

## **CODIFICACIÓN DE INFORMACIÓN DE MOVIMIENTO DE PREDICCIÓN AFÍN PARA CODIFICACIÓN DE VIDEO**

**[0001]** La presente solicitud reivindica el beneficio de la solicitud provisional estadounidense N.º 62/567.598, presentada el 3 de octubre de 2017, y la solicitud estadounidense N.º 16/148.738, presentada el 1 de octubre de 2018, cuyos contenidos completos se incorporan en la presente por referencia.

### **CAMPO TÉCNICO**

**[0002]** La presente divulgación se refiere a la codificación de video, y más en particular, a la codificación de información de movimiento de datos de video.

### **ANTECEDENTES**

**[0003]** Las capacidades de video digital se pueden incorporar a un amplio rango de dispositivos, que incluyen televisores digitales, sistemas digitales de emisión directa, sistemas de emisión inalámbricos, asistentes digitales personales (PDA), computadoras portátiles o de escritorio, tablets, libros electrónicos, cámaras digitales, dispositivos de grabación digitales, transmisores de medios digitales, dispositivos de videojuegos, consolas de videojuegos, radioteléfonos celulares o satelitales, denominados “smartphones”, dispositivos de videoteleconferencia, dispositivos de transmisión de video, y similares. Los dispositivos de video digital implementan técnicas de codificación de video, tales como aquellas descritas en los estándares definidos por MPEG-2, MPEG-4, ITU-T H.263, ITU-T H.264/MPEG-4, parte 10, codificación de video avanzada (AVC), el estándar de codificación de video de alta eficiencia (HEVC), ITU-T H.265/estándar de codificación de video de alta eficiencia (HEVC) y las extensiones de estos estándares. Los dispositivos de video pueden transmitir, recibir, codificar, decodificar y/o almacenar información de video digital de manera más eficaz implementando las técnicas de codificación de video.

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

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

**[0005]** La predicción espacial o temporal da como resultado un bloque predictivo para un bloque que se desea codificar. Los datos residuales representan las diferencias de píxeles entre el bloque original que se desea codificar y el bloque predictivo. Un bloque intercodificado se codifica de acuerdo con un vector de movimiento que apunta hacia un bloque de muestras de referencia que forman el bloque predictivo, y los datos residuales que indican la diferencia entre el bloque codificado y el bloque predictivo. Un bloque intracodificado se codifica de acuerdo con un modo de intracodificación y los datos residuales. Para una mayor compresión, los datos residuales se pueden transformar del dominio de píxeles a un dominio de la transformada, lo que da como resultado coeficientes de la transformada residuales, que luego se pueden cuantificar. Los coeficientes de la transformada cuantificados, inicialmente dispuestos en una matriz de dos dimensiones, se pueden escanear para producir un vector unidimensional de coeficientes de la transformada, y se puede aplicar codificación entrópica para alcanzar una compresión aún mayor.

## **SÍNTESIS**

**[0006]** En general, la presente divulgación describe técnicas que se refieren a la predicción interimagen. Por ejemplo, las técnicas de la presente divulgación incluyen codificación (codificación y/o decodificación) de vectores de movimiento para compensación de movimiento afín en codificación de video en bloques. Estas técnicas se pueden aplicar a estándares de decodificación de video existentes y/o futuros.

**[0007]** En un ejemplo, un método para codificar (por ejemplo, codificar o decodificar) datos de video incluye 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 video predicho usando predicción afín y un primer predictor de vectores de movimiento (MVP) para el primer vector de movimiento, predecir una segunda MVD de la primera MVD para un segundo vector de movimiento del bloque actual, y codificar el bloque actual usando predicción afín de acuerdo con el primer vector de movimiento y el segundo vector de movimiento.

**[0008]** En otro ejemplo, un dispositivo para codificar (por ejemplo, codificar o decodificar) datos de video incluye una memoria configurada para almacenar datos de

video, y uno o más procesadores implementados en circuito y configurados para 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 video predicho usando predicción afín y un primer predictor de vectores de movimiento (MVP) para el primer vector de movimiento, predecir una segunda MVD de la primera MVD para un segundo vector de movimiento del bloque actual, y codificar el bloque actual usando predicción afín de acuerdo con el primer vector de movimiento y el segundo vector de movimiento.

**[0009]** En otro ejemplo, un medio de almacenamiento legible por computadora almacenó allí instrucciones que, cuando se ejecutan, generan que un procesador codifique 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 video predicha usando predicción afín y un primer predictor de vectores de movimiento (MVP) para el primer vector de movimiento, predecir una segunda MVD de la primera MVD para un segundo vector de movimiento del bloque actual, y codificar el bloque actual usando predicción afín de acuerdo con el primer vector de movimiento y el segundo vector de movimiento.

**[0010]** En otro ejemplo, un dispositivo para codificar (por ejemplo, codificar o decodificar) datos de video incluye medios para 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 video predicho usando predicción afín y un primer predictor de vectores de movimiento (MVP) para el primer vector de movimiento, medios para predecir una segunda MVD de la primera MVD para un segundo vector de movimiento del bloque actual, y medios para codificar el bloque actual usando predicción afín de acuerdo con el primer vector de movimiento y el segundo vector de movimiento.

**[0011]** Los detalles de uno o más ejemplos se indican en los dibujos adjuntos y la descripción a continuación. Otras características, objetos y ventajas serán evidentes de la descripción y dibujos y de las reivindicaciones.

### **BREVE DESCRIPCIÓN DE LOS DIBUJOS**

**[0012]** Las Figuras 1A y 1B son diagramas conceptuales que ilustran ejemplos de candidatos vecinos espaciales para los modos de fusión y de predicción de vectores de movimiento avanzada (AMVP) de codificación de video de alta eficiencia (HEVC).

**[0013]** La Figura 2 es un diagrama conceptual que ilustra un vector de movimiento de dos puntos de ejemplo que es afín con cuatro parámetros afines para un bloque

actual.

**[0014]** La Figura 3 es un diagrama conceptual que ilustra un ejemplo de modo de interpredicción afín para un bloque actual.

**[0015]** Las Figuras 4A y 4B son diagramas conceptuales que ilustran un ejemplo de modo de fusión afín para un bloque actual.

**[0016]** La Figura 5 es un diagrama conceptual que ilustra un modelo afín de ejemplo con seis parámetros (tres vectores de movimiento).

**[0017]** La Figura 6 es un diagrama de bloques que ilustra un sistema de codificación y decodificación de video de ejemplo que puede usar las técnicas de la presente divulgación para codificar de manera eficaz la información de movimiento para predicción afín.

**[0018]** La Figura 7 es un diagrama de bloques que ilustra un ejemplo de un codificador de video que puede implementar las técnicas de la presente divulgación para codificar información de movimiento de predicción afín.

**[0019]** La Figura 8 es un diagrama de bloques que ilustra un ejemplo de decodificador de video 30 que puede implementar las técnicas de la presente divulgación para decodificar información de movimiento de predicción afín.

**[0020]** La Figura 9 es un diagrama conceptual que ilustra un ejemplo de predicción de diferencia de vectores de movimiento (MVD) para predicción de información de movimiento afín.

**[0021]** La Figura 10 es un diagrama conceptual que ilustra un ejemplo de predicción de MVD para predicción afín con tres vectores de movimiento (predicción afín de seis parámetros).

**[0022]** La Figura 11 es un diagrama de flujo que ilustra un método de ejemplo para codificar un bloque actual de datos de video de acuerdo con las técnicas de la presente divulgación.

**[0023]** La Figura 12 es un diagrama de flujo que ilustra un método de ejemplo para decodificar un bloque actual de datos de video de acuerdo con las técnicas de la presente divulgación.

#### **DESCRIPCIÓN DETALLADA**

**[0024]** Los estándares de codificación de video 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 de codificación de video escalable (SVC) y codificación de video multivisión (MVC).

**[0025]** Recientemente, el diseño de un nuevo estándar de codificación de video, a saber ITU-T H.265/codificación de video de alta eficacia (HEVC), fue terminado por el Equipo de Colaboración Conjunta sobre Codificación de Video (JCT-VC) de ITU-T del Grupo de Expertos en Codificación de Video (VCEG) y el ISO/IEC Grupo de Expertos en Imágenes en Movimiento (MPEG). La especificación borrador de HEVC más reciente, en adelante denominada HEVC WD, está disponible de [phenix.int-evry.fr/jct/doc\\_end\\_user/documents/15\\_Geneva/wg11/JCTVC-O1003-v2.zip](http://phenix.int-evry.fr/jct/doc_end_user/documents/15_Geneva/wg11/JCTVC-O1003-v2.zip). Las extensiones de rango para HEVC, a saber HEVC-Rext, también están siendo desarrolladas por el JCT-VC. Un borrador en progreso (WD) de extensiones de rango, en adelante denominado RExt WD6, está disponible de [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).

**[0026]** Las investigaciones de nuevas herramientas de codificación para codificación de video futura están en curso (por ejemplo, como se estudió en JVET-Equipo de Exploración de Video Conjunta), y se propusieron tecnologías que mejoran la eficacia de codificación para codificación de video. Existe evidencia de que se pueden obtener mejoras significativas en cuanto a eficacia de codificación al aprovechar las características de contenido de video, en especial para contenido de alta resolución tal como 4K, con novedosas herramientas de codificación dedicadas más allá de H.265/HEVC. Se ha invitado a las empresas y organizaciones a que presenten propuestas en la etapa de exploración para mejoras adicionales de eficacia de codificación de video posibles.

**[0027]** 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 en sentido directo e indirecto. Las direcciones de predicción en sentido directo e indirecto son dos direcciones de predicción de un modo de predicción bidireccional, y las expresiones “sentido directo” y “sentido indirecto” no tienen necesariamente 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 una lista de imágenes de referencia está disponible para una imagen o un segmento, solo RefPicList0 se encuentra disponible, y la información de movimiento de cada bloque de un segmento es siempre en sentido directo.

**[0028]** 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, a los fines de simplicidad, un vector de movimiento en sí mismo se puede denominar de manera que se asuma que el vector de movimiento tiene un índice de referencia asociado. Se usa un índice de referencia 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 vertical.

**[0029]** El contador de orden de imágenes (POC) se usa ampliamente en los estándares de codificación de video para identificar un orden de visualización de una imagen. Si bien existen casos en los que dos imágenes dentro de una secuencia de video codificada pueden tener el mismo valor de POC, en general, esto no ocurre dentro de una secuencia de video codificada. Cuando múltiples secuencias de video codificadas se encuentran presentes en una corriente de bits, las imágenes con un mismo valor de POC pueden estar más cerca entre sí en cuanto al orden de decodificación. En general, los valores de POC de las imágenes se usan para la construcción de listas de imágenes de referencia, la derivación de conjuntos de imágenes de referencia como en HEVC y la determinación de escala de vectores de movimiento.

**[0030]** En HEVC, la unidad de codificación más grande en un segmento se denomina bloque de codificación en árbol (CTB). Un CTB contiene un árbol cuaternario cuyos nodos son unidades de codificación. El tamaño de un CTB puede variar de 16x16 píxeles a 64x64 píxeles en el perfil principal de HEVC (aunque, técnicamente, es compatible con tamaños de CTB 8x8). Sin embargo, una unidad de codificación (CU) podría ser del mismo tamaño que un CTB y tan pequeña como 8x8 píxeles. Cada unidad de codificación se puede codificar con un modo. Cuando una CU está intercodificada, se puede particionar aún más en 2 o más unidades de predicción (PU) o volverse una sola PU cuando no se puede aplicar mayor partición. Cuando hay dos PU presentes en una CU, pueden ser rectángulos de la mitad de tamaño o dos rectángulos con un tamaño  $\frac{1}{4}$  o  $\frac{3}{4}$  del tamaño de la CU. Cuando la CU está intercodificada, hay un conjunto de información de movimiento presente para cada PU. Asimismo, cada PU se codifica 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.

**[0031]** En HEVC, hay dos modos de interpredicción, denominados modos de fusión (el salto se considera un caso especial de fusión) y modo de predicción avanzada de vectores de movimiento (AMVP), para una unidad de predicción (PU). Ya sea en AMVP o en modo de fusión, se mantiene una lista de candidatos de vectores de

movimiento (MV) para múltiples predictores de vectores de movimiento. Los vectores de movimiento, así como los índices de referencia en el modo de fusión, de la PU actual se generan tomando un candidato de la lista de candidatos de MV.

**[0032]** La lista de candidatos de MV, por HEVC, contiene hasta cinco 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 que corresponden tanto a las listas de imágenes de referencia (lista 0 y lista 1) como a los índices de referencia. Si un candidato de fusión es identificado por un índice de fusión, las imágenes de referencia se usan para la predicción de los bloques actuales, y además se determinan los vectores de movimiento asociados. Sin embargo, en el modo de AMVP, para cada posible dirección de predicción de la lista 0 o la lista 1, debe señalizarse explícitamente un índice de referencia, junto con un índice de predictor de vectores de referencia (MVP) para la lista de candidatos de MV, debido a que el candidato de AMVP contiene únicamente un vector de movimiento. En el modo de AMVP, los vectores de movimiento previstos se pueden refinar adicionalmente.

**[0033]** Como se puede observar más arriba, un candidato de fusión corresponde a un conjunto completo de información de movimiento mientras que un candidato de AMVP contiene solamente un vector de movimiento para una dirección de predicción y un índice de referencia específicos. Los candidatos de ambos modos provienen, de manera similar, de los mismos bloques vecinos espaciales y temporales.

**[0034]** Las Figuras 1A y 1B son diagramas conceptuales que ilustran ejemplos de candidatos vecinos espaciales para los modos de fusión y de 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 ( $PU_0$ ), aunque las técnicas para generar los candidatos de los bloques difieren de los modos de fusión y de AMVP.

**[0035]** En el modo de fusión de HEVC, se pueden derivar hasta 4 candidatos de MV espaciales con el orden que se muestra en la Figura 1A con los 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), de acuerdo con lo mostrado en la Figura 1A.

**[0036]** 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 se refiere a la misma imagen de referencia que la indicada por el índice de referencia señalado tiene la prioridad más alta de ser seleccionado para formar un candidato final del grupo. Es posible que ninguno de los bloques vecinos contenga un vector de movimiento que apunte hacia la misma imagen de referencia. Por lo tanto, si un candidato no puede ser encontrado, el primer candidato disponible será modificado para formar el candidato final. Por ende, se pueden compensar las diferencias de distancia temporal.

**[0037]** La compensación de movimiento en H.265/HEVC se usa para generar un predictor para un bloque intercodificado actual. Se puede usar un vector de movimiento con precisión de 1/4 píxel, y los valores de píxeles en posiciones fraccionales se pueden interpolar usando valores de píxeles enteros vecinos para componentes de luminancia y croma.

**[0038]** La Figura 2 es un diagrama conceptual que ilustra un vector de movimiento de dos puntos de ejemplo que es afín con cuatro parámetros afines para un bloque actual. En los estándares actuales de códecs de video existentes, 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, acercamiento y alejamiento, rotación, movimientos de perspectiva y otros movimientos irregulares. Si solo se aplica un modelo de movimiento traslacional para MCP en dichas secuencias de evaluación con movimientos irregulares, esto afectará la precisión de predicción y dará como resultado una baja eficacia de codificación. Por muchos años, muchos expertos de video trataron de diseñar diversos algoritmos para mejorar la MCP para una mayor eficacia de codificación. El modo de fusión afín y el intermodo afín (AMVP) se propusieron para tratar los modelos de movimiento afín con 4 parámetros como:

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

**[0039]** En la ecuación (1) anterior,  $(vx_0, vy_0)$  es el vector de movimiento del punto de control en la esquina superior izquierda del bloque actual de la Figura 2, y  $(vx_1, vy_1)$  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:

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

**[0040]** En el *software* JEM actual, la predicción de movimiento afín solo se aplica a bloques cuadrados. Como extensión natural, la predicción de movimiento afín se puede aplicar a bloques que no sean cuadrados.

**[0041]** 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 un bloque etiquetado “V0” en una esquina superior izquierda y otro etiquetado “V1” en una esquina superior derecha, y bloques vecinos etiquetados 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.

**[0042]** Para cada CU/PU cuyo tamaño es 16x16 o mayor, el modo de interpredicción afín (modo AF\_INTER) se puede aplicar de la siguiente manera. Si la CU/PU actual está en modo AF\_INTER, una bandera afín a nivel CU/PU se puede señalar en la corriente de bits. Se forma una lista de candidatos  $\{(v_0, v_1) | v_0 = \{v_A, v_B, v_C\}, v_1 = \{v_D, v_E\}\}$  usando los bloques reconstruidos válidos vecinos.

**[0043]** Como se muestra en la Figura 3, la información de movimiento  $v_0$  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. El enfoque para seleccionar  $v_1$  de los bloques vecinos D y E es similar. Si el número de la lista de candidatos es menor de 2, los candidatos de AMVP se asignan a  $v_0$  and  $v_1$ . Se usa un costo de optimización tasa-distorsión (RDO) de la CU/PU actual para determinar cuál  $(v_0, v_1)$  se selecciona como la predicción de vector de movimiento del punto de control (CPMVP) de la CU/PU actual. El índice para indicar la posición de la CPMVP en la lista de candidatos se señala en la corriente de bits.

**[0044]** Después de que se determina la CPMVP de la CU/PU afín actual, se aplica la estimación de movimiento afín y se encuentra el CPMV. Entonces, la diferencia del CPMV y la CPMVP se codifica en la corriente de bits. La predicción de compensación

de movimiento afín mencionada anteriormente se aplica para generar los residuos de la CU/PU actual. Finalmente, los residuos de la CU/PU actual se transforman, se cuantifican, y se codifican en la corriente de bits, de acuerdo con procedimientos convencionales.

**[0045]** Las Figuras 4A y 4B son diagramas conceptuales que ilustran un ejemplo de modo de fusión afín 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.

**[0046]** Cuando la CU/PU actual se aplica en modo de fusión afín (modo AF\_MERGE), se logra que el primer bloque se codifique con modo afín 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, hasta abajo a la izquierda, como se muestra en la Figura 4A. Por ejemplo, si el bloque vecino A de abajo a la izquierda se codifica en modo afín como se muestra en la Figura 4B, se derivan los vectores de movimiento  $v_2$ ,  $v_3$  y  $v_4$  de la esquina superior izquierda, esquina superior derecha, y esquina inferior izquierda de la CU/PU que contiene el bloque A. El vector de movimiento  $v_0$  de la esquina superior izquierda en la CU/PU actual se calcula de acuerdo con  $v_2$ ,  $v_3$  y  $v_4$ . De manera similar, el vector de movimiento  $v_1$  de arriba a la derecha de la CU/PU actual se calcula en función de  $v_2$ ,  $v_3$ , y  $v_4$ .

**[0047]** Después de que se calculan el CPMV de la CU/PU actual  $v_0$  y  $v_1$ , de acuerdo con el modelo de movimiento afín simplificado definido en la ecuación (2) anterior, se genera el MVF de la CU/PU actual. Luego, se aplica la MCP afín. A fin de identificar si la CU/PU actual está codificada con el modo AF\_MERGE, se señala una bandera afín en la corriente de bits cuando hay al menos un bloque vecino codificado en modo afín. Si no existe ningún bloque afín que sea vecino del bloque actual como se muestra en la Figura 4A, no se escribe ninguna bandera afín en la corriente de bits.

**[0048]** 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 eficaz a través del mapeo único de elementos sintácticos no binarios a una secuencia de bits, los cuales se denominan bins. En el *software* de referencia JEM2.0, para modo de fusión afín, solo se codifica la bandera afín, y se infiere que el índice de fusión es el primer

modelo afín vecino disponible en el orden de verificación predefinido  $A \rightarrow B \rightarrow C \rightarrow D \rightarrow E$ . Para el intermodo afín, se codifican dos sintaxis de MVD para cada lista de predicción, indicando la diferencia de vectores de movimiento entre vector de movimiento afín derivado y vector de movimiento previsto.

**[0049]** La Figura 5 es un diagrama conceptual que ilustra un modelo afín de ejemplo con seis parámetros (tres vectores de movimiento). En Zou et al., “AFFINE MOTION PREDICTION FOR VIDEO CODING”, solicitud estadounidense N.º 15/587.044, presentada el 4 de mayo de 2017, se describió un esquema de predicción de movimiento afín intercambiable. Un bloque con predicción afín puede usar un modelo afín con cuatro parámetros o un modelo afín con seis parámetros de manera adaptativa. Un modelo afín con seis parámetros se puede definir de la siguiente manera:

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

**[0050]** Un modelo afín con seis parámetros tiene tres puntos de control. En otras palabras, un modelo afín con seis parámetros se determina mediante 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 afín creado con los tres vectores de movimiento se calcula de la siguiente manera:

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

**[0051]** La ecuación (4) anterior es para un bloque cuadrado que tiene lados iguales a  $w$ . Para un bloque que no es cuadrado (por ejemplo, un bloque rectangular) que tiene un ancho de  $w$  y una altura de  $h$ , se puede usar el siguiente modelo afín:

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

**[0052]** También se puede usar una manera similar a fusión afín 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 para derivar los MVP para la

esquina superior izquierda, la esquina superior derecha y la esquina inferior izquierda. Se describen ejemplos adicionales en Chen et al., "MOTION VECTOR PREDICTION FOR AFFINE MOTION MODEL", solicitud provisional estadounidense N.º 62/404.719, presentada el 5 de octubre de 2016.

**[0053]** La Figura 6 es un diagrama de bloques que ilustra un sistema de codificación y decodificación de video 10 de ejemplo que puede usar las técnicas de la presente divulgación para codificar de manera eficaz la información de movimiento para predicción afín. Como se muestra en la Figura 6, el sistema 10 incluye un dispositivo fuente 12 que proporciona datos de video codificados que se desean decodificar posteriormente mediante un dispositivo de destino 14. En particular, el dispositivo fuente 12 proporciona los datos de video al dispositivo de destino 14 mediante un medio legible por computadora 16. El dispositivo fuente 12 y el dispositivo de destino 14 pueden comprender cualquier dispositivo del amplio rango de dispositivos, que incluyen computadoras de escritorio, notebooks (es decir, computadoras portátiles), tablets, decodificadores de televisión, microteléfonos, tales como los denominados "smartphones", "smartpads", televisiones, cámaras, dispositivos de visualización, transmisores de medios digitales, consolas de videojuegos, dispositivos de streaming de videos o similares. En algunos casos, el dispositivo fuente 12 y el dispositivo de destino 14 pueden estar equipados para la comunicación inalámbrica.

**[0054]** El dispositivo de destino 14 puede recibir los datos de video codificados que se desean decodificar mediante el medio legible por computadora 16. El medio legible por computadora 16 puede comprender cualquier tipo de medio o dispositivo que puede mover los datos de video codificados del dispositivo fuente 12 al dispositivo de destino 14. En un ejemplo, el medio legible por computadora 16 puede comprender un medio de comunicación que permite que el dispositivo fuente 12 transmita los datos de video codificados directamente al dispositivo de destino 14 en tiempo real. Los datos de video 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 basada en paquetes, tal como una red de área local, una red de área amplia o una red global, tal como Internet. El medio de comunicación puede incluir routers, conmutadores, estaciones base o cualquier otro equipo que pueda ser útil para facilitar la comunicación del dispositivo fuente 12 al dispositivo de destino 14.

**[0055]** En algunos ejemplos, los datos codificados pueden extraerse de la interfaz de salida 22 a un dispositivo de almacenamiento. De manera similar, se puede acceder a los datos codificados del dispositivo de almacenamiento mediante la interfaz de entrada. El dispositivo de almacenamiento puede incluir cualquiera de los diversos medios de almacenamiento de datos distribuidos o que se pueden acceder localmente, tales como un disco duro, discos Blu-ray, DVD, CD-ROM, memoria flash, memoria volátil o no volátil, o cualquier otro medio de almacenamiento digital adecuado para almacenar datos de video codificados. En otro ejemplo, el dispositivo de almacenamiento puede corresponder a un servidor de archivos u otro dispositivo de almacenamiento intermediario que pueda almacenar el video codificado generado por el dispositivo fuente 12. El dispositivo de destino 14 puede acceder a los datos de video almacenados del dispositivo de almacenamiento mediante *streaming* o descarga. El servidor de archivos puede ser cualquier tipo de servidor que pueda almacenar datos de video codificados y transmitir esos datos de video codificados al dispositivo de destino 14. Los ejemplos de servidores de archivos incluyen un servidor web (por ejemplo, para un sitio web), un servidor FTP, dispositivos de almacenamiento conectado en red (NAS) o una unidad de disco local. El dispositivo de destino 14 puede acceder a los datos de video 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, cable módem, etc.) o una combinación de ambos que sea adecuada para acceder a datos de video codificados almacenados en un servidor de archivos. La transmisión de datos de video codificados del dispositivo de almacenamiento puede ser una transmisión por *streaming*, una transmisión por descarga o una combinación de las dos.

**[0056]** Las técnicas de la presente divulgación no se limitan necesariamente a aplicaciones o configuraciones inalámbricas. Las técnicas se pueden aplicar a la codificación de video para diversas aplicaciones multimedia, tales como transmisiones por televisión inalámbrica, transmisiones por televisión por cable, transmisiones por televisión satelital, transmisiones de video por *streaming* en Internet, tales como *streaming* adaptativo dinámico en HTTP (DASH), video digital que se codifica en un medio de almacenamiento de datos, decodificación de video digital almacenado en un medio de almacenamiento de datos, u otras aplicaciones. En algunos ejemplos, el sistema 10 se puede configurar para permitir la transmisión de video unidireccional o bidireccional, a fin de permitir aplicaciones, tales como *streaming* de video, repetición

de video, transmisión de video y/o videotelefonía.

**[0057]** En el ejemplo de la Figura 6, el dispositivo fuente 12 incluye la fuente de video 18, el codificador de video 20 y la interfaz de salida 22. El dispositivo de destino 14 incluye la interfaz de entrada 28, el decodificador de video 30 y el dispositivo de visualización 32. De acuerdo con la presente divulgación, el codificador de video 20 del dispositivo fuente 12 se puede configurar para aplicar las técnicas para codificar de manera eficaz la información de movimiento para 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 video de una fuente de video 18 externa, tal como una cámara externa. Asimismo, el dispositivo de destino 14 puede interactuar con un dispositivo externo de visualización, en lugar de incluir un dispositivo integrado de visualización.

**[0058]** El sistema 10 ilustrado de la Figura 6 es solo un ejemplo. Las técnicas para codificar, de manera eficaz, información de movimiento para predicción afín se pueden llevar a cabo mediante cualquier dispositivo de codificación y/o decodificación de video digital. Si bien las técnicas de la presente divulgación se llevan a cabo generalmente mediante un dispositivo de codificación de video, también se pueden llevar a cabo mediante un codificador/decodificador de video, denominado generalmente “códec”. Además, las técnicas de la presente divulgación también se pueden llevar a cabo mediante un preprocesador de video. El dispositivo fuente 12 y el dispositivo de destino 14 son solo ejemplos de los dispositivos de codificación, en donde el dispositivo fuente 12 genera datos de video codificados para la transmisión al dispositivo de destino 14. En algunos ejemplos, los dispositivos 12, 14 pueden operar de manera sustancialmente simétrica, de modo que cada uno de los dispositivos 12, 14 incluya componentes de codificación y decodificación de video. Por lo tanto, el sistema 10 puede permitir la transmisión de video unidireccional o bidireccional entre los dispositivos de video 12, 14, por ejemplo, para *streaming* de video, repetición de video, transmisión de video o videotelefonía.

**[0059]** La fuente de video 18 del dispositivo fuente 12 puede incluir un dispositivo de captura de video, tal como una cámara de video, un archivo de video que contiene video capturado anteriormente y/o una interfaz de transmisión de video para recibir videos de un proveedor de contenidos de video. Como alternativa adicional, la fuente de video 18 puede generar datos basados en gráficos informáticos como el video fuente, o una combinación de video en vivo, video archivado y video generado por computadora.- En algunos casos, si la fuente de video 18 es una cámara de video, el

dispositivo fuente 12 y el dispositivo de destino 14 pueden formar los denominados teléfonos con cámara o videoteléfonos. Sin embargo, como se mencionó anteriormente, las técnicas descritas en la presente divulgación se pueden aplicar a la codificación de video en general, y se pueden aplicar a aplicaciones inalámbricas y/o alámbricas. En cada caso, el video capturado, precapturado o generado por computadora puede ser codificado por el codificador de video 20. La información de video codificada luego puede ser extraída por la interfaz de salida 22 en un medio legible por computadora 16.

**[0060]** El medio legible por computadora 16 puede incluir medios transitorios, tales como una transmisión de red inalámbrica o alámbrica, o medios de almacenamiento (es decir, medios de almacenamiento no transitorios), tales como disco duro, unidad de disco flash, disco compacto, disco de video digital, disco Blu-ray, u otros medios legibles por computadora. En algunos ejemplos, un servidor de red (no se muestra) puede recibir datos de video codificados del dispositivo fuente 12 y proporcionar los datos de video codificados al dispositivo de destino 14, por ejemplo, mediante transmisión de red. De manera similar, un dispositivo informático de una instalación de producción de medios, tal como una instalación para estampar discos, puede recibir datos de video codificados del dispositivo fuente 12 y producir un disco que contiene los datos de video codificados. Por lo tanto, el medio legible por computadora 16 puede incluir uno o más medios legibles por computadora de diversas formas, en diversos ejemplos.

**[0061]** La interfaz de entrada 28 del dispositivo de destino 14 recibe información del medio legible por computadora 16. La información del medio legible por computadora 16 puede incluir información sintáctica definida por el codificador de video 20, que también es usada por el decodificador de video 30, que incluye elementos sintácticos que describen características y/o el procesamiento de bloques y otras unidades codificadas. El dispositivo de visualización 32 muestra los datos de video decodificados a un usuario, y puede comprender cualquiera de los diversos dispositivos de visualización, tales como un tubo de rayos catódicos (CRT), una pantalla de cristal líquido (LCD), una pantalla de plasma, una pantalla de diodo orgánico de emisión de luz (OLED), u otro tipo de dispositivo de visualización.

**[0062]** El codificador de video 20 y el decodificador de video 30 pueden funcionar de acuerdo con un estándar de codificación de video, tal como el estándar de codificación de video de alta eficacia (HEVC), también denominado ITU-T H.265. De manera alternativa, el codificador de video 20 y el decodificador de video 30 pueden funcionar

de acuerdo con otros estándares industriales o propietarios, tal como el estándar ITU-T H.264, denominado de manera alternativa MPEG-4, Parte 10, codificación de video avanzada (AVC), o extensiones de dichos estándares. Sin embargo, las técnicas de la presente divulgación no se limitan a ningún estándar de codificación particular. Otros ejemplos de estándares de codificación de video incluyen MPEG-2 e ITU-T H.263. Si bien no se muestra en la Figura 6, en algunos aspectos, el codificador de video 20 y el decodificador de video 30 pueden estar integrados, cada uno, con un codificador y un decodificador de audio, y pueden incluir unidades MUX-DEMUX adecuadas, u otro *hardware* y *software*, para la codificación de audio y video en una corriente de datos común o corrientes de datos separadas. Si correspondiera, las unidades MUX-DEMUX pueden adecuarse al protocolo de multiplexor ITU H.223, o a otros protocolos, tales como el protocolo de datagrama de usuario (UDP).

**[0063]** El codificador de video 20 y el decodificador de video 30 se pueden implementar, cada uno, como cualquiera de los diversos circuitos codificadores adecuados, tales como uno o más microprocesadores, procesadores de señal digital (DSP), circuitos integrados para aplicaciones específicas (ASIC), matrices de compuerta programable en campo (FPGA), lógica discreta, *software*, *hardware*, *firmware* o cualquier combinación de estos. Cuando las técnicas se implementan de manera parcial en *software*, un dispositivo puede almacenar instrucciones para el *software* en un medio legible por computadora no transitorio adecuado y ejecutar las instrucciones en *hardware* usando uno o más procesadores para llevar a cabo las técnicas de la presente divulgación. El codificador de video 20 y el decodificador de video 30 pueden estar incluidos, cada uno, en uno o más codificadores o decodificadores, cualquiera de los cuales puede estar integrado como parte de un codificador/decodificador (CÓDEC) combinado en un dispositivo respectivo.

**[0064]** En general, de acuerdo con ITU-T H.265, una imagen de video se puede dividir en una secuencia de unidades de árboles de codificación (CTU) (o unidades de codificación más grandes (LCU)) que pueden incluir muestras de luminancia y croma. De manera alternativa, las CTU pueden incluir datos monocromáticos (es decir, solo muestras de luminancia). Los datos sintácticos en una corriente de bits pueden definir un tamaño para la CTU, la cual es una unidad de codificación más grande en cuanto a cantidad de píxeles. Un segmento incluye una cantidad de CTU consecutivas en orden de codificación. Una imagen de video se puede particionar en uno o más segmentos. Cada CTU se puede dividir 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 que corresponde a la CTU. Si una CU se divide en cuatro sub-CU, el nodo que corresponde a la CU incluye cuatro nodos hoja, cada uno de los cuales corresponde a una de las sub-CU.

**[0065]** Cada nodo de la estructura de datos de árbol cuaternario puede proporcionar datos sintácticos para la CU correspondiente. Por ejemplo, un nodo en el árbol cuaternario puede incluir una bandera dividida, indicando si la CU que corresponde al nodo se divide en sub-CU. Los elementos sintácticos para una CU se pueden definir repetidamente, y pueden depender de si la CU se divide en sub-CU. Si una CU no se divide adicionalmente, a esta se la denomina CU hoja. En la presente divulgación, cuatro sub-CU de una CU hoja también se denominarán CU hoja aun si no hay división explícita de la CU hoja original. Por ejemplo, si una CU de tamaño 16x16 no se vuelve a dividir, las cuatro sub-CU de 8x8 también se denominarán CU hoja aunque la CU de 16x16 no se haya dividido.

**[0066]** Una CU tiene un fin similar al de un macrobloque del estándar H.264, a menos que una CU no tenga distinción de tamaño. Por ejemplo, una CTU se puede dividir en cuatro nodos hijos (también denominados sub-CU), y cada nodo hijo, a su vez, puede ser un nodo padre y dividirse en otros cuatro nodos hijos. Un nodo hijo final y sin dividir, denominado nodo hoja del árbol cuaternario, comprende un nodo de codificación, también denominado CU hoja. Los datos sintácticos asociados a una corriente de bits codificada pueden definir una cantidad máxima de veces que se puede dividir una CTU, denominada profundidad de CU máxima, y también pueden definir un tamaño mínimo de los nodos de codificación. En consecuencia, una corriente de bits también puede definir una unidad de codificación más pequeña (SCU). En la presente divulgación se usa el término “bloque” para referirse a cualquier CU, unidad de predicción (PU), o unidad de transformada (TU), en el contexto de HEVC, o estructuras de datos similares en el contexto de otros estándares (por ejemplo, macrobloques y subbloques de estos en H.264/AVC).

**[0067]** Una CU incluye un nodo de codificación y unidades de predicción (PU) y unidades de transformada (TU) asociadas al nodo de codificación. El tamaño de la CU corresponde a un tamaño del nodo de codificación y, en general, puede tener forma cuadrada. El tamaño de la CU puede variar de 8x8 píxeles hasta el tamaño de la CTU con un tamaño máximo, por ejemplo, de 64x64 píxeles o mayor. Cada CU puede contener una o más PU y una o más TU. Los datos sintácticos asociados a una CU pueden describir, por ejemplo, la partición de la CU en una o más PU. Los modos de partición pueden diferir entre si la CU se codifica en modo directo o de salto, se

codifica en modo intrapredicción o se codifica en modo interpredicción. Las PU se pueden particionar para que no tengan forma cuadrada. Los datos sintácticos asociados a una CU también pueden describir, por ejemplo, la partición de la CU en una o más TU de acuerdo con un árbol cuaternario. Una TU puede tener forma cuadrada o no cuadrada (por ejemplo, forma rectangular).

**[0068]** El estándar de HEVC permite transformaciones de acuerdo con las TU, las cuales pueden ser diferentes para diferentes CU. En general, las TU se dimensionan en función del tamaño de las PU (o particiones de una CU) dentro de una CU determinada que se define para una CTU particionada, aunque esto no ocurre siempre. En general, las TU tienen el mismo tamaño o son más pequeñas que las PU (o particiones de una CU, por ejemplo, en el caso de intrapredicción). En algunos ejemplos, las muestras residuales que corresponden a una CU se pueden subdividir en unidades más pequeñas usando una estructura de árbol cuaternario conocida como “árbol cuaternario residual” (RQT). Los nodos hoja del RQT se denominan unidades de transformada (TU). Los valores de diferencia de píxeles asociados a las TU se pueden transformar para producir coeficientes de la transformada, los cuales se pueden cuantificar.

**[0069]** Una CU hoja puede incluir una o más unidades de predicción (PU) cuando predice usando interpredicción. En general, una PU representa un área espacial que corresponde a la totalidad o una porción de la CU correspondiente, y puede incluir datos para recuperar y/o generar una muestra de referencia para la PU. Asimismo, una PU incluye datos relacionados a la predicción. Cuando la CU se codifica intermodo, una o más PU de la CU pueden incluir datos que definen la información de movimiento, tal como uno o más vectores de movimiento, o las PU se pueden codificar en modo de salto. 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 1/4 píxel o precisión de 1/8 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.

**[0070]** Las CU hoja también se pueden predecir intramodo. En general, la intrapredicción implica predecir una CU hoja (o particiones de esta) usando un intramodo. Un codificador de video puede seleccionar un conjunto de píxeles vecinos previamente codificados para la CU hoja a fin de que se use para predecir la CU hoja (o particiones de esta).

**[0071]** Una CU hoja también puede incluir una o más unidades de transformada (TU). Las unidades de transformada se pueden especificar usando un RQT (también denominado estructura de árbol cuaternario de TU), como se describió anteriormente. Por ejemplo, una bandera dividida puede indicar si una CU hoja se divide en cuatro unidades de transformada. Luego, cada TU se puede volver a dividir en otras sub-TU. Cuando una TU no se divide adicionalmente, a esta se la puede denominar TU hoja. Generalmente, para 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, en general, para calcular valores previstos para todas las TU de una CU hoja. Para intracodificación, un codificador de video puede calcular un valor residual para cada TU hoja usando el modo de intrapredicción, como una diferencia entre la porción de la CU que corresponde a la TU y el 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 intracodificación, las particiones de una CU, o la CU misma, se pueden asignar 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.

**[0072]** Además, las TU de las CU hoja también se pueden asociar a estructuras de datos de árbol cuaternario respectivas, denominadas árboles cuaternarios residuales (RQT). Es decir, una CU hoja puede incluir un árbol cuaternario que indica cómo se particiona la CU hoja en TU. El nodo raíz de un árbol cuaternario de TU en general corresponde a una CU hoja, mientras que el nodo raíz de un árbol cuaternario de CU en general corresponde a una CTU (o LCU). Las TU del RQT que no se dividen se denominan TU hoja. En general, la presente divulgación usa los términos CU y TU para referirse a la CU hoja y TU hoja, respectivamente, a menos que se indique lo contrario.

**[0073]** Una secuencia de video, en general, incluye una serie de imágenes o cuadros de video, que comienza con una imagen de punto de acceso aleatorio (RAP). Una secuencia de video puede incluir datos sintácticos en un conjunto de parámetros de secuencia (SPS) con características de la secuencia de video. Cada segmento de una imagen puede incluir datos sintácticos de segmento que describen un modo de codificación para el segmento respectivo. En general, el codificador de video 20 opera en bloques de video dentro de segmentos de video individuales para codificar los datos de video. Un bloque de video puede corresponder al nodo de codificación dentro de una CU. Los bloques de video pueden tener tamaños fijos o variables, y pueden

diferir en tamaño de acuerdo con un estándar de codificación especificado.

**[0074]** Como un ejemplo, se puede llevar a cabo la predicción para las PU de diversos tamaños. Suponiendo que el tamaño de una CU particular es  $2N \times 2N$ , se puede llevar a cabo intrapredicción en tamaños de PU de  $2N \times 2N$  o  $N \times N$ , y se puede llevar a cabo interpredicción en tamaños de PU simétricos de  $2N \times 2N$ ,  $2N \times N$ ,  $N \times 2N$ , o  $N \times N$ . La partición asimétrica para interpredicción también se puede llevar a cabo 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 que corresponde a la partición del 25 % se indica mediante una “n” seguida de una indicación de “Up”, “Down,” “Left” o “Right”. Por ende, por ejemplo, “ $2N \times nU$ ” se refiere a una CU de  $2N \times 2N$  que está particionada horizontalmente con una PU de  $2N \times 0,5N$  por arriba y una PU de  $2N \times 1,5N$  por debajo.

**[0075]** En la presente divulgación, “ $N \times N$ ” y “N por N” se pueden usar indistintamente para hacer referencia a las dimensiones de píxeles de un bloque de video en términos de dimensiones vertical y horizontal, 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 ( $x = 16$ ). De manera similar, un bloque de  $N \times N$  tiene por lo general N píxeles en una dirección vertical y N píxeles en una dirección horizontal, en donde N representa un valor entero no negativo. Los píxeles en un bloque se pueden disponer en filas y columnas. Además, los bloques no tienen necesariamente la misma cantidad de píxeles en la dirección horizontal que en la dirección vertical. Por ejemplo, los bloques pueden comprender  $N \times M$  píxeles, en donde M no es necesariamente igual a N.

**[0076]** Posteriormente a la codificación intrapredictiva o interpredictiva mediante el uso de las PU de una CU, el codificador de video 20 puede calcular los datos residuales para las TU de la CU. Las PU pueden comprender datos sintácticos que describen un método o modo de generar datos de píxeles predictivos en el dominio espacial (también denominado el dominio de píxeles) y las TU pueden comprender coeficientes en el dominio de la transformada luego de la aplicación de una transformada, por ejemplo, una transformada discreta del coseno (DCT), una transformada entera, una transformada de ondícula, o una transformada conceptualmente similar a datos de video residuales. Los datos residuales pueden corresponder a diferencias de píxeles entre píxeles de la imagen no codificada y los valores de predicción que corresponden a las PU. El codificador de video 20 puede formar las TU para que incluyan coeficientes de la transformada cuantificados representativos de los datos residuales

para la CU. Es decir, un codificador de video 20 puede calcular los datos residuales (en forma de un bloque residual), transformar el bloque residual para producir un bloque de coeficientes de la transformada, y luego cuantificar los coeficientes de la transformada para formar coeficientes de la transformada cuantificados. El codificador de video 20 puede formar una TU que incluye los coeficientes de la transformada cuantificados, así como otra información sintáctica (por ejemplo, información de división para la TU).

**[0077]** Como se indicó anteriormente, luego de las transformaciones para producir coeficientes de la transformada, el codificador de video 20 puede llevar a cabo la cuantificación de los coeficientes de la transformada. En general, la cuantificación se refiere a un proceso en donde los coeficientes de la transformada se cuantifican para reducir posiblemente la cantidad de datos que se usan para representar los coeficientes, lo que proporciona una mayor compresión. El proceso de cuantificación puede reducir la profundidad de bits asociada a algunos o todos los coeficientes. Por ejemplo, un valor de  $n$  bits puede redondearse hacia abajo a un valor de  $m$  bits durante la cuantificación, en donde  $n$  es mayor que  $m$ .

**[0078]** Después de la cuantificación, el codificador de video puede explorar los coeficientes de la transformada, lo que produce un vector unidimensional de la matriz bidimensional que incluye los coeficientes de la transformada cuantificados. La exploración se puede diseñar para colocar coeficientes de mayor energía (y por lo tanto de menor frecuencia) al frente de la matriz y para colocar coeficientes de menor energía (y por lo tanto mayor frecuencia) en la parte posterior de la matriz. En algunos ejemplos, el codificador de video 20 puede utilizar un orden de barrido predefinido para barrer los coeficientes de la transformada cuantificados para producir un vector serializado que se puede codificar por entropía. En otros ejemplos, el codificador de video 20 puede llevar a cabo una exploración adaptativa. Luego de explorar los coeficientes de la transformada cuantificados para formar un vector unidimensional, el codificador de video 20 puede codificar por entropía el vector unidimensional, por ejemplo, de acuerdo con codificación de longitud variable adaptativa según el contexto (CAVLC), codificación aritmética binaria adaptativa según el contexto (CABAC), codificación aritmética binaria adaptativa según el contexto en función de la sintaxis (SBAC), codificación entrópica de partición en intervalos de probabilidad (PIPE) u otra metodología de codificación entrópica. El codificador de video 20 también puede codificar por entropía los elementos sintácticos asociados a los datos de video codificados para usar mediante decodificador de video 30 en la decodificación de los

datos de video.

**[0079]** Para llevar a cabo la CABAC, el codificador de video 20 puede asignar un contexto dentro de un modelo de contexto para un símbolo a transmitir. El contexto se refiere, por ejemplo, a la cuestión de si los valores vecinos del símbolo son distintos a cero o no. Para llevar a cabo la CAVLC, el codificador de video 20 puede seleccionar un código de longitud variable para un símbolo que se debe transmitir. Las palabras codificadas en VLC se pueden construir de tal manera que códigos relativamente más cortos correspondan a símbolos más probables, mientras que códigos más largos correspondan a símbolos menos probables. De esta forma, el uso de VLC puede lograr un ahorro de bits mayor que, por ejemplo, usar palabras codificadas de la misma longitud para cada símbolo que se debe transmitir. La determinación de probabilidad puede ser en función de un contexto asignado al símbolo.

**[0080]** En general, el decodificador de video 30 lleva a cabo un proceso sustancialmente similar, no obstante recíproco, a aquel llevado a cabo mediante el codificador de video 20 para decodificar datos codificados. Por ejemplo, el decodificador de video 30 cuantifica de manera inversa y transforma de manera inversa coeficientes de una TU recibida para reproducir un bloque residual. El decodificador de video 30 usa un modo de predicción señalizada (intrapredicción o interpredicción) para formar un bloque predicho. Luego, el decodificador de video 30 combina el bloque predicho y el bloque residual (píxel por píxel) para reproducir el bloque original. Se puede llevar a cabo procesamiento adicional, tal como llevar a cabo un proceso de eliminación de bloques para reducir artefactos visuales junto con límites del bloque. Además, el decodificador de video 30 puede decodificar los elementos sintácticos con el uso de CABAC de manera sustancialmente similar, no obstante recíproca, a los procesos de codificación de CABAC del codificador de video 20.

**[0081]** En general, el codificador de video 20 y el decodificador de video 30 se pueden configurar para codificar más eficazmente (codificar o decodificar, respectivamente) información de movimiento para predicción afín, de acuerdo con las técnicas de la presente divulgación. El codificador de video 20 y/o el decodificador de video 30 se pueden configurar para aplicar cualquiera a las diversas técnicas analizadas más adelante, solas o combinadas.

**[0082]** En algunos ejemplos, el codificador de video 20 y/o el decodificador de video 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 se puede definir como la diferencia entre el MV y la predicción

de vector de movimiento (MVP):  $MVD = MV - MVP$ . Más específicamente, si un vector de movimiento ( $MV_x, MV_y$ ) se indica por su componente horizontal ( $MV_x$ ) y componente vertical ( $MV_y$ ), y el predictor de vectores 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 ende, la MVD se puede definir como ( $MVD_x, MVD_y$ ), en donde  $MVD_x = MV_x - MVP_x$ , y  $MVD_y = MV_y - MVP_y$ .

**[0083]** De manera adicional o alternativa, en algunos ejemplos, el codificador de video 20 y/o el decodificador de video 30 se pueden configurar para usar la MVD del primer MV para predecir las MVD de uno o más de otros MV en predicción afín. La Figura 9 es un diagrama conceptual que ilustra un ejemplo de tal predicción de MVD. En algunos ejemplos, el codificador de video 20 y/o el decodificador de video 30 se pueden configurar para usar la MVD de un primer MV para predecir el MVD de un segundo MV en predicción afín (por ejemplo, afín de cuatro parámetros). La Figura 9 más adelante muestra un ejemplo de predicción de MVD para predicción afín con dos vectores de movimiento, en donde MVD1 se predice mediante MVD0.

**[0084]** La Figura 10 es un diagrama conceptual que ilustra un ejemplo de predicción de MVD para predicción afín con tres vectores de movimiento (predicción afín de seis parámetros). Para la predicción afín de seis parámetros, el codificador de video 20 y/o el decodificador de video 30 pueden usar la MVD del primer MV para predecir la MVD del segundo MV. Además, el codificador de video 20 y/o el decodificador de video 30 pueden usar la MVD del primer MV para predecir la MVD del tercer MV en predicción afín con tres vectores de movimiento. La Figura 10 muestra un ejemplo de predicción de MVD para predicción afín con tres vectores de movimiento, en donde MVD1 se predice mediante MVD0 y MVD2 también se predice mediante MVD0.

**[0085]** Con respecto a la Figura 6, en algunos ejemplos, el codificador de video 20 y el decodificador de video 30 se pueden configurar de modo que el primer MV en los ejemplos anteriores se define como el MV asociado al punto de control superior izquierdo indicado como “MV0” en las Figuras. 3, 9 y 10. De manera alternativa, el codificador de video 20 y el decodificador de video 30 se pueden configurar para seleccionar de manera adaptativa el primer punto de control asociado al primer MV. Por ejemplo, el primer punto de control puede depender de información codificada, tal como la forma del bloque. De manera alternativa, el codificador de video 20 y el decodificador de video 30 pueden derivar de manera implícita el primer punto de control asociado al primer MV.

**[0086]** De manera adicional o alternativa, en algunos ejemplos, el codificador de video 20 y/o el decodificador de video 30 se pueden configurar para aplicar predicción de MVD entre cualesquiera dos MVD para predicción afín. Por ejemplo, el codificador de video 20 y el decodificador de video 30 pueden predecir MVD0 de MVD1 para predicción afín con dos vectores de movimiento. En otro ejemplo, el codificador de video 20 y el decodificador de video 30 pueden predecir MVD1 de MVD0, y predecir MVD2 de MVD1, para predicción afín con tres vectores de movimiento.

**[0087]** De manera adicional o alternativa, en algunos ejemplos, el codificador de video 20 y/o el decodificador de video 30 se pueden configurar para predecir MVDa de MVDb. El codificador de video 20 puede calcular  $MVDa' = MVDa - MVDb$ , y codificar información que representa MVDa' como parte de la corriente de bits, de manera que el decodificador de video 30 puede decodificar esta información para determinar MVDa'. Entonces, el decodificador de video 30 puede calcular  $MVDa = MVDa' + MVDb$ . En un ejemplo,  $a=1$  y  $b = 0$  para predicción afín con cuatro parámetros.

**[0088]** De manera adicional o alternativa, en algunos ejemplos, el codificador de video 20 y/o el decodificador de video 30 se pueden configurar para predecir MVDa de MVDb. Entonces, el codificador de video 20 puede calcular  $MVDa' = MVDa - w * MVDb$ , y codificar información que representa MVDa' como parte de la corriente de bits, de manera que el decodificador de video 30 puede decodificar esta información para determinar MVDa'. Entonces, el decodificador de video 30 puede 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 predicción afín con cuatro parámetros. Este ejemplo se puede implementar 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 video 20 codifica datos que representan  $w$  como parte de la corriente de bits, por ejemplo, a nivel de la secuencia (tal como un conjunto de parámetros de secuencia (SPS)), a nivel de la imagen (tal como un conjunto de parámetros de imagen (PPS)), a nivel del segmento (tal como en una cabecera de segmento), o a nivel del bloque (tal como en una cabecera de bloque). El decodificador de video 30 puede adicionalmente extraer esta información señalizada del nivel de información correspondiente.

**[0089]** De manera adicional o alternativa, en algunos ejemplos, el codificador de video 20 y/o el decodificador de video 30 se pueden configurar para predecir la MVD de un punto de control de las MVD de más de uno de otro punto de control. Por ejemplo, el codificador de video 20 y el decodificador de video 30 pueden predecir MVD2 de

MVD0 y MVD1 en el modelo afín de seis parámetros. Como un ejemplo, el codificador de video 20 puede calcular  $MVD2' = MVD2 - ((MVD0 + MVD1) >> 1)$ , y codificar información que representa a MVD2' como parte de la corriente de bits, de manera que el decodificador de video 30 puede decodificar esta información para determinar MVD2'. Entonces, el decodificador de video 30 puede usar esta información para calcular  $MVD2 = MVD2' + ((MVD0 + MVD1) >> 1)$ .

**[0090]** De manera adicional o alternativa, en algunos ejemplos, el codificador de video 20 y/o el decodificador de video 30 se pueden configurar para determinar si predecir o no una MVD en función de un valor de MVD. Por ejemplo, el codificador de video 20 puede determinar predecir MVDa de MVDb si  $|MVDb^x + MVDb^y| < T$ ; de lo contrario, el codificador de video 20 no predice MVDa de MVDb. En otro ejemplo, el codificador de video 20 puede determinar predecir MVDa de MVDb si  $\max(|MVDb^x|, |MVDb^y|) < T$ ; de lo contrario, el codificador de video 20 no predice MVDa de MVDb. En aun otro ejemplo, el codificador de video 20 puede predecir MVDa de MVDb si  $|MVDb^x + MVDb^y| > T$ ; de lo contrario, el codificador de video 20 no predice MVDa de MVDb. En aun otro ejemplo, el codificador de video 20 puede predecir MVDa de MVDb si  $\min(|MVDb^x|, |MVDb^y|) > T$ ; de lo contrario, el codificador de video 20 no predice MVDa de MVDb. En los ejemplos anteriores, T representa un umbral, que puede ser una cantidad fija o señalizada mediante el codificador de video 20 y decodificada mediante el decodificador de video 30. El codificador de video 20 puede codificar datos que representan si predecir o no MVDa de MVDb en función de cualquiera de las determinaciones de ejemplo anteriores, y el decodificador de video 30 puede decodificar estos datos codificados para determinar si predecir MVDa de MVDb.

**[0091]** De manera adicional o alternativa, en algunos ejemplos, el codificador de video 20 y/o el decodificador de video 30 se pueden configurar para predecir los componentes horizontales (x) y/o verticales (y) de una MVD de diferentes maneras. Por ejemplo, el codificador de video 20 y el decodificador de video 30 pueden predecir solo el componente x de MVDb del componente x de MVDa, pero no predecir el componente y de MVDb de otra MVD (por ejemplo, MVDa).

**[0092]** De manera adicional o alternativa, en algunos ejemplos, el codificador de video 20 y/o el decodificador de video 30 se pueden configurar para determinar si predecir la MVD en función del valor del MVP. Por ejemplo, el codificador de video 20 puede determinar predecir MVDa de MVDb si  $|MVPa^x - MVPb^x| + |MVPa^y - MVPb^y| < S$ ; de lo contrario, el codificador de video 20 no predice MVDa de MVDb. En otro ejemplo, el codificador de video 20 puede determinar predecir MVDa de MVDb si  $\max(|MVPa^x -$

$MVPb^x]$ ,  $|MVPa^y - MVPb^y|) < S$ ; de lo contrario, el codificador de video 20 no predice MVDa de MVDb. En los ejemplos anteriores, S representa un umbral, que puede ser una cantidad fija o señalizada mediante el codificador de video 20 y decodificada mediante el decodificador de video 30. El codificador de video 20 puede codificar datos que representan si predecir o no MVDa de MVDb en función de cualquiera de las determinaciones de ejemplo anteriores, y el decodificador de video 30 puede decodificar estos datos codificados para determinar si predecir MVDa de MVDb.

**[0093]** De manera adicional o alternativa, en algunos ejemplos, el codificador de video 20 y/o el decodificador de video 30 se pueden configurar para determinar si predecir una MVD para un bloque predicho usando predicción afín basada en un método de predicción de movimiento. Por ejemplo, si el MVP proviene del método de derivación de MVP en JEM como se describió anteriormente con respecto a la Figura 3, el codificador de video 20 y el decodificador de video 30 pueden determinar no usar predicción de MVD. Como otro ejemplo, si el MVP proviene del método de derivación de MVP similar a fusión afín como se describió anteriormente con respecto a la solicitud provisional estadounidense N.º 62/404.719, el codificador de video 20 y el decodificador de video 30 pueden determinar usar predicción de MVD. De manera adicional o alternativa, el codificador de video 20 y el decodificador de video 30 pueden determinar si usar predicción de MVD en función de si se usa compensación de iluminación para el bloque fuente del MVP.

**[0094]** De manera adicional o alternativa, en algunos ejemplos, el codificador de video 20 y/o el decodificador de video 30 se pueden configurar para determinar si predecir una MVD en función de un tamaño y/o forma de un bloque actual. Por ejemplo, el codificador de video 20 y el decodificador de video 30 pueden determinar usar predicción de MVD cuando  $W \cdot H > T$ , en 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 una cantidad fija o señalizada del codificador de video 20 al decodificador de video 30 en la corriente de bits. En otro ejemplo, el codificador de video 20 y el decodificador de video 30 pueden determinar usar predicción de MVD cuando  $W \cdot H < T$ .

**[0095]** De manera adicional o alternativa, en algunos ejemplos, el codificador de video 20 y/o el decodificador de video 30 se pueden configurar para codificar datos que representan si predecir o no una MVD (o un componente de esta) en una corriente de bits. Es decir, el codificador de video 20 puede codificar datos que representan si predecir uno o ambos de los componentes (horizontal y vertical) de una MVD en la

corriente de bits, y el decodificador de video 30 puede determinar si predecir uno o ambos de los componentes de la MVD de los datos codificados de la corriente de bits (decodificando los datos).

**[0096]** De manera adicional o alternativa, en algunos ejemplos, el codificador de video 20 y/o el decodificador de video 30 se pueden configurar para codificar datos que representan qué puntos de control usar como referencia para la predicción de MVD en la corriente de bits. Es decir, el codificador de video 20 puede codificar estos datos, y el decodificador de video 30 puede decodificar estos datos para determinar qué puntos de control usar como referencia para la predicción de MVD.

**[0097]** De manera adicional o alternativa, en algunos ejemplos, el codificador de video 20 y/o el decodificador de video 30 se pueden configurar para generar el MVP de un MV de la MVD de otro MV en un bloque predicho usando predicción afín.

**[0098]** De manera adicional o alternativa, en algunos ejemplos, el codificador de video 20 y/o el decodificador de video 30 se pueden configurar para usar la MVD de un primer MV para generar los MVP de uno o más de otros MV en predicción afín. En un ejemplo, el codificador de video 20 y el decodificador de video 30 pueden usar la MVD del primer MV para generar el MVP del segundo MV en predicción afín (por ejemplo, afín de cuatro parámetros). En otro ejemplo, para predicción afín de seis parámetros, el codificador de video 20 y el decodificador de video 30 pueden usar la MVD del primer MV para generar el MVP del segundo MV en predicción afín con tres vectores de movimiento. Además, el codificador de video 20 y el decodificador de video 30 pueden usar la MVD del primer MV para generar el MVP del tercer MV en predicción afín con tres vectores de movimiento. De manera alternativa, el codificador de video 20 y el decodificador de video 30 pueden usar la MVD del segundo MV para generar el MVP del tercer MV en predicción afín con tres vectores de movimiento.

**[0099]** De manera adicional o alternativa, en algunos ejemplos, el codificador de video 20 y/o el decodificador de video 30 se pueden configurar para generar MVP0 de MVD1 para predicción afín con dos vectores de movimiento. En otro ejemplo, el codificador de video 20 y/o el decodificador de video 30 se pueden configurar para generar MVP1 de MVD0, y para generar MVP2 de MVD1, para predicción afín con tres vectores de movimiento.

**[00100]** De manera adicional o alternativa, en algunos ejemplos, el codificador de video 20 y/o el decodificador de video 30 se pueden configurar para calcular MVPa de acuerdo con  $MVPa = MVP'a + MVDb$ . MVP'a representa el MVP generado de la manera original sin considerar MVDb como se describió anteriormente para AMVP y modo de

fusión para predicción afín. En un ejemplo,  $a=1$  y  $b=0$  para predicción afín con cuatro parámetros (dos vectores de movimiento).

**[00101]** De manera adicional o alternativa, en algunos ejemplos, el codificador de video 20 y/o el decodificador de video 30 se pueden configurar para calcular MVPa de acuerdo con  $MVPa = MVP'a + w * MVD_b$ , en donde  $w$  es un valor de ponderación, tal como 0,5. En este ejemplo, MVP'a es el MVP generado de la manera original sin considerar MVD<sub>b</sub> como se describió anteriormente para AMVP y modo de fusión para predicción afín. En un ejemplo,  $a=1$  y  $b=0$  para predicción afín con dos parámetros. Este ejemplo se puede implementar 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 video 20 determina  $w$  y señaliza el valor de  $w$  en la corriente de bits a nivel de la secuencia, nivel de la imagen, nivel del segmento, o nivel del bloque. El decodificador de video 30, en consecuencia, decodifica el valor de  $w$  del nivel de información apropiado.

**[00102]** De manera adicional o alternativa, en algunos ejemplos, el codificador de video 20 y/o el decodificador de video 30 se pueden configurar para generar el MVP de un punto de control de las MVD de otros múltiples puntos de control. Por ejemplo, el codificador de video 20 y/o el decodificador de video 30 pueden generar MVP2 de MVD0 y MVD1 en el modelo afín de seis parámetros. El codificador de video 20 y/o el decodificador de video 30 pueden calcular MVP2 como  $MVP2 = MVP'2 + ((MVD0 + MVD1) >> 1)$ .

**[00103]** De manera adicional o alternativa, en algunos ejemplos, el codificador de video 20 y/o el decodificador de video 30 se pueden configurar para usar el MV de un punto de control para generar los MVP de los MV de uno o más de otros puntos de control en un bloque predicho usando predicción afín. En un ejemplo, el codificador de video 20 y/o el decodificador de video 30 pueden usar el primer MV para generar el MVP del segundo MV en predicción afín (por ejemplo, afín de cuatro parámetros). En otro ejemplo, para la predicción afín de seis parámetros, el codificador de video 20 y/o el decodificador de video 30 pueden usar el primer MV para generar el MVP del segundo MV en predicción afín con tres vectores de movimiento, y usar el primer MV para generar el MVP del tercer MV en predicción afín con tres vectores de movimiento. De manera alternativa, para la predicción afín de seis parámetros, el codificador 20 y/o el decodificador de video 30 pueden usar el segundo MV para generar el MVP del tercer MV en predicción afín con tres vectores de movimiento.

**[00104]** De manera adicional o alternativa, en algunos ejemplos, el codificador de

video 20 y/o el decodificador de video 30 se pueden configurar para calcular MVPa como  $MVPa = (MVP'a + MVb) \gg 1$ . MVP'a es el MVP generado de la manera original sin considerar MVb como se describió anteriormente en las secciones que analizan AMVP y fusión para predicción afín. En un ejemplo,  $a=1$  y  $b = 0$  para predicción afín con cuatro parámetros.

**[00105]** De manera adicional o alternativa, en algunos ejemplos, el codificador de video 20 y/o el decodificador de video 30 se pueden configurar para calcular MVPa como  $MVPa = w1 * MVP'a + w2 * MVb$ . En este ejemplo,  $w1$  y  $w2$  son valores de ponderación que pueden tener el mismo o diferente valor, por ejemplo,  $w1=w2= 0,5$ . MVP'a es el MVP generado de la manera original sin considerar MVb como se describió en las secciones anteriores sobre AMVP y modo de fusión para predicción afín. En un ejemplo,  $a=1$  y  $b=0$  para predicción afín con cuatro parámetros. Este ejemplo se puede implementar en forma de entero, como  $MVP1 = (3 * MVP'1 + MV0 + 2) \gg 2$  cuando  $w1 = 0,75$  y  $w2 = 0,25$ . En un ejemplo, el codificador de video 20 codifica datos para  $w1$  y  $w2$  en la corriente de bits a nivel de la secuencia, nivel de la imagen, nivel del segmento, o nivel del bloque. De igual modo, el decodificador de video 30 determina  $w1$  y  $w2$  decodificando estos datos en el nivel apropiado.

**[00106]** El codificador de video 20 también puede enviar datos sintácticos, tales como datos sintácticos basados en bloque, datos sintácticos basados en imagen, y datos sintácticos basados en secuencia, al decodificador de video 30, por ejemplo, en una cabecera de imagen, una cabecera de bloque, una cabecera de segmento, u otros datos sintácticos, tales como el conjunto de parámetros de secuencias (SPS), el conjunto de parámetros de imágenes (PPS), o el conjunto de parámetros de video (VPS).

**[00107]** El codificador de video 20 y el decodificador de video 30 se pueden implementar, cada uno, como cualquiera de los diversos circuitos codificadores o decodificadores adecuados, según corresponda, tales como uno o más microprocesadores, procesadores de señal digital (DSP), circuitos integrados para aplicaciones específicas (ASIC), matrices de compuerta programable en campo (FPGA), circuito lógico discreto, software, hardware, firmware o cualquier combinación de estos. El codificador de video 20 y el decodificador de video 30 pueden estar incluidos, cada uno, en uno o más codificadores o decodificadores, cualquiera de los cuales puede estar integrado como parte de un codificador/decodificador (CÓDEC) de video combinado. Un dispositivo que incluye codificador de video 20 y/o decodificador de video 30 puede comprender un circuito integrado, un microprocesador, y/o un

dispositivo de comunicación inalámbrico, tal como un teléfono celular.

**[00108]** La Figura 7 es un diagrama de bloques que ilustra un ejemplo de codificador de video 20 que puede implementar las técnicas de la presente divulgación para codificar información de movimiento de predicción afín. El codificador de video 20 puede realizar intracodificación e intercodificación de bloques de video dentro de segmentos de video. La intracodificación depende de la predicción espacial para reducir o eliminar la redundancia espacial en el video dentro de un cuadro de video o imagen determinados. La intercodificación depende de la predicción temporal para reducir o eliminar la redundancia temporal en el video dentro de cuadros o imágenes adyacentes de una secuencia de video. El intramodo (modo I) puede hacer referencia a cualquiera de los diversos modos de codificación en función del espacio. Los intermodos, tales como predicción unidireccional (modo P) o bipredicción (modo B), pueden hacer referencia a cualquiera de los diversos modos de codificación en función del tiempo.

**[00109]** Como se muestra en la Figura 7, el codificador de video 20 recibe un bloque de video actual dentro de un cuadro de video que se desea codificar. En el ejemplo de la Figura 7, el codificador de video 20 incluye una unidad de selección de modo 40, memoria de imágenes de referencia 64 (que también puede denominarse memoria de imagen decodificada (DPB)), sumador 50, unidad de procesamiento de la transformada 52, unidad de cuantificación 54 y unidad de codificación entrópica 56. A su vez, la unidad de selección de modo 40 incluye unidad de compensación de movimiento 44, unidad de estimación de movimiento 42, unidad de intrapredicción 46 y unidad de partición 48. Para la reconstrucción del bloque de video, el codificador de video 20 también incluye unidad de cuantificación inversa 58, unidad de la transformada inversa 60 y sumador 62. Un filtro de eliminación de bloques (no se muestra en la Figura 7) también se puede incluir para filtrar los límites del bloque para eliminar artefactos de compresión del video reconstruido. Si se desea, el filtro de eliminación de bloques filtraría, en general, la salida del sumador 62. También se pueden usar filtros adicionales (en bucle o posbucle) además del filtro de eliminación de bloques. Tales filtros no se muestran para mayor brevedad, pero si se desea, pueden filtrar la salida del sumador 50 (como un filtro en bucle).

**[00110]** Durante el proceso de codificación, el codificador de video 20 recibe un cuadro o un segmento de video que se desean codificar. El cuadro o el segmento se pueden dividir en múltiples bloques de video. La unidad de estimación de movimiento 42 y la unidad de compensación de movimiento 44 realizan la codificación

interpredictiva del bloque de video recibido con respecto a uno o más bloques en uno o más cuadros de referencia para proporcionar predicción temporal. La unidad de intrapredicción 46 puede realizar de manera alternativa la codificación intrapredictiva del bloque de video recibido con respecto a uno o más bloques vecinos en el mismo cuadro o segmento como el bloque que se desea codificar para proporcionar predicción espacial. El codificador de video 20 puede realizar múltiples pases de codificación, por ejemplo, para seleccionar un modo de codificación adecuado para cada bloque de datos de video.

**[00111]** Asimismo, la unidad de partición 48 puede particionar bloques de datos de video en subbloques, en función de la evaluación de los esquemas de partición previos en pases de codificación previos. Por ejemplo, la unidad de partición 48 puede particionar inicialmente un cuadro o un segmento en CTU, y particionar cada una de las CTU en sub-CU en función del análisis de tasa-distorsión (por ejemplo, optimización de tasa-distorsión). La unidad de selección de modo 40 puede producir adicionalmente una estructura de datos de árbol cuaternario que indica la partición de una CTU en sub-CU. Las CU con nodo hoja del árbol cuaternario pueden incluir una o más PU y una o más TU.

**[00112]** La unidad de selección de modo 40 puede seleccionar uno de los modos de predicción, intra o inter, por ejemplo, en función de los resultados de errores, y proporcionar el bloque predicho resultante al sumador 50 para generar datos residuales y al sumador 62 para reconstruir el bloque codificado para usarlo como un cuadro de referencia. La unidad de selección de modo 40 también proporciona elementos sintácticos, tales como vectores de movimiento, indicadores intramodo, información de partición y otra información sintáctica, a la unidad de codificación entrópica 56.

**[00113]** La unidad de estimación de movimiento 42 y la unidad de compensación de movimiento 44 pueden estar altamente integradas, pero se ilustran por separado para fines conceptuales. La estimación de movimiento, realizada por la unidad de estimación de movimiento 42, es el proceso de generación de vectores de movimiento, que estiman movimiento para los bloques de video. Por ejemplo, un vector de movimiento, puede indicar el desplazamiento de una PU de un bloque de video dentro de un cuadro de video o una imagen actuales con respecto a un bloque predictivo dentro de un cuadro de referencia (u otra unidad codificada) con respecto al bloque actual que se codifica dentro del cuadro actual (u otra unidad codificada). Un bloque predictivo es un bloque que coincide en gran medida con el bloque que se desea

codificar, en cuanto a diferencia de píxeles, que se puede determinar mediante la suma de la diferencia absoluta (SAD), suma de diferencia al cuadrado (SSD) u otras medidas de diferencia. En algunos ejemplos, el codificador de video 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 video 20 puede interpolar valores de posiciones de 1/4 píxel, posiciones de 1/8 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 con respecto a las posiciones de píxeles completos y posiciones de píxeles fraccionarios y extraer un vector de movimiento con precisión de píxeles fraccionarios.

**[00114]** La unidad de estimación de movimiento 42 calcula un vector de movimiento para una PU de un bloque de video en un segmento intercodificado mediante la comparación de la posición de la PU con la posición de un bloque predictivo de una imagen de referencia. La imagen de referencia se puede seleccionar de una primera lista de imágenes de referencia (Lista 0) o una segunda lista de imágenes de referencia (Lista 1), cada una de las cuales identifica una o más imágenes de referencia almacenadas en la memoria de imágenes de referencia 64. La unidad de estimación de movimiento 42 envía el vector de movimiento calculado a la unidad de codificación entrópica 56 y a la unidad de compensación de movimiento 44.

**[00115]** La compensación de movimiento, realizada por la unidad de compensación de movimiento 44, puede incluir buscar o generar el bloque predictivo en función del vector de movimiento determinado por la unidad de estimación de movimiento 42. Nuevamente, la unidad de estimación de movimiento 42 y la unidad de compensación de movimiento 44 pueden estar funcionalmente integradas, en algunos ejemplos. Al recibir el vector de movimiento para la PU del bloque de video actual, la unidad de compensación de movimiento 44 puede localizar el bloque predictivo al que el vector de movimiento señala en una de las listas de imágenes de referencia. El sumador 50 forma un bloque de video residual restando valores de píxeles del bloque predictivo de los valores de píxeles del bloque de video actual que se codifica, lo que genera valores de diferencia de píxeles, como se indica más adelante. En general, la unidad de estimación de movimiento 42 realiza la estimación de movimiento con respecto a componentes de luminancia, y la unidad de compensación de movimiento 44 usa vectores de movimiento calculados en función de los componentes de luminancia para los componentes de crominancia y componentes de luminancia. La unidad de

compensación de movimiento 44 puede generar un bloque de predicción usando los vectores de movimiento, que puede incluir interpolar o de lo contrario manipular matemáticamente valores de los bloques predictivos a los que hacen referencia los vectores de movimiento. La unidad de selección de modo 40 también puede producir elementos sintácticos asociados a los bloques de video y el segmento de video para usar con el decodificador de video 30 en la decodificación de los bloques de video del segmento de video.

**[00116]** El codificador de video 20 se puede configurar para llevar a cabo cualquiera de las diversas técnicas de la presente divulgación analizadas anteriormente con respecto a la Figura 6. Por ejemplo, la unidad de compensación de movimiento 44 se puede configurar para codificar información de movimiento para un bloque de datos de video usando AMVP o modo de fusión de acuerdo con HEVC, y/o se puede configurar para codificar información de movimiento afín o un bloque de datos de video usando intermodo afín o modo de fusión afín de acuerdo con las técnicas de la presente divulgación.

**[00117]** La unidad de intrapredicción 46 puede intraprededir 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 para codificar un bloque actual. En algunos ejemplos, la unidad de intrapredicción 46 puede codificar un bloque actual usando diversos modos de intrapredicción, por ejemplo, durante pases de codificación diferentes, y la unidad de intrapredicción 46 (o la unidad de selección de modo 40, en algunos ejemplos) puede seleccionar un modo de intrapredicción adecuado para usar de los modos evaluados.

**[00118]** Por ejemplo, la unidad de intrapredicción 46 puede calcular los valores de tasa-distorsión usando un análisis de tasa-distorsión para diversos modos de intrapredicción evaluados, y seleccionar el modo de intrapredicción que tiene las mejores características de tasa-distorsión entre los modos evaluados. En general, el análisis de tasa-distorsión determina una cantidad de distorsión (o error) entre un bloque codificado y un bloque original no codificado que se codificó para producir el bloque codificado, así como una tasa de bits (es decir, una cantidad de bits) usada para producir el bloque codificado. La unidad de intrapredicción 46 puede calcular relaciones de las distorsiones y las tasas para diversos bloques codificados para determinar qué modo de intrapredicción exhibe el mejor valor de tasa-distorsión para

el bloque.

**[00119]** Después de seleccionar un modo de intrapredicción para un bloque, la unidad de intrapredicción 46 puede proporcionar información indicativa del modo de intrapredicción seleccionado para el bloque a la unidad de codificación entrópica 56. La unidad de codificación entrópica 56 puede codificar la información que indica el modo de intrapredicción seleccionado. El codificador de video 20 puede incluir en los datos de la configuración de la corriente de bits transmitidos, que pueden incluir una pluralidad de tablas indexadas de modo de intrapredicción y una pluralidad de tablas indexadas de modo de intrapredicción modificado (también denominadas tablas de mapeo de *codeword*), definiciones de contextos de codificación para diversos bloques e indicaciones del modo de intrapredicción más probable, una tabla indexada de modo de intrapredicción y una tabla indexada de modo de intrapredicción modificado para usar en cada uno de los contextos.

**[00120]** El codificador de video 20 forma un bloque de video residual restando los datos de predicción de la unidad de selección de modo 40 del bloque de video original que se codifica. El sumador 50 representa el componente o los componentes que realizan esta operación de sustracción. La unidad de procesamiento de la transformada 52 aplica una transformada, tal como una transformada discreta del coseno (DCT) o una transformada conceptualmente similar, al bloque residual, lo que produce un bloque de video que comprende los valores del coeficiente de la transformada. Se pueden usar transformadas de ondícula, transformadas enteras, transformadas de subbanda, transformadas discretas del seno (DST) u otros tipos de transformadas en lugar de una DCT. En cualquier caso, la unidad de procesamiento de la transformada 52 aplica la transformada al bloque residual, lo que produce un bloque de coeficientes de la transformada. La transformada puede convertir la información residual de un dominio de píxel en un dominio de la transformada, tal como un dominio de frecuencia. La unidad de procesamiento de la transformada 52 puede enviar los coeficientes de la transformada resultantes a la unidad de cuantificación 54. La unidad de cuantificación 54 cuantifica los coeficientes de la transformada para reducir adicionalmente la tasa de bits. El proceso de cuantificación puede reducir la profundidad de bits asociada a algunos o todos los coeficientes. El grado de cuantificación se puede modificar ajustando un parámetro de cuantificación.

**[00121]** Luego de la cuantificación, la unidad de codificación entrópica 56 codifica por entropía los coeficientes de la transformada cuantificados. Por ejemplo, la unidad de codificación entrópica 56 puede realizar codificación adaptativa según el contexto de

longitud variable (CAVLC), codificación aritmética binaria adaptativa según el contexto (CABAC), codificación aritmética binaria adaptativa según el contexto en función de la sintaxis (SBAC), codificación entrópica de partición en intervalos de probabilidad (PIPE) u otra técnica de codificación entrópica. En el caso de codificación entrópica según el contexto, el contexto puede ser en función de bloques vecinos. Luego de la codificación entrópica mediante la unidad de codificación entrópica 56, la corriente de bits codificada se puede transmitir a otro dispositivo (por ejemplo, el decodificador de video 30) o archivar para la posterior transmisión o recuperación.

**[00122]** La unidad de cuantificación inversa 58 y la unidad de la transformada inversa 60 aplican cuantificación inversa y transformación inversa, respectivamente, para reconstruir el bloque residual en el dominio de píxeles. En particular, el sumador 62 agrega el bloque residual reconstruido al bloque de predicción de movimiento compensado que se produjo anteriormente mediante la unidad de compensación de movimiento 44 o unidad de intrapredicción 46 para producir un bloque de video reconstruido para almacenamiento en la memoria de imagen de referencia 64. La unidad de estimación de movimiento 42 y la unidad de compensación de movimiento 44 pueden usar el bloque de video reconstruido como un bloque de referencia para intercodificar un bloque en un cuadro de video posterior.

**[00123]** La Figura 8 es un diagrama de bloques que ilustra un ejemplo de decodificador de video 30 que puede implementar las técnicas de la presente divulgación para decodificar información de movimiento de predicción afín. En el ejemplo de la Figura 8, el decodificador de video 30 incluye unidad de decodificación entrópica 70, unidad de compensación de movimiento 72, unidad de intrapredicción 74, unidad de cuantificación inversa 76, unidad de la transformada inversa 78, memoria de imagen de referencia 82 y sumador 80. El decodificador de video 30 puede, en algunos ejemplos, realizar un pase de decodificación generalmente recíproco al pase de codificación descrito con respecto al codificador de video 20 (Figura 7). La unidad de compensación de movimiento 72 puede generar datos de predicción en función de vectores de movimiento recibidos de la unidad de codificación entrópica 70, mientras que la unidad de intrapredicción 74 puede generar datos de predicción en función de los indicadores de modo de intrapredicción recibidos de la unidad de decodificación entrópica 70.

**[00124]** Durante el proceso de decodificación, el decodificador de video 30 recibe una corriente de bits del video codificado que representa bloques de video de un segmento de video codificado y elementos sintácticos asociados del codificador de video 20. La

unidad de decodificación entrópica 70 del decodificador de video 30 decodifica por entropía la corriente de bits para generar coeficientes cuantificados, vectores de movimiento o indicadores de modo de intrapredicción, y otros elementos sintácticos. La unidad de decodificación entrópica 70 reenvía los vectores de movimiento y otros elementos sintácticos a la unidad de compensación de movimiento 72. El decodificador de video 30 puede recibir los elementos sintácticos al nivel del segmento de video y/o al nivel del bloque de video.

**[00125]** Cuando el segmento de video se codifica como un segmento intracodificado (I), la unidad de intrapredicción 74 puede generar datos de predicción para un bloque de video del segmento de video actual en función de un modo de intrapredicción señalizado y datos de los bloques decodificados previamente del cuadro o la imagen actuales. Cuando el cuadro de video 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 video del segmento de video actual en función de los vectores de movimiento y otros elementos sintácticos recibidos de la unidad de decodificación entrópica 70. Los bloques predictivos se pueden producir de una de las imágenes de referencia dentro de una de las listas de imágenes de referencia. El decodificador de video 30 puede construir las listas de cuadros de referencia, Lista 0 y Lista 1, usando técnicas de construcción predeterminadas en función de 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 video del segmento de video actual mediante el análisis de los vectores de movimiento y otros elementos sintácticos, y usa la información de predicción para producir bloques predictivos para el bloque de video actual que se decodifica. Por ejemplo, la unidad de compensación de movimiento 72 usa algunos de los elementos sintácticos recibidos para determinar un modo de predicción (por ejemplo, intra- o interpredicción) usado para codificar los bloques de video del segmento de video, un tipo de segmento de interpredicción (por ejemplo, segmento B o segmento P), información de construcción para una o más de las listas de imágenes de referencia para el segmento, vectores de movimiento para cada bloque de video intercodificado del segmento, 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.

**[00126]** El decodificador de video 30 se puede configurar para llevar a cabo cualquiera de las diversas técnicas de la presente divulgación analizadas

anteriormente con respecto a la Figura 6. Por ejemplo, la unidad de compensación de movimiento 72 se puede configurar para llevar a cabo predicción de vectores de movimiento usando AMVP o modo de fusión de acuerdo con HEVC, y/o se puede configurar para llevar a cabo información de movimiento afín o un bloque de datos de video usando intermodo afín o modo de fusión afín de acuerdo con las técnicas de la presente divulgación. La unidad de decodificación entrópica 70 puede decodificar uno o más elementos sintácticos que representan cómo se codifica la información de movimiento (por ejemplo, información de movimiento afín) para el bloque actual.

**[00127]** La unidad de compensación de movimiento 72 también puede realizar interpolación en función de filtros de interpolación. La unidad de compensación de movimiento 72 puede usar filtros de interpolación como los utiliza 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 usados por el codificador de video 20 de los elementos sintácticos recibidos y usar los filtros de interpolación para producir bloques predictivos.

**[00128]** La unidad de cuantificación inversa 76 cuantifica de manera inversa, es decir, de-cuantifica, los coeficientes de la transformada cuantificados proporcionados en la corriente de bits y decodificados por la unidad de decodificación entrópica 70. El proceso de cuantificación inversa puede incluir el uso de un parámetro de cuantificación  $QP_Y$  calculado por 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 se debería aplicar.

**[00129]** La unidad de la transformada inversa 78 aplica una transformada inversa, por ejemplo, una DCT inversa, una transformada entera inversa o un proceso de la transformada inversa conceptualmente similar, a los coeficientes de la transformada con el fin de producir bloques residuales en el dominio de píxeles.

**[00130]** Después de que la unidad de compensación de movimiento 72 genera el bloque predictivo para el bloque de video actual en función de los vectores de movimiento y otros elementos sintácticos, el decodificador de video 30 forma un bloque de video decodificado sumando los bloques residuales de la unidad de la transformada inversa 78 con los correspondientes bloques predictivos generados por la unidad de compensación de movimiento 72. El sumador 80 representa el componente o los componentes que realizan esta operación de suma. Si se desea, un filtro de eliminación de bloques también se puede aplicar para filtrar los bloques

decodificados con el fin de eliminar los artefactos de compresión. Otros filtros en bucle (ya sea en el bucle de codificación o después del bucle de codificación) también se pueden usar para suavizar las transiciones de píxeles o, de otro modo, mejorar la calidad de video. Los bloques de video decodificados en un cuadro o imagen determinados luego se almacenan en la memoria de imágenes de referencia 82, que almacena imágenes de referencia utilizadas para la posterior compensación de movimiento. La memoria de imágenes de referencia 82 también almacena videos decodificados para la posterior presentación en un dispositivo de visualización, tal como un dispositivo de visualización 32 de la Figura 6.

**[00131]** La Figura 11 es un diagrama de flujo que ilustra un método de ejemplo para codificar un bloque actual de datos de video de acuerdo con las técnicas de la presente divulgación. A modo de ejemplo y explicación, el método de la Figura 11 se explica con respecto al codificador de video 20 de las Figuras 6 y 7. Sin embargo, cabe destacar que otros dispositivos se pueden configurar para llevar a cabo este método o uno similar.

**[00132]** 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 asume que la unidad de selección de modo 40 selecciona un modo de predicción afín, que incluye predicción mediante el uso de al menos dos vectores de movimiento. Por ende, la unidad de selección de modo 40 ocasiona que la unidad de estimación de movimiento 42 lleve a cabo 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 tasa-distorsión entre una variedad de métodos de predicción, tales como intrapredicción, interpredicción, y predicción afín, y determinar que la predicción afín da como resultado los mejores resultados de tasa-distorsión entre los diversos modos de predicción evaluados.

**[00133]** Luego de que la unidad de estimación de movimiento 42 determina el primer y segundo vector de movimiento, el codificador de video 20 puede calcular un primer predictor de vectores de movimiento (MVP) (102) para el primer vector de movimiento. El primer MVP puede corresponder a un vector de movimiento de un bloque vecino. Entonces, el codificador de video 20 puede calcular una primera diferencia de vectores de movimiento (MVD) como la diferencia entre el primer vector de movimiento y el primer predictor de vectores de movimiento (104). En particular, el codificador de video 20 puede calcular diferencias entre los componentes x e y del vector de movimiento y

la MVD, respectivamente.

**[00134]** Entonces, el codificador de video 20 puede determinar un segundo MVP (106) para el segundo vector de movimiento. El codificador de video 20 también puede calcular una segunda MVD (108) como la diferencia entre el segundo vector de movimiento y la segunda MVD.

**[00135]** La unidad de compensación de movimiento 44 también puede predecir el bloque actual (110) usando predicción afín para generar un bloque de predicción para el bloque actual. Aunque dos vectores de movimiento se analizaron a modo de ejemplo, cabe destacar que tres vectores de movimiento se pueden usar para la predicción afín para generar el bloque de predicción. De igual modo, el codificador de video 20 puede generar una tercera MVD y un tercer MVP de acuerdo con las técnicas de la presente divulgación para un tercer vector de movimiento, como se analizó anteriormente.

**[00136]** Luego de generar el bloque de predicción, el codificador de video 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. Entonces, el codificador de video 20 puede 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 la presente divulgación, el codificador de video 20 puede codificar la primera MVD usando, por ejemplo, técnicas de predicción de vectores de movimiento, tales como modo de fusión o modo de AMVP, y codificar la segunda MVD mediante predicción de la segunda MVD de la primera MVD. Por ende, para codificar la segunda MVD, el codificador de video 20 puede codificar datos representativos de una diferencia entre la primera MVD y la segunda MVD, tal como 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 la transformada 52 puede transformar el bloque residual, la unidad de cuantificación 54 puede cuantificar coeficientes de la transformada del bloque de la transformada resultante, y la unidad de codificación entrópica 56 puede codificar por entropía los coeficientes de la transformada cuantificados resultantes.

**[00137]** De este modo, el método de la Figura 11 representa un ejemplo de un método que incluye codificar (a saber, 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 video predicho usando predicción afín y

un primer predictor de vectores de movimiento (MVP) para el primer vector de movimiento; predecir una segunda MVD de la primera MVD para un segundo vector de movimiento del bloque actual; y codificar (a saber, codificar) el bloque actual usando predicción afín de acuerdo con el primer vector de movimiento y el segundo vector de movimiento. Llevando a cabo el método de la Figura 11, el codificador de video 20 puede generar una corriente de bits que es más eficaz para el ancho de banda que cuando se llevan a cabo técnicas previas, porque los datos que representan a la segunda MVD pueden ser más pequeños porque la segunda MVD se predice de la primera MVD.

**[00138]** La Figura 12 es un diagrama de flujo que ilustra un método de ejemplo para decodificar un bloque actual de datos de video de acuerdo con las técnicas de la presente divulgación. El método de la Figura 12 se explica con respecto al decodificador de video 30 de las Figuras 6 y 8 a modo de ejemplo. Sin embargo, cabe destacar que otros dispositivos se pueden configurar para llevar a cabo las técnicas de este método o uno similar.

**[00139]** El decodificador de video 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 la presente divulgación, el decodificador de video 30 puede decodificar la primera MVD usando, por ejemplo, técnicas de predicción de vectores de movimiento, tales como modo de fusión o modo de AMVP, y decodificar la segunda MVD mediante predicción de la segunda MVD de la primera MVD. Por ende, para decodificar la segunda MVD, el decodificador de video 30 puede decodificar datos representativos de una diferencia entre la primera MVD y la segunda MVD, tal como diferencias entre los componentes x e y de la primera MVD y la segunda MVD. Para decodificar el bloque residual, la unidad de decodificación entrópica 70 puede decodificar por entropía coeficientes de la transformada cuantificados, la unidad de cuantificación inversa 76 puede cuantificar de manera inversa los coeficientes de la transformada, y unidad de la transformada inversa 78 puede transformar de manera inversa los coeficientes de la transformada para reproducir el bloque residual.

**[00140]** Entonces, la unidad de compensación de movimiento 72 puede determinar un primer predictor de vectores de movimiento (MVP) (122) para un primer vector de movimiento del bloque actual y calcular el primer vector de movimiento (124) del primer MVP. En particular, la unidad de compensación de movimiento 72 puede sumar la primera MVD al primer MVP para calcular el primer vector de movimiento. De

manera similar, la unidad de compensación de movimiento 72 puede determinar un segundo MVP (126) para un segundo vector de movimiento del bloque actual y calcular el segundo vector de movimiento (128) del segundo MVP. En particular, la unidad de compensación de movimiento 72 puede sumar la segunda MVD al segundo MVP para calcular el segundo vector de movimiento. En algunos ejemplos, un tercer vector de movimiento se puede incluir, en cuyo caso la unidad de decodificación entrópica 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 del tercer MVP y un tercer MVP de una forma similar.

**[00141]** Entonces, la unidad de compensación de movimiento 72 puede predecir el bloque actual (130), por ejemplo, de acuerdo con predicción de movimiento afín usando el primer y segundo (y potencialmente tercer) vector de movimiento. Entonces, el decodificador de video 30 puede decodificar el bloque actual (132), por ejemplo, ocasionando que el sumador 80 sume los valores del bloque de predicción a los valores del bloque residual píxel por píxel.

**[00142]** De este modo, el método de la Figura 12 representa un ejemplo de un método que incluye codificar (a saber, 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 video predicho usando predicción afín y un primer predictor de vectores de movimiento (MVP) para el primer vector de movimiento; predecir una segunda MVD de la primera MVD para un segundo vector de movimiento del bloque actual; y codificar (a saber, decodificar) el bloque actual usando predicción afín de acuerdo con el primer vector de movimiento y el segundo vector de movimiento. Llevando a cabo el método de la Figura 12, el decodificador de video 30 puede decodificar una corriente de bits que es más eficaz para el ancho de banda que cuando se llevan a cabo técnicas previas, porque los datos que representan a la segunda MVD pueden ser más pequeños porque la segunda MVD se predice de la primera MVD.

**[00143]** Cabe aclarar que según el ejemplo, algunos actos o eventos de cualquiera de las técnicas descritas en la presente se pueden realizar en una secuencia diferente, se pueden agregar, fusionar o excluir completamente (por ejemplo, no todos los actos o eventos descritos son necesarios para la práctica de las técnicas). Asimismo, en algunos ejemplos, los actos o eventos se pueden realizar de manera concurrente, por ejemplo, a través de procesamiento multihilo, procesamiento interrumpido o

procesadores múltiples, en lugar de secuencialmente.

**[00144]** En uno o más ejemplos, las funciones descritas pueden implementarse en hardware, software, firmware o cualquier combinación de estos. Si se implementan en software, las funciones pueden almacenarse o transmitirse como una o más instrucciones o código en un medio legible por computadora y pueden ejecutarse mediante una unidad de procesamiento de hardware. Los medios legibles por computadora pueden incluir medios de almacenamiento legibles por computadora, lo que corresponde a un medio tangible, tal como medios de almacenamiento de datos o medios de comunicación que incluyen cualquier medio que facilita la transferencia de un programa de computadora desde un lugar a otro, por ejemplo, de acuerdo con un protocolo de comunicación. De esta manera, los medios legibles por computadora generalmente pueden corresponder a (1) un medio de almacenamiento tangible legible por computadora que no sea transitorio o (2) un medio de comunicación, tal como una señal o una onda portadora. Los medios de almacenamiento de datos pueden ser cualquier medio disponible al que se puede acceder mediante una o más computadoras o uno o más procesadores para obtener instrucciones, códigos y/o estructuras de datos para la implementación de las técnicas descritas en la presente divulgación. Un producto de programa de computación puede incluir un medio legible por computadora.

**[00145]** A modo ejemplificativo, y no limitativo, tal medio de almacenamiento legible por computadora puede comprender RAM, ROM, EEPROM, CD-ROM u otros dispositivos de almacenamiento en disco óptico, de almacenamiento en disco magnético u otro almacenamiento magnético, o cualquier otro medio que pueda utilizarse para almacenar el código de programación deseado en forma de instrucciones o estructuras de datos y al que puede accederse por medio de una computadora. Además, cualquier conexión se conoce con el término medio legible por computadora. Por ejemplo, si las instrucciones son transmitidas desde una página web, servidor u otra fuente remota que utilice un cable coaxial, cable de fibra óptica, par trenzado, línea de abonado digital (DSL), o tecnologías inalámbricas tales como infrarrojo, radio y microondas, entonces el cable coaxial, cable de fibra óptica, par trenzado, DSL, o tecnologías inalámbricas tales como infrarrojo, radio, y microondas se incluyen en la definición de medio. Sin embargo, cabe aclarar que los medios de almacenamiento legibles por computadora no incluyen conexiones, ondas portadoras, señales u otros medios transitorios, sino que, en su lugar, se refieren a medios de almacenamiento tangibles no transitorios. Disquete y disco, como se usa en la

presente, incluye disco compacto (CD), disco láser, disco óptico, disco digital versátil (DVD), disco flexible y disco Blu-ray, en donde los disquetes reproducen datos en forma magnética, mientras que los discos reproducen datos en forma óptica con láser. También deberían incluirse combinaciones de lo anterior dentro del alcance de medios legibles por computadora.

**[00146]** Las instrucciones pueden ser ejecutadas por uno o más procesadores, tales como uno o más procesadores de señal digital (DSP), procesadores de propósito general, circuitos integrados para aplicaciones específicas (ASIC), matrices de compuerta programable en campo (FPGA) u otro circuito lógico integrado o separado equivalente. En consecuencia, como se usa en la presente, el término “procesador” puede referirse a cualquiera de las estructuras anteriores o a cualquier otra estructura adecuada para la implementación de las técnicas descritas en la presente. Además, en algunos aspectos, la funcionalidad descrita en la presente puede estar proporcionada dentro de módulos de hardware y/o software dedicados configurados para codificar y decodificar, o puede estar incorporada en un códec combinado. Además, las técnicas podrían implementarse totalmente en uno o más circuitos o elementos lógicos.

**[00147]** Las técnicas de la presente divulgación se pueden implementar en una amplia variedad de dispositivos o aparatos, que incluyen un microteléfono inalámbrico, un circuito integrado (IC) o un conjunto de IC (por ejemplo, un conjunto de chips). En la presente divulgación se describen distintos componentes, módulos o unidades para enfatizar los aspectos funcionales de los dispositivos configurados para realizar las técnicas descritas, pero no es necesario que sean realizadas por diferentes unidades de hardware. En su lugar, como se describió anteriormente, distintas unidades pueden combinarse en una unidad de hardware de códec o pueden ser proporcionadas por un grupo de unidades de hardware interoperativas, que incluyen uno o más procesadores, como se describió anteriormente, junto con software y/o firmware adecuado.

**[00148]** Se han descrito distintos ejemplos. Estos y otros ejemplos se encuentran dentro del alcance de las siguientes reivindicaciones.