD en base a un procedimiento de predicción de movimiento para el bloque actual, en el que determinar predecir la segunda MVD a partir de la primera MVD comprende determinar que el procedimiento de predicción de movimiento es un modo de fusión afín.

8. El procedimiento de la reivindicación 1, en el que la segunda MVD incluye un componente horizontal,  $MVD2^x$ , y un componente vertical,  $MVD2^y$ , y en el que predecir la segunda MVD comprende predecir  $MVD2^x$  de manera diferente a predecir  $MVD2^y$ .

9. El procedimiento de la reivindicación 1, que comprende además generar un tercer MVP para un tercer vector de movimiento de un tercer punto de control del bloque actual a partir de al menos una de la primera MVD o la segunda MVD.

10. Un procedimiento de codificación de datos de vídeo, el procedimiento que comprende:

determinar un primer predictor de vector de movimiento, MVP, para un primer vector de movimiento de un punto de control de un bloque actual de datos de vídeo;  
 restar el primer MVP del primer vector de movimiento para generar una primera diferencia de vectores de movimiento, MVD;  
 codificar la primera MVD;  
 determinar un segundo MVP para un segundo vector de movimiento de un segundo punto de control del bloque actual;  
 restar el segundo MVP del segundo vector de movimiento para generar una segunda MVD para el segundo vector de movimiento;  
 codificar datos que representan  $MVD'2$  para la segunda MVD, en el que  $MVD'2$  representa un valor residual para la segunda MVD cuando se predice mediante el uso de la primera MVD; y  
 codificar el bloque actual mediante el uso de la predicción afín al: formar un bloque de predicción para el bloque actual mediante el uso del primer vector de movimiento y el segundo vector de movimiento, restar el bloque de predicción del bloque actual para generar un bloque residual, y codificar el bloque residual, en el que el primer vector de movimiento se origina en una esquina superior izquierda del bloque actual, y en el que el segundo vector de movimiento se origina en una esquina superior derecha del bloque actual.

11. El procedimiento de la reivindicación 10, en el que:

- i) la primera MVD comprende  $MVD1$ , la segunda MVD comprende  $MVD2$ ,  $MVD'2 = MVD1 - w * MVD2$ , y  $w$  comprende un valor de ponderación; o
- ii) la primera MVD comprende  $MVD1$ , la segunda MVD comprende  $MVD2$ , y  $MVD'2 = MVD2 - ((MVD1 + 1) >> 1)$  para un valor de ponderación de 0,5; o bien
- iii) la primera MVD comprende  $MVD1$ , la segunda MVD comprende  $MVD2$ , y  $MVD'2 = MVD2 - ((MVD1 + 2) >> 2)$  para un valor de ponderación de 0,25.

12. Un dispositivo para decodificar datos de vídeo, el dispositivo que comprende:

una memoria configurada para almacenar datos de vídeo; y  
 uno o más procesadores implementados en circuitos y configurados para:

decodificar (120) una primera diferencia de vectores de movimiento, MVD, que representa una diferencia entre un primer vector de movimiento de un primer punto de control de un bloque actual de datos de vídeo predicho mediante el uso de la predicción afín y un primer predictor de vector de movimiento, MVP, para el primer vector de movimiento;  
 añadir (124) la primera MVD al primer MVP para reconstruir el primer vector de movimiento;  
 determinar (126) un segundo MVP para un segundo vector de movimiento de un segundo punto de control del bloque actual  
 predecir una segunda MVD a partir de la primera MVD para el segundo vector de movimiento del bloque actual, representando la segunda MVD una diferencia entre el segundo vector de movimiento y el segundo MVP;  
 decodificar datos que representan  $MVD'2$  para la segunda MVD, en el que  $MVD'2$  representa un valor residual para la segunda MVD en relación con la primera MVD;  
 reconstruir la segunda MVD mediante el uso de  $MVD'2$  y la predicción de la primera MVD;  
 añadir (128) la segunda MVD al segundo MVP para reconstruir el segundo vector de movimiento; y  
 decodificar (132) el bloque actual mediante el uso de la predicción afín al: formar un bloque de predicción

para el bloque actual mediante el uso del primer vector de movimiento y el segundo vector de movimiento, decodificar un bloque residual para el bloque actual y añadir el bloque residual y el bloque de predicción para reconstruir el bloque actual,  
 en el que el primer vector de movimiento se origina en una esquina superior izquierda del bloque actual, y  
 en el que el segundo vector de movimiento se origina en una esquina superior derecha del bloque actual.

13. El dispositivo de la reivindicación 12, en el que:

- i) la primera MVD comprende MVD1, la segunda MVD comprende MVD2,  $MVD'2 = MVD1 - w * MVD2$ , y w comprende un valor de ponderación; o
- ii) la primera MVD comprende MVD1, la segunda MVD comprende MVD2, y  $MVD'2 = MVD2 - ((MVD1 + 1) >> 1)$  para un valor de ponderación de 0,5; o bien
- iii) la primera MVD comprende MVD1, la segunda MVD comprende MVD2, y  $MVD'2 = MVD2 - ((MVD1 + 2) >> 2)$  para un valor de ponderación de 0,25.

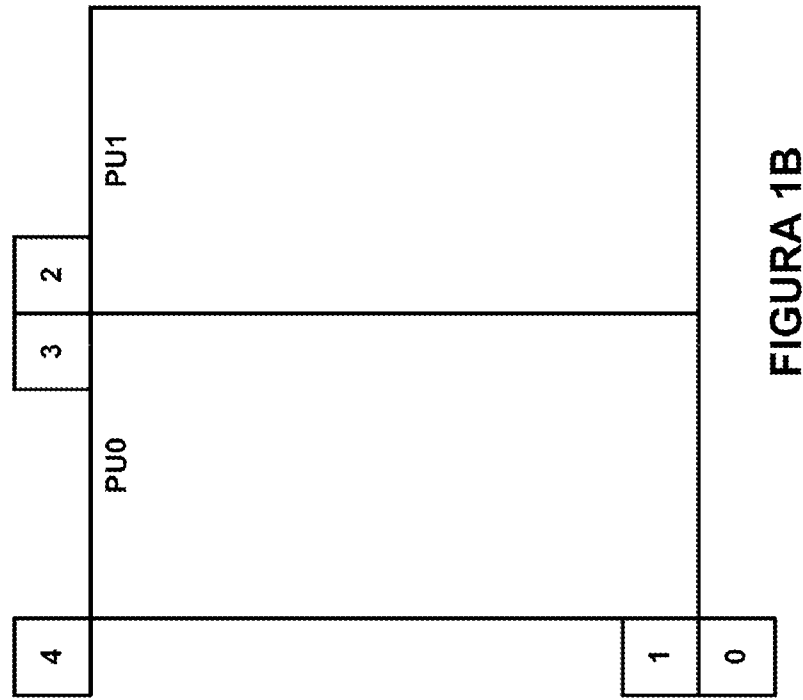
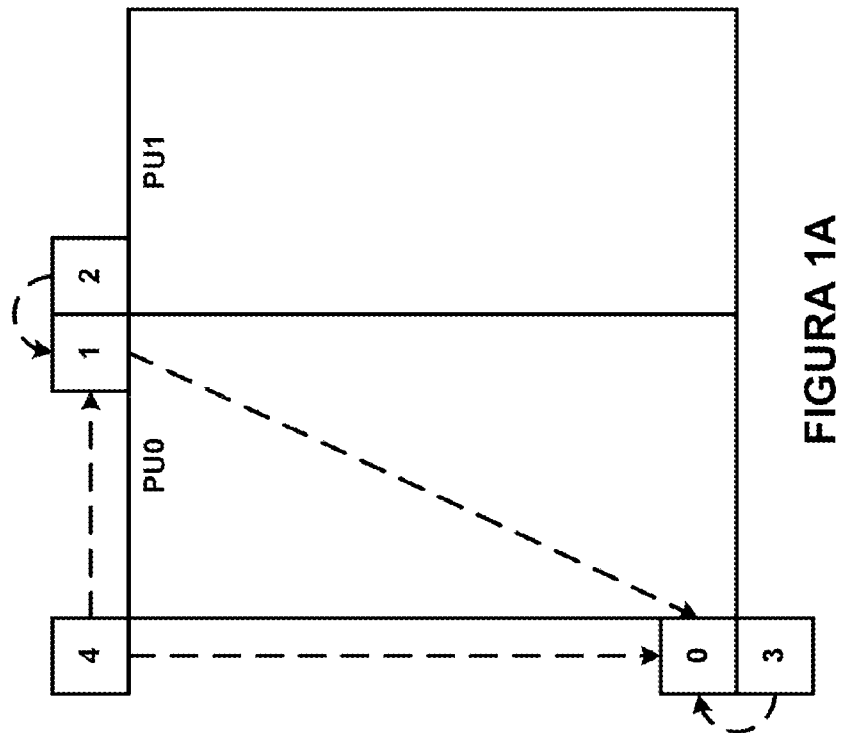
14. Un dispositivo para codificar datos de vídeo, el dispositivo que comprende:

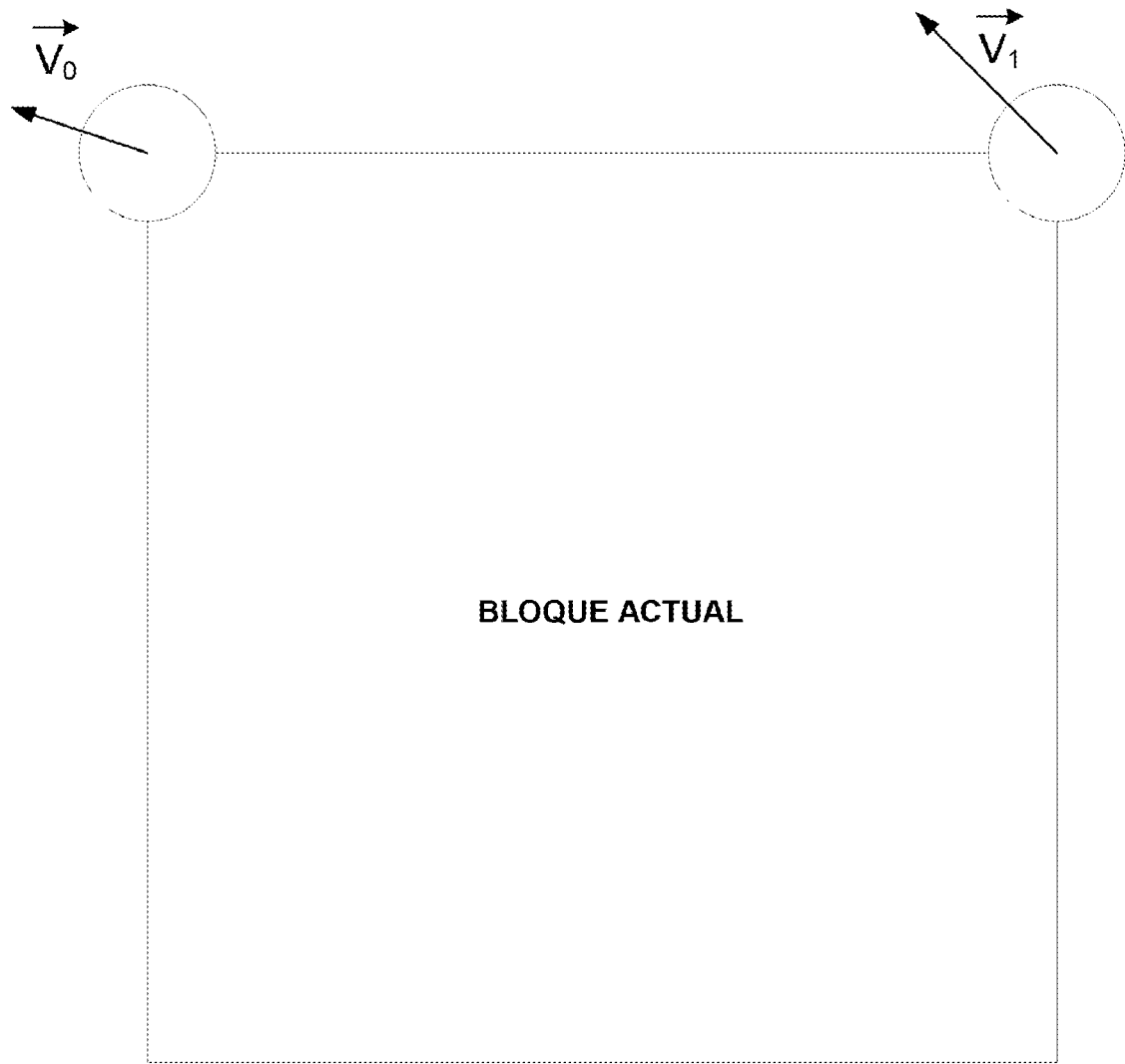
una memoria configurada para almacenar datos de vídeo; y  
 uno o más procesadores implementados en circuitos y configurados para:

determinar un primer predictor de vector de movimiento, MVP, para un primer vector de movimiento de un punto de control de un bloque actual de datos de vídeo;  
 restar el primer MVP del primer vector de movimiento para generar una primera diferencia de vectores de movimiento, MVD;  
 codificar la primera MVD;  
 determinar un segundo MVP para un segundo vector de movimiento de un segundo punto de control del bloque actual;  
 restar el segundo MVP del segundo vector de movimiento para generar una segunda MVD para el segundo vector de movimiento;  
 codificar datos que representan MVD'2 para la segunda MVD, en el que MVD'2 representa un valor residual para la segunda MVD cuando se predice mediante el uso de la primera MVD; y  
 codificar el bloque actual mediante el uso de la predicción afín al: formar un bloque de predicción para el bloque actual mediante el uso del primer vector de movimiento y el segundo vector de movimiento, restar el bloque de predicción del bloque actual para generar un bloque residual, y codificar el bloque residual,  
 en el que el primer vector de movimiento se origina en una esquina superior izquierda del bloque actual, y  
 en el que el segundo vector de movimiento se origina en una esquina superior derecha del bloque actual.

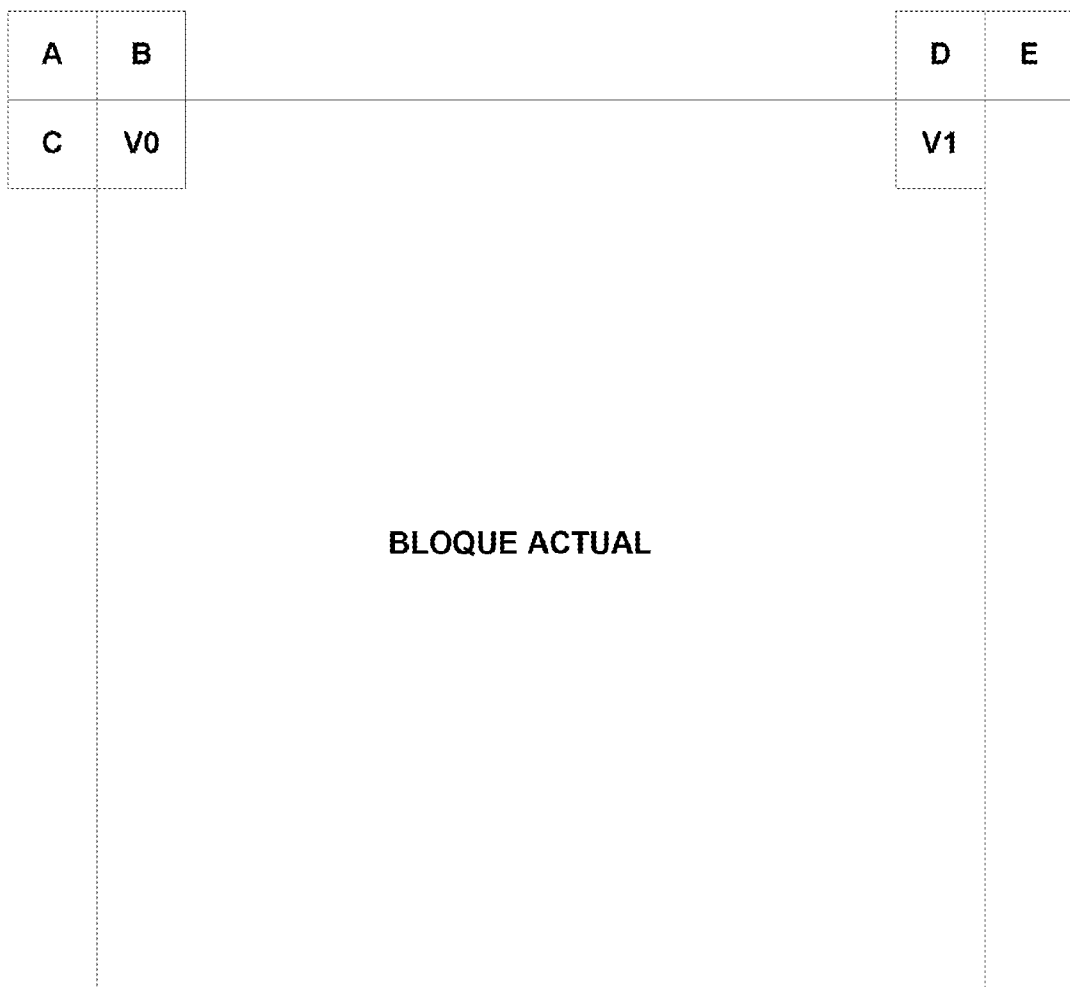
15. Un medio de almacenamiento legible por ordenador que tiene almacenadas en el mismo instrucciones que, cuando se ejecutan por un procesador de un dispositivo, hacen que un procesador lleve a cabo un procedimiento de acuerdo con cualquiera de las reivindicaciones 1 a 11.







**FIGURA 2**



**FIGURA 3**

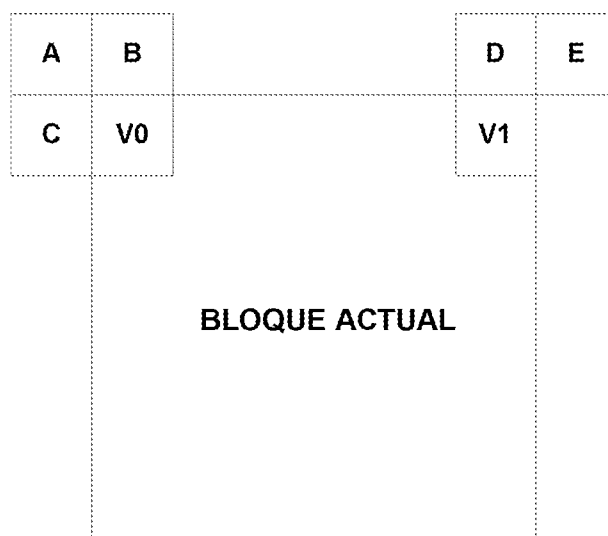


FIGURA 4A

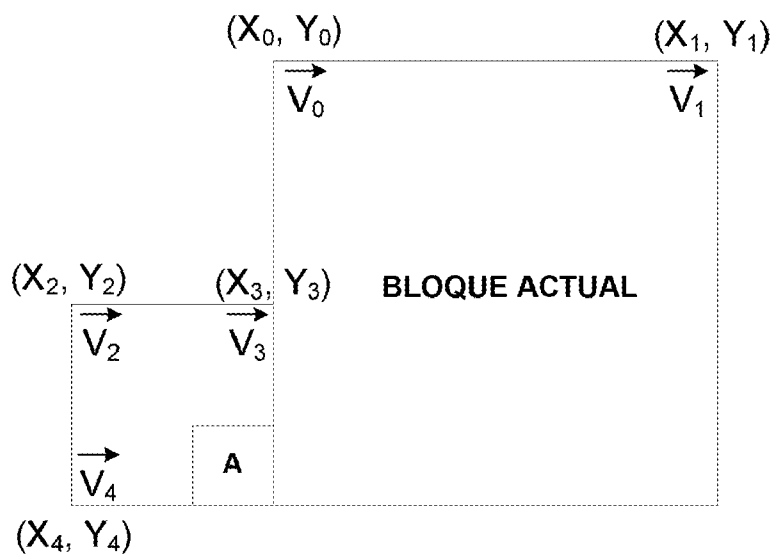


FIGURA 4B

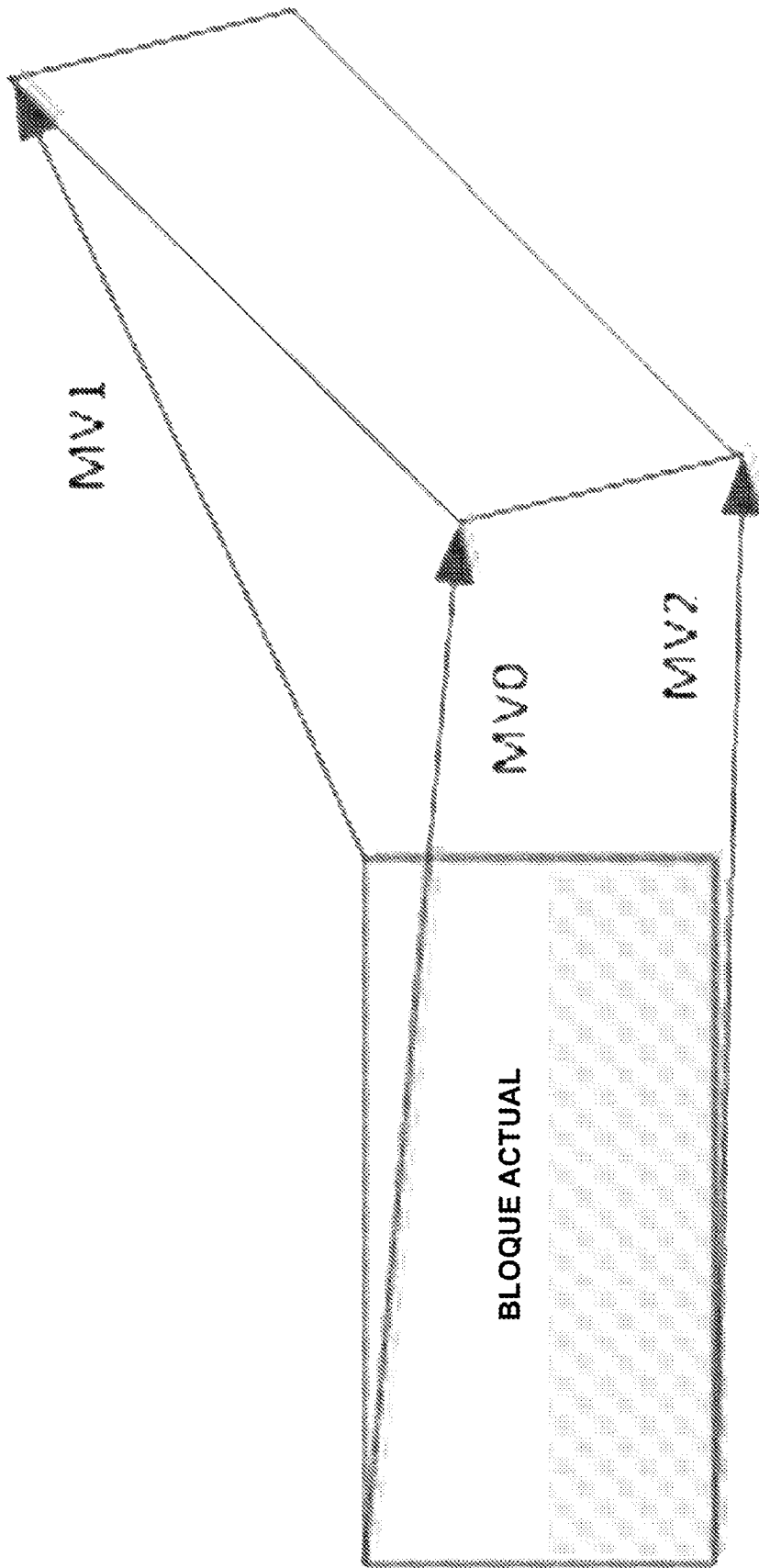


FIGURA 5

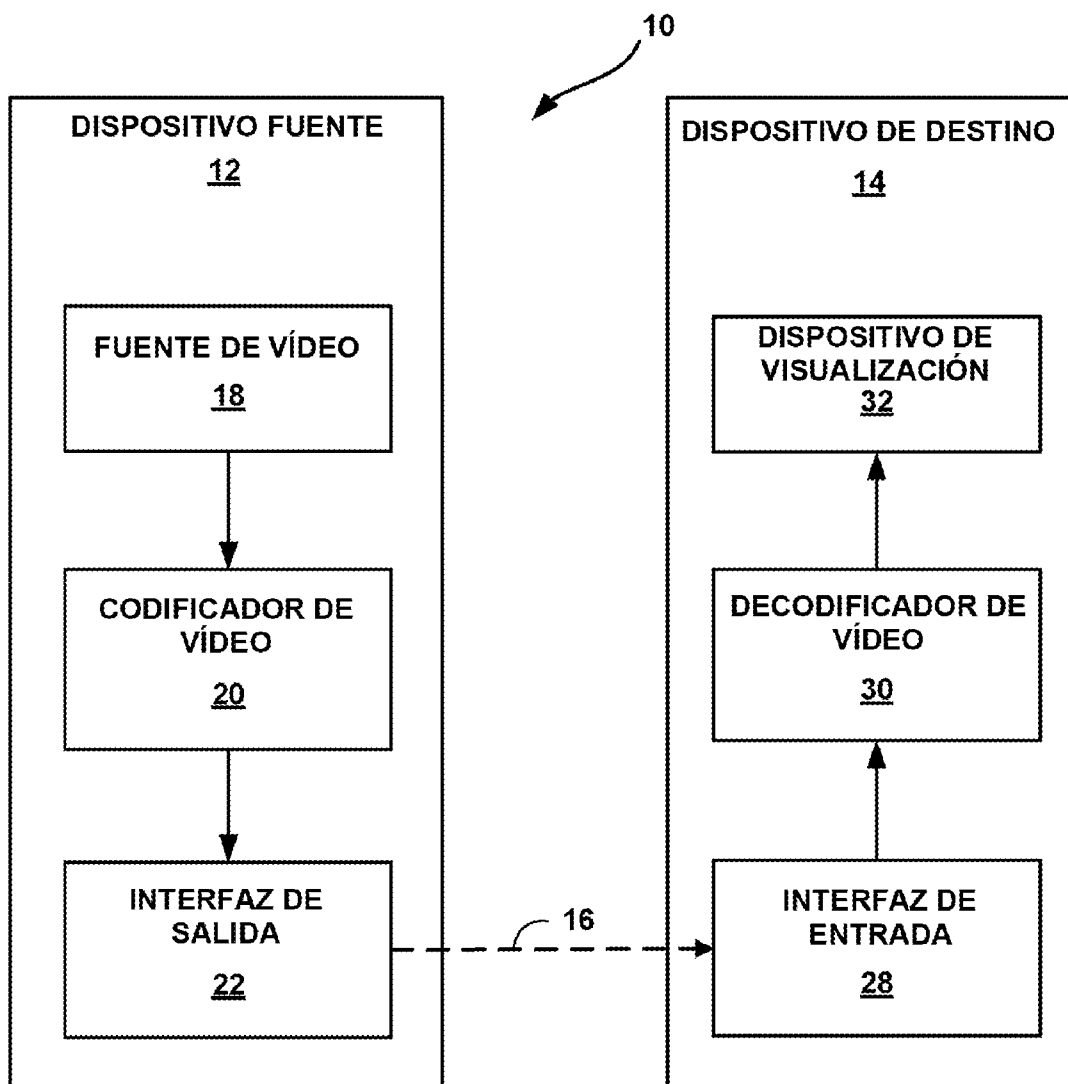


FIGURA 6

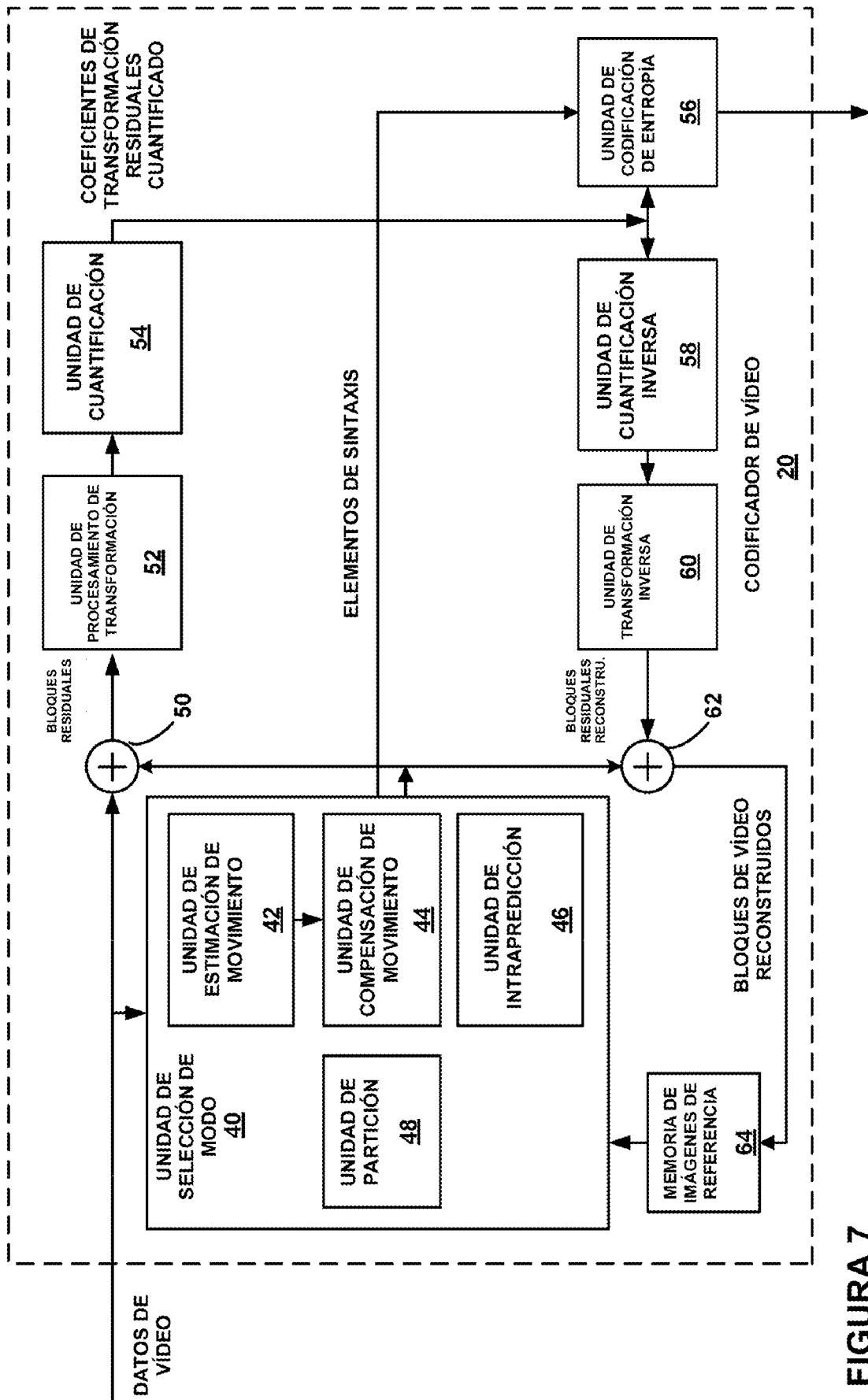


FIGURA 7

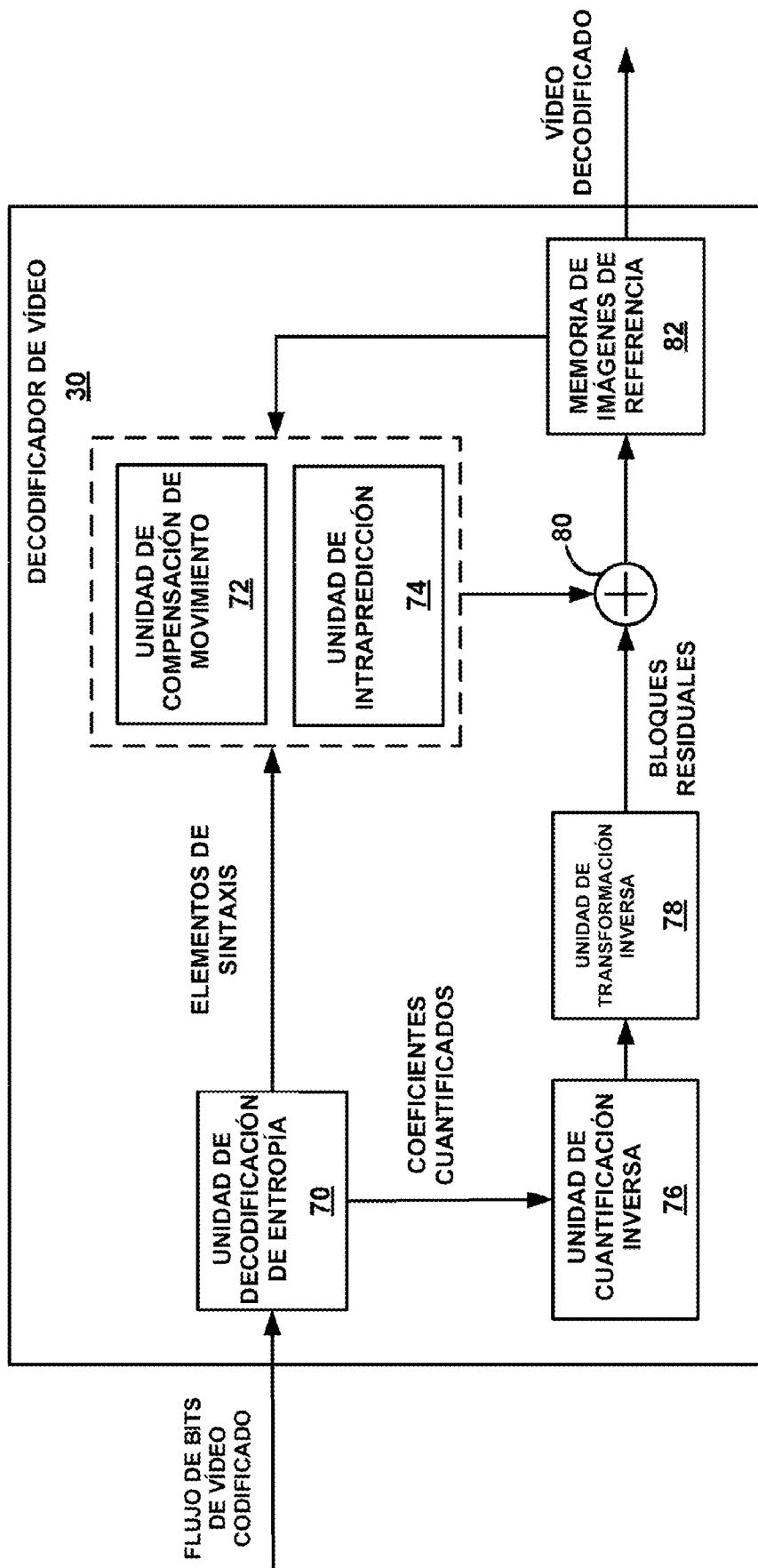


FIGURA 8



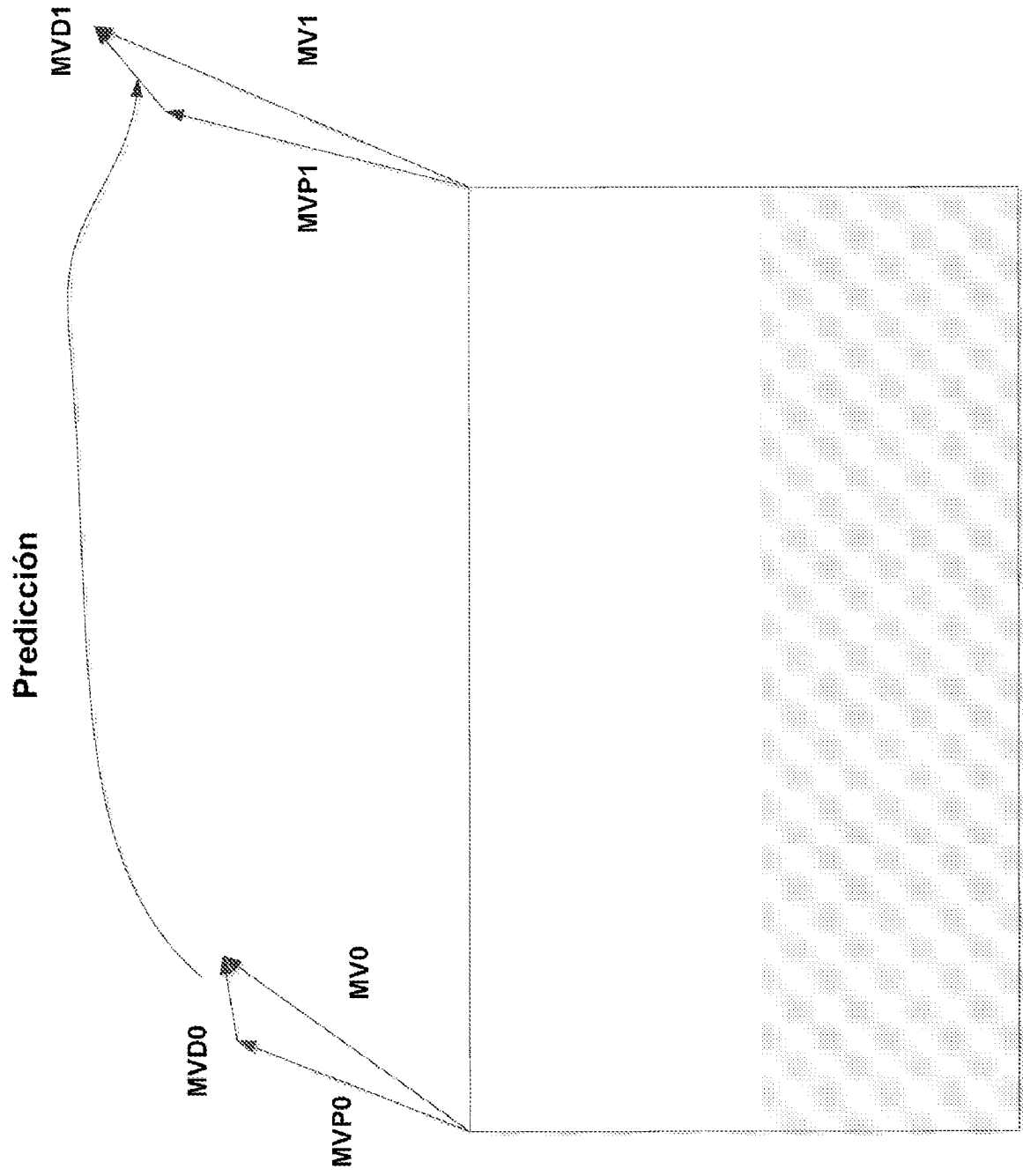


FIGURA 9

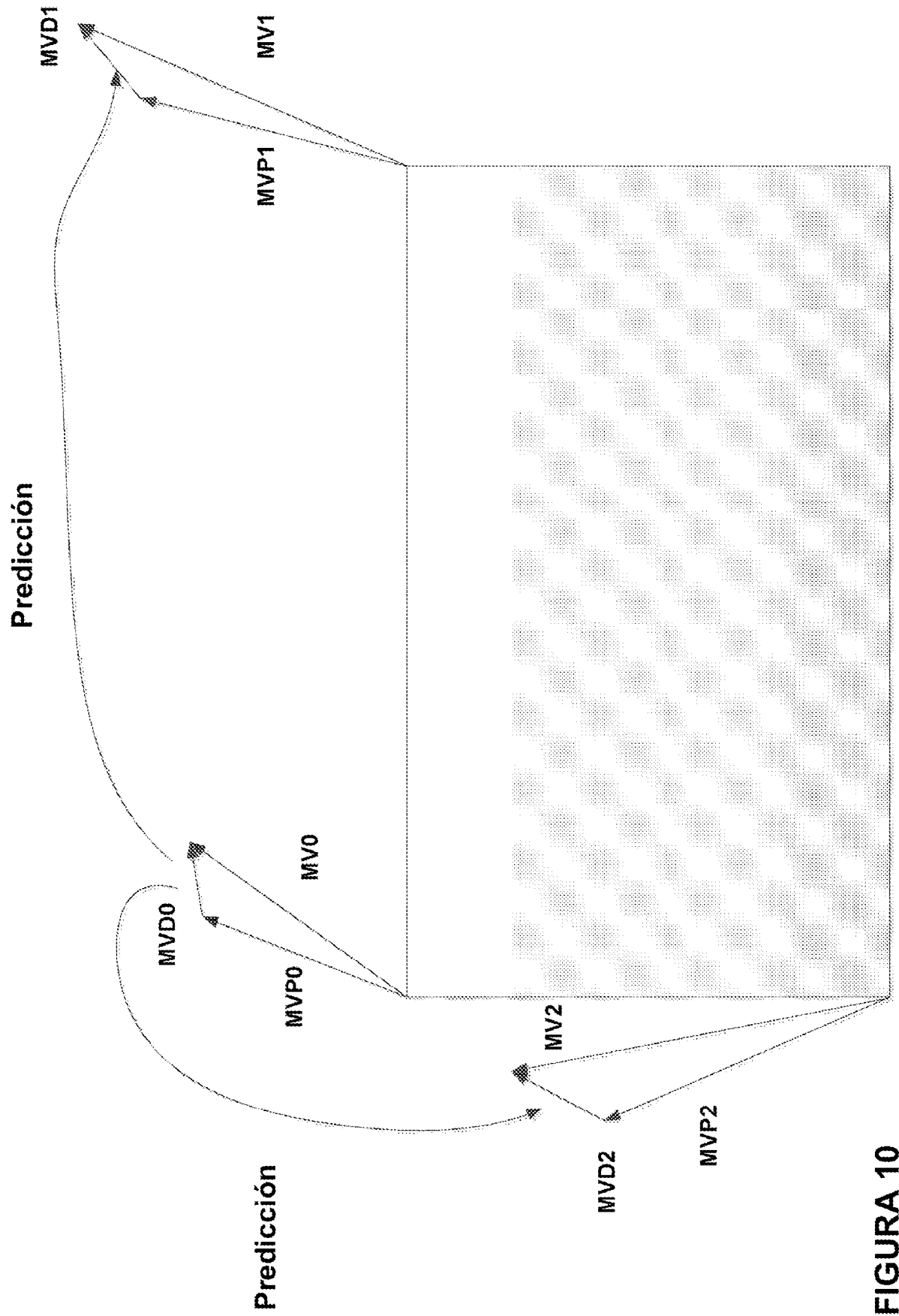


FIGURA 10

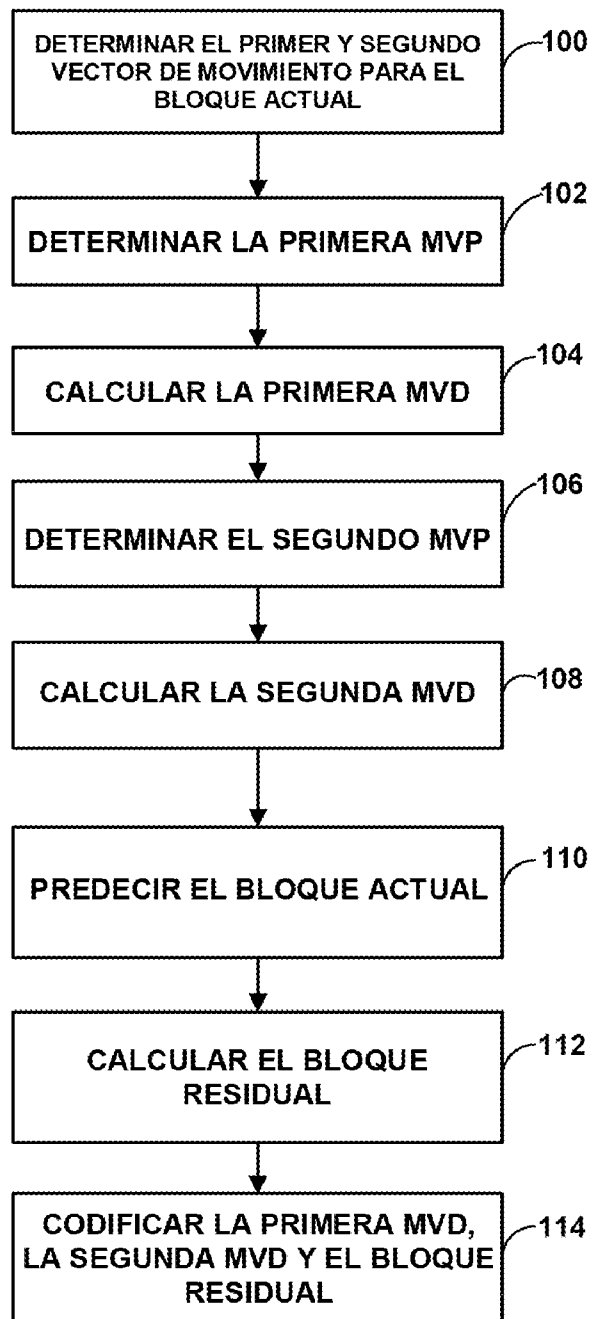


FIGURA 11

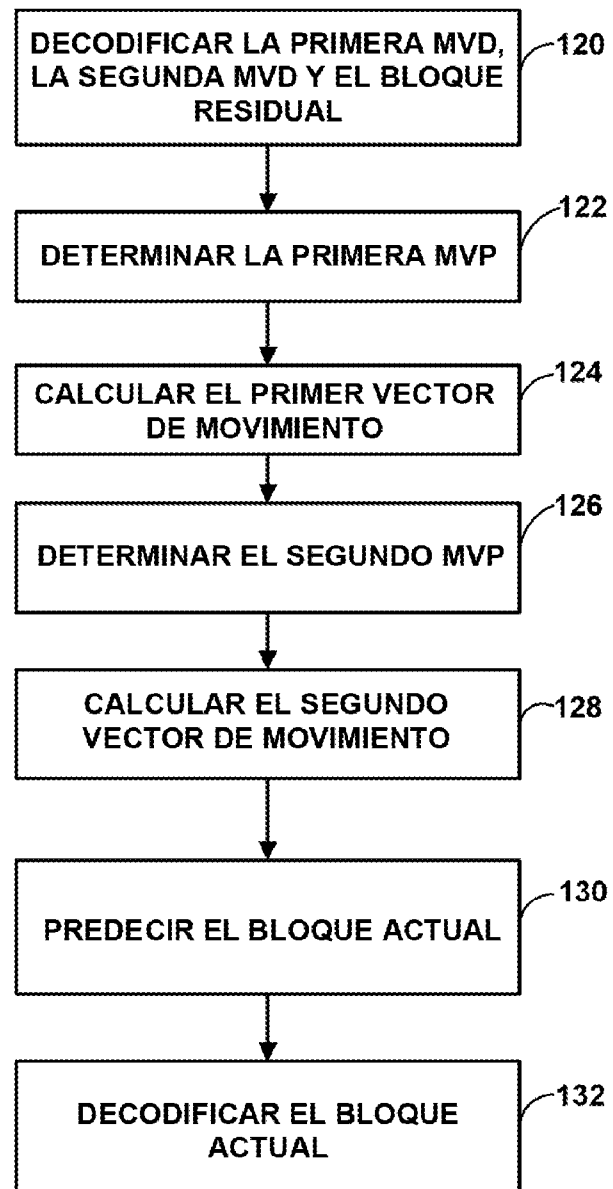


FIGURA 12