(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 988 146**

(51) Int. Cl.:

H04N 19/436 (2014.01)
H04N 19/17 (2014.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **09.01.2020 PCT/US2020/012862**

(87) Fecha y número de publicación internacional: **16.07.2020 WO20146582**

(96) Fecha de presentación y número de la solicitud europea: **09.01.2020 E 20738533 (7)**

(97) Fecha y número de publicación de la concesión europea: **19.06.2024 EP 3906684**

(54) Título: **Señalización de distribución de subimágenes en codificación de vídeo**

(30) Prioridad:

09.01.2019 US 201962790207 P

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
19.11.2024

(73) Titular/es:

HUAWEI TECHNOLOGIES CO., LTD. (100.0%)
Huawei Administration Building, Bantian, Longgang District
Shenzhen, Guangdong 518129, CN

(72) Inventor/es:

WANG, YE-KUI y
HENDRY, FNU

(74) Agente/Representante:

LEHMANN NOVO, María Isabel

DESCRIPCIÓN

Señalización de distribución de subimágenes en codificación de vídeo

CAMPO TÉCNICO DE LA INVENCIÓN

La presente divulgación está relacionada generalmente con la codificación de video y está específicamente relacionada con la gestión de subimágenes en la codificación de video.

ANTECEDENTES DE LA INVENCIÓN

La cantidad de datos de video necesarios para representar incluso un video relativamente corto puede ser sustancial, lo que puede resultar en dificultades cuando los datos deben transmitirse o comunicarse a través de una red de comunicaciones con capacidad de ancho de banda limitada. Por lo tanto, los datos de video generalmente se comprimen antes de comunicarse a través de las redes de telecomunicaciones de hoy en día. El tamaño de un video también podría ser un problema cuando el video se almacena en un dispositivo de almacenamiento porque los recursos de memoria pueden ser limitados. Los dispositivos de compresión de video a menudo usan software y/o hardware en la fuente/origen para codificar los datos de video antes de su transmisión o almacenamiento, disminuyendo así la cantidad de datos necesarios para representar imágenes de video digitales. Los datos comprimidos son luego recibidos en el destino por un dispositivo de descompresión de video que decodifica los datos de video. Con recursos de red limitados y demandas cada vez mayores de mayor calidad de video, son deseables técnicas mejoradas de compresión y descompresión que mejoren la relación de compresión con poco o ningún sacrificio en la calidad de la imagen.

El documento US 2014/003504 A1 proporciona un método, aparato y producto de programa informático para codificación y decodificación de vídeo escalable.

El documento COBAN M ET AL: "AHG4: Support of independent sub-pictures", 9. REUNIÓN DEL JCT-VC; 100. REUNIÓN DEL MPEG; 27-4-2012 - 7-5-2012; GINEBRA; (EQUIPO COLABORATIVO MIXTO SOBRE LA CODIFICACIÓN DE VÍDEO DE ISO/IEC JTC1/SC29/WG11 E ITU-T SG.16); n.° JCTVC-10356, 27 de abril de 2012 (27-04-2012), presenta el concepto de soportar subimágenes en HEVC.

El documento ANDREW SEGALL (SHARPLABS): "BoG report on high level parallelization", 9. REUNIÓN DEL JCT-VC; 20120427 - 20120507; GINEBRA; (EQUIPO COLABORATIVO MIXTO SOBRE LA CODIFICACIÓN DE VÍDEO DE ISO/IEC JTC1/SC29/WG11 E ITU-T SG.16), n.° JCTVC-10581 7 de mayo de 2012 (07-05-2012), documento XP030234344 divulga notas de reunión para paralelización de alto nivel.

BREVE DESCRIPCIÓN DE LA INVENCIÓN

Las modalidades de la presente invención se definen mediante las reivindicaciones independientes. Se presentan características adicionales de las modalidades de la invención en las reivindicaciones dependientes. A continuación, las partes de la descripción y los dibujos que hacen referencia a realizaciones anteriores que no comprenden necesariamente todas las características para implementar las modalidades de la invención reivindicada no se representan como modalidades de la invención, sino como ejemplos útiles para entender las modalidades de la invención.

En una modalidad, la divulgación incluye un método implementado en un decodificador, el método que comprende: recibir, por un receptor del decodificador, un flujo de bits que comprende unas subimágenes particionadas de una imagen y un conjunto de parámetros de secuencia (SPS) que comprende parámetros, en donde los parámetros comprenden, un tamaño de subimagen para cada una de las subimágenes y ubicación de subimagen para cada una de las subimágenes, en donde el SPS comprende además un identificador (ID) de subimagen para cada una de las subimágenes; analizar, mediante un procesador del decodificador, el SPS para obtener el ID de subimagen, el tamaño de subimagen y la ubicación de subimagen para cada una de las subimágenes; generar, por el procesador, las subimágenes en base al ID de subimagen, el tamaño de subimagen y la ubicación de subimagen para cada una de las subimágenes para crear una secuencia de vídeo. Algunos sistemas incluyen información de mosaico e información de subimagen en un conjunto de parámetros de imagen (PPS), ya que los mosaicos y las subimágenes son más pequeños que una imagen. Sin embargo, las subimágenes pueden usarse para soportar aplicaciones de región de interés (ROI) y esquemas de acceso basados en subimágenes. Estas aplicaciones no cambian en una base por imagen. Los ejemplos divulgados incluyen la información de diseño para subimágenes en un SPS en lugar de un PPS. La información de diseño de la subimagen incluye la ubicación de subimagen y el tamaño de subimagen. La ubicación de subimagen es un desplazamiento entre la muestra superior izquierda de la subimagen y la muestra superior izquierda de la imagen. El tamaño de subimagen es la altura y el ancho de la subimagen medidos en muestras de luma. Una secuencia de video puede incluir un solo SPS (o uno por segmento de video) y puede incluir hasta un PPS por imagen. La colocación de información de diseño para subimágenes en el SPS garantiza que el diseño solo se señalice una vez para una secuencia/segmento en lugar de señalizarlo de forma redundante para cada PPS. Por consiguiente, la disposición de la subimagen de señalización en el SPS aumenta la eficacia de la codificación y, por tanto, reduce el uso de recursos de red, recursos de memoria y/o recursos de procesamiento en el codificador y el decodificador. Además, algunos sistemas tienen la información de subimagen derivada por el decodificador. La señalización de la información de subimagen reduce la posibilidad de error

en caso de pérdida de paquetes y admite una funcionalidad adicional en términos de extracción de subimágenes. En consecuencia, la disposición de subimagen de señalización en el SPS mejora la funcionalidad de un codificador y/o decodificador.

Opcionalmente, en cualquiera de los aspectos anteriores, otra implementación del aspecto proporciona, en el que la ubicación de subimagen incluye una distancia de desplazamiento entre una muestra superior izquierda de la subimagen y una muestra superior izquierda de la imagen.

Opcionalmente, en cualquiera de los aspectos anteriores, otra implementación del aspecto proporciona, en el que el tamaño de subimagen incluye una altura de subimagen en las muestras de luma y un ancho de subimagen en las muestras de luma.

En una modalidad, la divulgación incluye un método implementado en un codificador, comprendiendo el método: codificar, mediante un procesador del codificador, subimágenes, particionadas a partir de una imagen, en un flujo de bits; codificar, mediante el procesador, un identificador (ID) de subimagen, un tamaño de subimagen y una ubicación de subimagen para cada una de las subimágenes en un SPS en el flujo de bits. Opcionalmente, en cualquiera de los aspectos anteriores, otra implementación del aspecto proporciona, que comprende además almacenar, en una memoria del codificador, el flujo de bits para la comunicación hacia un decodificador. Algunos sistemas incluyen información de mosaico e información de subimagen en un conjunto de parámetros de imagen (PPS), ya que los mosaicos y las subimágenes son más pequeños que una imagen. Sin embargo, las subimágenes pueden usarse para soportar aplicaciones de región de interés (ROI) y esquemas de acceso basados en subimágenes. Estas aplicaciones no cambian en una base por imagen. Los ejemplos divulgados incluyen la información de diseño para subimágenes en un SPS en lugar de un PPS. La información de diseño de la subimagen incluye la ubicación de subimagen y el tamaño de subimagen. La ubicación de subimagen es un desplazamiento entre la muestra superior izquierda de la subimagen y la muestra superior izquierda de la imagen. El tamaño de subimagen es la altura y el ancho de la subimagen medidos en muestras de luma. Una secuencia de video puede incluir un solo SPS (o uno por segmento de video) y puede incluir hasta un PPS por imagen. La colocación de información de diseño para subimágenes en el SPS garantiza que el diseño solo se señalice una vez para una secuencia/segmento en lugar de señalizarlo de forma redundante para cada PPS. Por consiguiente, la disposición de la subimagen de señalización en el SPS aumenta la eficacia de la codificación y, por tanto, reduce el uso de recursos de red, recursos de memoria y/o recursos de procesamiento en el codificador y el decodificador. Además, algunos sistemas tienen la información de subimagen derivada por el decodificador. La señalización de la información de subimagen reduce la posibilidad de error en caso de pérdida de paquetes y admite una funcionalidad adicional en términos de extracción de subimágenes. En consecuencia, la disposición de subimagen de señalización en el SPS mejora la funcionalidad de un codificador y/o decodificador.

En una modalidad, la divulgación incluye un dispositivo de codificación de video que comprende: un procesador, una memoria, un receptor acoplado al procesador y un transmisor acoplado al procesador, el procesador, la memoria, el receptor y el transmisor configurados para realizar el método de cualquiera de los aspectos anteriores.

En una modalidad, la divulgación incluye un medio legible por computadora no transitorio que comprende un producto de programa de computadora para su uso por un dispositivo de codificación de video, el producto de programa de computadora comprende instrucciones ejecutables por computadora almacenadas en el medio legible por computadora no transitorio de tal manera que cuando se ejecuta por un procesador hace que el dispositivo de codificación de vídeo realice el método de cualquiera de los aspectos anteriores.

En una modalidad, la divulgación incluye un decodificador que comprende: medios de recepción para recibir un flujo de bits que comprende subimágenes particionadas a partir de una imagen y un SPS que comprende parámetros, en donde los parámetros comprenden un tamaño de subimagen para cada una de las subimágenes y una ubicación de subimagen para cada una de las subimágenes, en donde el SPS comprende además un identificador (ID) de subimagen para cada una de las subimágenes; medios de análisis para analizar el SPS para obtener el ID de subimagen, el tamaño de subimagen y la ubicación de subimagen para cada una de las subimágenes; medios de decodificación para decodificar la subimagen en base al ID de subimagen, el tamaño de subimagen y la ubicación de subimagen para cada una de las subimágenes para crear una secuencia de vídeo.

Opcionalmente, en cualquiera de los aspectos anteriores, otra implementación del aspecto proporciona, en el que el decodificador está configurado además para realizar el método de cualquiera de los aspectos anteriores.

En una modalidad, la divulgación incluye un codificador que comprende: medios de codificación para: codificar subimágenes, particionadas a partir de una imagen, en un flujo de bits; y codificar un identificador (ID) de subimagen, un tamaño de subimagen y una ubicación de subimagen para cada una de las subimágenes en un SPS en el flujo de bits.

Opcionalmente, en cualquiera de los aspectos anteriores, otra implementación del aspecto proporciona, en el que el codificador está configurado además para realizar el método de cualquiera de los aspectos anteriores.

Para mayor claridad, cualquiera de las modalidades anteriores puede combinarse con una o más de las otras modalidades anteriores para crear una nueva realización dentro del alcance de la presente divulgación.

Estas y otras características se entenderán más claramente a partir de la siguiente descripción detallada tomada junto con los dibujos y reivindicaciones adjuntos.

BREVE DESCRIPCIÓN DE LOS DIBUJOS

Para una comprensión más completa de esta divulgación, se hace ahora referencia a la siguiente breve descripción, tomada en relación con los dibujos adjuntos y la descripción detallada, en donde los números de referencia similares representan partes similares.

La FIG. 1 es un diagrama de flujo de un método de ejemplo de codificación de señal de video.

La FIG. 2 es un diagrama esquemático de un ejemplo de sistema de codificación y decodificación (códec) para codificación de vídeo.

La FIG. 3 es un diagrama esquemático que ilustra un codificador de vídeo de ejemplo.

La FIG. 4 es un diagrama esquemático que ilustra un decodificador de video de ejemplo.

La FIG. 5 es un diagrama esquemático que ilustra un flujo de bits y un subflujo de bits de ejemplo extraídos del flujo de bits.

La FIG. 6 es un diagrama esquemático que ilustra un ejemplo de imagen particionada en subimágenes.

La FIG. 7 es un diagrama esquemático que ilustra un mecanismo de ejemplo para relacionar segmentos con un diseño de subimagen.

La FIG. 8 es un diagrama esquemático que ilustra otra imagen de ejemplo particionada en subimágenes.

La FIG. 9 es un diagrama esquemático de un dispositivo de codificación de video de ejemplo.

La FIG. 10 es un diagrama de flujo de un método de ejemplo de codificación de un diseño de subimagen en un flujo de bits de imágenes para soportar la extracción de subimágenes.

La FIG. 11 es un diagrama de flujo de un método de ejemplo de decodificación de un flujo de bits de subimágenes en base a un diseño de subimagen señalizado.

La FIG. 12 es un diagrama esquemático de un sistema de ejemplo para señalizar un diseño de subimagen a través de un flujo de bits.

DESCRIPCIÓN DETALLADA

Debe entenderse desde el principio que aunque a continuación se proporciona una implementación ilustrativa de una o más modalidades, los sistemas y/ o métodos descritos pueden implementarse usando cualquier número de técnicas, ya sean conocidas actualmente o existentes. La divulgación no debe limitarse de ninguna manera a las implementaciones ilustrativas, dibujos y técnicas ilustradas a continuación, incluidos los diseños e implementaciones ejemplares ilustrados y descritos en este documento, pero puede modificarse dentro del alcance de las reivindicaciones adjuntas junto con su alcance completo de equivalentes.

Aquí se emplean varios acrónimos, como bloque de árbol de codificación (CTB), unidad de árbol de codificación (CTU), unidad de codificación (CU), secuencia de video codificado (CVS), equipo conjunto de expertos en video (JVET), conjunto de mosaicos con restricción de movimiento (MCTS), unidad de transferencia máxima (MTU), capa de abstracción de red (NAL), recuento/conteo de orden de imágenes (POC), carga útil de secuencia de bytes sin procesar (RBSP), conjunto de parámetros de secuencia (SPS), codificación de video versátil (WC) y borrador de trabajo (WD).

Se pueden emplear muchas técnicas de compresión de video para reducir el tamaño de los archivos de video con una pérdida mínima de datos. Por ejemplo, las técnicas de compresión de vídeo pueden incluir la realización de una predicción espacial (por ejemplo, intra-imagen) y/o temporal (por ejemplo, inter-imagen) para reducir o eliminar la redundancia de datos en las secuencias de vídeo. Para la codificación de vídeo basada en bloques, un segmento de vídeo (por ejemplo, una imagen de vídeo o una porción de una imagen de vídeo) puede dividirse en bloques de vídeo, que también pueden denominarse bloques de árbol, bloques de árbol de codificación (CTBs), unidades de árbol de codificación (CTUs), unidades de codificación (CUs) y/o nodos de codificación. Los bloques de vídeo de una porción intracodificada (I) de una imagen se codifican utilizando la predicción espacial con respecto a las muestras de referencia de los bloques vecinos de la misma imagen. Los bloques de vídeo en un segmento de predicción unidireccional (P) o bidireccional (B) inter-codificado de una imagen pueden codificarse empleando la predicción espacial con respecto a las muestras de referencia de los bloques vecinos de la misma imagen o la predicción temporal con respecto a las muestras

de referencia de otras imágenes de referencia. Las imágenes pueden denominarse fotogramas y/o imágenes, y las imágenes de referencia pueden denominarse fotogramas de referencia y/o imágenes de referencia. La predicción espacial o temporal da como resultado un bloque predictivo que representa un bloque de imagen. Los datos residuales representan las diferencias de píxeles entre el bloque de imagen original y el bloque predictivo. En consecuencia, un bloque intercodificado se codifica según un vector de movimiento que apunta a un bloque de muestras de referencia que forman el bloque predictivo y los datos residuales que indican la diferencia entre el bloque codificado y el bloque predictivo. Un bloque intracodificado se codifica según un modo de intracodificación y los datos residuales. Para una compresión adicional, los datos residuales se pueden transformar del dominio de píxeles a un dominio de transformación. Estos dan como resultado coeficientes de transformación residuales, que pueden cuantificarse. Los coeficientes de transformación cuantificados pueden disponerse inicialmente en un arreglo bidimensional. Los coeficientes de transformación cuantificados se pueden escanear para producir un vector unidimensional de coeficientes de transformación. Se puede aplicar codificación por entropía para lograr una compresión aún mayor. Dichas técnicas de compresión de video se describen con mayor detalle a continuación.

Para garantizar que un video codificado se pueda decodificar con precisión, el video se codifica y decodifica de acuerdo con los estándares de codificación de video correspondientes. Los estándares de codificación de video incluyen el Sector de Normalización de la Unión Internacional de Telecomunicaciones (UIT) (UIT-T) H.261, Grupo de Expertos en Cinematografía (MPEG) -1 Parte 2 de la Organización Internacional de Normalización / Comisión Electrotécnica Internacional (ISO / IEC), UIT-T H.262 o ISO / IEC MPEG-2 Parte 2, ITU-T H.263, ISO / IEC MPEG-4 Parte 2, Codificación de video avanzada (AVC), también conocida como ITU-T H.264 o ISO / IEC MPEG -4 Parte 10 y Codificación de video de alta eficiencia (HEVC), también conocida como ITU-T H.265 o MPEG-H Parte 2. AVC incluye extensiones como Codificación de video escalable (SVC), Codificación de video de vista múltiple (MVC) y Codificación de video de vista múltiple más profundidad (MVC+D), y AVC tridimensional (3D) (3D-AVC). HEVC incluye extensiones como HEVC escalable (SHVC), HEVC de vista múltiple (MV-HEVC) y HEVC 3D (3D-HEVC). El equipo conjunto de expertos en video (JVET) de ITU-T e ISO / IEC ha comenzado a desarrollar un estándar de codificación de video denominado Codificación de video versátil (WC). WC se incluye en un borrador de trabajo (WD), que incluye JVET-L1001-v9.

Para codificar una imagen de video, primero se particiona la imagen y las particiones se codifican en un flujo de bits. Se encuentran disponibles varios esquemas de partición de imágenes. Por ejemplo, una imagen se puede dividir en segmentos/cortes regulares, segmentos dependientes, mosaicos y/o de acuerdo con Procesamiento paralelo del frente de onda (WPP). En aras de la simplicidad, HEVC restringe los codificadores de modo que solo se puedan usar segmentos regulares, segmentos dependientes, mosaicos, WPP y combinaciones de los mismos al particionar un segmento en grupos de CTBs para la codificación de video. Dicha partición se puede aplicar para admitir la coincidencia de tamaño de la Unidad de transferencia máxima (MTU), el procesamiento en paralelo y el retardo reducido de un extremo a otro. MTU denota la cantidad máxima de datos que se pueden transmitir en un solo paquete. Si la carga útil de un paquete excede la MTU, esa carga útil se divide en dos paquetes a través de un proceso llamado fragmentación.

Un segmento regular, también denominado simplemente segmento, es una porción particionada de una imagen que se puede reconstruir independientemente de otros segmentos regulares dentro de la misma imagen, a pesar de algunas interdependencias debidas a las operaciones de filtrado de bucle. Cada segmento regular se encapsula en su propia unidad de capa de abstracción de red (NAL) para su transmisión. Además, la predicción en imagen (predicción intramuestra, predicción de información de movimiento, predicción del modo de codificación) y la dependencia de la codificación de entropía a través de los límites de los segmentos pueden desactivarse para soportar la reconstrucción independiente. Tal reconstrucción independiente apoya la paralelización. Por ejemplo, la paralelización basada en segmentos regulares emplea una comunicación mínima entre procesadores o entre núcleos. Sin embargo, como cada sector regular es independiente, cada sector se asocia con un encabezado de sector separado. El uso de segmentos regulares puede incurrir en una sobrecarga de codificación sustancial debido al costo de bits del encabezado de segmento para cada segmento y debido a la falta de predicción a través de los límites del segmento. Además, se pueden emplear segmentos regulares para soportar la coincidencia con los requisitos de tamaño de MTU. Específicamente, como un segmento regular se encapsula en una unidad NAL separada y se puede codificar de forma independiente, cada segmento regular debe ser más pequeño que la MTU en los esquemas MTU para evitar dividir el segmento en múltiples paquetes. Como tal, el objetivo de la paralelización y el objetivo de la coincidencia del tamaño de MTU pueden plantear demandas contradictorias para un diseño de segmentos en una imagen.

Los segmentos dependientes son similares a los segmentos regulares, pero tienen encabezados de segmentos más cortos y permiten la partición de los límites del bloque de árbol de la imagen sin romper la predicción en imagen. En consecuencia, los segmentos dependientes permiten que un segmentos regular se fragmente en múltiples unidades NAL, lo que proporciona un retardo de extremo a extremo reducido al permitir que se envíe una parte de un segmentos regular antes de que se complete la codificación de todo el segmentos regular.

Un mosaico es una parte particionada de una imagen creada por límites horizontales y verticales que crean columnas y filas de mosaicos. Los mosaicos se pueden codificar en orden de exploración de trama (de derecha a izquierda y de arriba a abajo). El orden de exploración de los CTBs es local dentro de un mosaico. En consecuencia, los CTBs en un primer mosaico se codifican en orden de exploración de trama, antes de pasar a los CTBs en el siguiente mosaico. De manera similar a los segmentos regulares, los mosaicos rompen las dependencias de predicción en imagen, así como las dependencias de decodificación por entropía. Sin embargo, los mosaicos no se pueden incluir en unidades

NAL individuales y, por lo tanto, los mosaicos no se pueden usar para la coincidencia de tamaño de MTU. Cada mosaico puede ser procesado por un procesador / núcleo, y la comunicación entre procesadores / entre núcleos empleada para la predicción en imagen entre unidades de procesamiento que decodifican mosaicos vecinos puede limitarse a transmitir un encabezado de segmento compartido (cuando los mosaicos adyacentes están en el mismo segmentos), y realizar el intercambio relacionado con el filtrado de bucle de muestras y metadatos reconstruidos. Cuando se incluye más de un mosaico en un segmento, el desplazamiento de byte de punto de entrada para cada mosaico que no sea el primer desplazamiento de punto de entrada en el segmento se puede señalar en el encabezado del segmento. Para cada segmento y mosaico, se debe cumplir al menos una de las siguientes condiciones: 1) todos los bloques de árboles codificados en un segmento pertenecen al mismo mosaico; y 2) todos los bloques de árboles codificados en un mosaico pertenecen al mismo segmento.

En WPP, la imagen se particiona en filas individuales de CTBs. Los mecanismos de decodificación y predicción por entropía pueden utilizar datos de CTBs en otras filas. El procesamiento en paralelo es posible mediante la decodificación en paralelo de filas de CTBs. Por ejemplo, una fila actual se puede decodificar en paralelo con una fila anterior. Sin embargo, la decodificación de la fila actual se retrasa del proceso de decodificación de las filas precedentes por dos CTBs. Este retraso asegura que los datos relacionados con el CTB superior/arriba y el CTB superior derecho/arriba y a la derecha del CTB actual en la fila actual estén disponibles antes de que se codifique el CTB actual. Este enfoque aparece como un frente de onda cuando se representa gráficamente. Este inicio escalonado permite la paralelización con hasta tantos procesadores / núcleos como la imagen contenga filas de CTB. Debido a que se permite la predicción en imagen entre filas vecinas de bloques de árbol dentro de una imagen, la comunicación entre procesadores / núcleos para permitir la predicción en imagen puede ser sustancial. La partición WPP considera los tamaños de unidad NAL. Por lo tanto, WPP no admite la coincidencia de tamaño de MTU. Sin embargo, los segmentos regulares se pueden usar junto con WPP, con cierta sobrecarga de codificación, para implementar la coincidencia de tamaño de MTU como se desee.

Los mosaicos también pueden incluir conjuntos de mosaicos restringidos de movimiento. Un conjunto de mosaicos restringidos de movimiento (MCTS) es un conjunto de mosaicos diseñado de tal manera que los vectores de movimiento asociados están restringidos para apuntar a ubicaciones de muestra completa dentro del MCTS y a ubicaciones de muestra fraccionaria que solo requieren ubicaciones de muestra completa dentro del MCTS para interpolación. Además, no se permite el uso de candidatos de vector de movimiento para la predicción de vectores de movimiento temporal derivados de bloques fuera del MCTS. De esta manera, cada MCTS puede decodificarse independientemente sin que existan mosaicos no incluidos en el MCTS. Los mensajes de información de mejora suplementaria (SEI) de MCTS temporales pueden utilizarse para indicar la existencia de MCTS en el flujo de bits y señalizar los MCTS. El mensaje SEI de MCTS proporciona información complementaria que se puede utilizar en la extracción del subflujo de bits de MCTS (especificada como parte de la semántica del mensaje SEI) para generar un flujo de bits conforme para un conjunto de MCTS. La información incluye varios conjuntos de información de extracción, cada uno de los cuales define un número de conjuntos de MCTS y contiene bytes de carga útil de secuencia de bytes sin procesar (RBSP) de los conjuntos de parámetros de vídeo de sustitución (VPS), conjuntos de parámetros de secuencia (SPS) y conjuntos de parámetros de imagen (PPS) para ser utilizado durante el proceso de extracción de subflujo de bits de MCTS. Cuando se extrae un subflujo de bits de acuerdo con el proceso de extracción del subflujo de bits de MCTS, los conjuntos de parámetros (VPS, SPS y PPS) pueden reescribirse o reemplazarse, y los encabezados de segmento pueden actualizarse porque uno o todos los elementos de sintaxis relacionados con la dirección de segmento (incluyendo first_slice_segment_in_pic_flag y slice_segment_address) pueden emplear diferentes valores en el subflujo de bits extraído.

Una imagen también puede particionarse en una o más subimágenes. Una subimagen es un conjunto rectangular de grupos/sectores de mosaicos que comienza con un grupo de mosaicos que tiene un tile_group_address igual a cero. Cada subimagen puede referirse a un PPS separado y, por lo tanto, puede tener una partición de mosaico separada. Las subimágenes pueden tratarse como imágenes en el proceso de decodificación. Las subimágenes de referencia para decodificar una subimagen actual se generan extrayendo el área colocada con la subimagen actual de las imágenes de referencia en el búfer/memoria intermedia de imágenes decodificadas. El área extraída se trata como una subimagen decodificada. La interpredicción puede tener lugar entre subimágenes del mismo tamaño y la misma ubicación dentro de la imagen. Un grupo de mosaicos, también conocido como segmento, es una secuencia de mosaicos relacionados en una imagen o una subimagen. Se pueden derivar varios elementos para determinar la ubicación de subimagen en una imagen. Por ejemplo, cada subimagen actual puede colocarse en la siguiente ubicación desocupada en el orden de exploración de trama de una CTU dentro de una imagen que sea lo suficientemente grande para contener la subimagen actual dentro de los límites de la imagen.

Además, la partición de imágenes puede basarse en mosaicos de nivel de imagen y mosaicos de nivel de secuencia. Los mosaicos de nivel de secuencia pueden incluir la funcionalidad de MCTS y pueden implementarse como subimágenes. Por ejemplo, un mosaico de nivel de imagen puede definirse como una región rectangular de bloques de árbol de codificación dentro de una columna de mosaico particular y una fila de mosaico particular en una imagen. Un mosaico de nivel de secuencia se puede definir como un conjunto de regiones rectangulares de bloques de árbol de codificación incluidos en diferentes marcos donde cada región rectangular comprende además uno o más mosaicos de nivel de imagen y el conjunto de regiones rectangulares de bloques de árbol de codificación se pueden decodificar independientemente de cualquier otro conjunto de regiones rectangulares similares. Un conjunto de grupos de mosaicos

de nivel de secuencia (STOPS) es un grupo de tales mosaicos de nivel de secuencia. El STOPS se puede señalizar en una unidad NAL de capa de codificación sin vídeo (VCL) con un identificador (ID) asociado en el encabezado de unidad NAL.

El esquema de particionamiento basado en la subimagen anterior puede estar asociado con ciertos problemas. Por ejemplo, cuando las subimágenes están habilitadas, el mosaico dentro de las subimágenes (partición de las subimágenes en mosaicos) se puede utilizar para soportar el procesamiento paralelo. La partición de mosaico de subimágenes para fines de procesamiento en paralelo puede cambiar de una imagen a otra (por ejemplo, para fines de equilibrio de carga de procesamiento en paralelo) y, por lo tanto, puede gestionarse a nivel de imagen (por ejemplo, en el PPS). Sin embargo, la partición de subimágenes (partición de imágenes en subimágenes) puede emplearse para soportar la región de interés (ROI) y el acceso de imágenes basado en subimágenes. En tal caso, la señalización de subimágenes o MCTS en el PPS no es eficaz.

En otro ejemplo, cuando cualquier subimagen de una imagen se codifica como una subimagen restringida de movimiento temporal, todas las subimágenes de la imagen se pueden codificar como subimágenes restringidas de movimiento temporal. Tal partición de imágenes puede ser limitante. Por ejemplo, codificar una subimagen como una subimagen restringida de movimiento temporal puede reducir la eficacia de la codificación a cambio de una funcionalidad adicional. Sin embargo, en la región de las aplicaciones basadas en intereses, normalmente sólo una o algunas de las subimágenes utilizan una funcionalidad basada en subimágenes restringidas de movimiento temporal. Por lo tanto, las subimágenes restantes sufren de una eficiencia de codificación reducida sin proporcionar ningún beneficio práctico.

En otro ejemplo, los elementos de sintaxis para especificar el tamaño de una subimagen se pueden especificar en unidades de tamaños de CTU de luma. En consecuencia, tanto el ancho como el alto de subimagen deben ser un múltiplo entero de CtbSizeY. Este mecanismo de especificar el ancho y la altura de subimagen puede dar lugar a varios problemas. Por ejemplo, la partición de subimágenes solo se aplica a imágenes con ancho de imagen y/o alto de imagen que son un múltiplo entero de CtbSizeY. Esto hace que la partición de subimágenes no esté disponible para imágenes que contienen dimensiones que no son múltiplos enteros de CTbSizeY. Si la partición de subimagen se aplicó al ancho y/o alto de imagen cuando la dimensión de imagen no es un múltiplo entero de CtbSizeY, la derivación del ancho de subimagen y/o la altura de subimagen en muestras de luma para la subimagen más a la derecha y la subimagen más al fondo sería incorrecta. Tal derivación incorrecta causaría resultados erróneos en algunas herramientas de codificación.

En otro ejemplo, puede no señalizarse la ubicación de una subimagen en una imagen. En cambio, la ubicación se deriva utilizando la siguiente regla. La subimagen actual se posiciona en la siguiente ubicación desocupada en el orden de exploración de trama de CTU dentro de una imagen que es lo suficientemente grande para contener la subimagen dentro de los límites de la imagen. La derivación de las ubicaciones de las subimágenes de esta forma puede provocar errores en algunos casos. Por ejemplo, si una subimagen se pierde durante la transmisión, las ubicaciones de otras subimágenes se derivan incorrectamente y las muestras decodificadas se colocan en ubicaciones erróneas. El mismo problema se aplica cuando las subimágenes llegan en el orden incorrecto.

En otro ejemplo, la decodificación de una subimagen puede requerir la extracción de subimágenes co-ubicadas en imágenes de referencia. Esto puede imponer una complejidad adicional y las cargas resultantes en términos de uso de recursos de memoria y procesador.

En otro ejemplo, cuando una subimagen se designa como una subimagen restringida de movimiento temporal, los filtros de lazo/bucle que atraviesan el límite de la subimagen se desactivan. Esto ocurre independientemente de si los filtros de bucle que atraviesan los límites de los mosaicos están habilitados. Tal restricción puede ser demasiado restrictiva y puede resultar en artefactos visuales para imágenes de video que emplean múltiples subimágenes.

En otro ejemplo, la relación entre los encabezados de grupo de mosaicos SPS, STGPS, PPS y es la siguiente. El STGPS se refiere al SPS, el PPS se refiere al STGPS y los encabezados de grupo de mosaicos/encabezados de segmento se refieren al PPS. Sin embargo, el STGPS y el PPS deben ser ortogonales en lugar del PPS que se refiere al STGPS. La disposición anterior también puede impedir que todos los grupos de mosaicos de la misma imagen se refieran al mismo PPS.

En otro ejemplo, cada STOPS puede contener IDs para cuatro lados de una subimagen. Estos IDs se utilizan para identificar subimágenes que comparten el mismo borde, de modo que se pueda definir su relación espacial relativa. Sin embargo, esta información puede no ser suficiente para derivar la información de posición y tamaño de un conjunto de grupos de mosaicos de nivel de secuencia en algunos casos. En otros casos, la señalización de la información de posición y tamaño puede ser redundante.

En otro ejemplo, un ID de STGPS se puede señalizar en un encabezado de unidad NAL de una unidad VCL NAL utilizando ocho bits. Esto puede ayudar con la extracción de subimágenes. Tal señalización puede aumentar innecesariamente la longitud del encabezado de la unidad NAL. Otro problema es que, a menos que los conjuntos de grupos de mosaicos a nivel de secuencia estén limitados para evitar superposiciones, un grupo de mosaicos puede asociarse con múltiples conjuntos de grupos de mosaicos a nivel de secuencia.

En el presente documento se describen varios mecanismos para abordar uno o más de los problemas mencionados anteriormente. En un primer ejemplo, la información de diseño de las subimágenes se incluye en un SPS en lugar de en un PPS. La información de diseño de la subimagen incluye la ubicación de subimagen y el tamaño de subimagen. La ubicación de subimagen es un desplazamiento entre la muestra superior izquierda de la subimagen y la muestra superior izquierda de la imagen. El tamaño de subimagen es la altura y el ancho de la subimagen medidos en muestras de luma. Como se señaló anteriormente, algunos sistemas incluyen información de mosaico en el PPS, ya que los mosaicos pueden cambiar de una imagen a otra. Sin embargo, se pueden utilizar subimágenes para soportar aplicaciones de ROI y acceso basado en subimágenes. Estas funciones no cambian en una base por imagen. Además, una secuencia de video puede incluir un solo SPS (o uno por segmento de video) y puede incluir hasta un PPS por imagen. La colocación de información de diseño para subimágenes en el SPS garantiza que el diseño solo se señalice una vez para una secuencia/segmento en lugar de señalizarlo de forma redundante para cada PPS. Por consiguiente, la disposición de la subimagen de señalización en el SPS aumenta la eficacia de la codificación y, por tanto, reduce el uso de recursos de red, recursos de memoria y/o recursos de procesamiento en el codificador y el decodificador. Además, algunos sistemas tienen la información de subimagen derivada por el decodificador. La señalización de la información de subimagen reduce la posibilidad de error en caso de pérdida de paquetes y admite una funcionalidad adicional en términos de extracción de subimágenes. En consecuencia, la disposición de subimagen de señalización en el SPS mejora la funcionalidad de un codificador y/o decodificador.

En un segundo ejemplo, los anchos de subimagen y las alturas de subimagen están restringidas a ser múltiplos del tamaño de la CTU. Sin embargo, estas restricciones se eliminan cuando se coloca una subimagen en el borde derecho de la imagen o en el borde inferior de la imagen, respectivamente. Como se indicó anteriormente, algunos sistemas de video pueden limitar las subimágenes para incluir alturas y anchos que son múltiplos del tamaño de la CTU. Esto evita que las subimágenes funcionen correctamente con muchos diseños de imagen. Al permitir que las subimágenes inferior y derecha incluyan alturas y anchos, respectivamente, que no sean múltiplos del tamaño de CTU, las subimágenes pueden usarse con cualquier imagen sin causar errores de decodificación. Esto da como resultado un aumento de la funcionalidad del codificador y decodificador. Además, la mayor funcionalidad permite que un codificador codifique imágenes de manera más eficiente, lo que reduce el uso de recursos de red, recursos de memoria y/o recursos de procesamiento en el codificador y el decodificador.

En un tercer ejemplo, las subimágenes están restringidas para cubrir una imagen sin espacios/huecos ni superposiciones. Como se señaló anteriormente, algunos sistemas de codificación de video permiten que las subimágenes incluyan espacios y superposiciones. Esto crea la posibilidad de que los grupos/segmentos de mosaicos se asocien con múltiples subimágenes. Si esto está permitido en el codificador, los decodificadores deben construirse para soportar tal esquema de codificación incluso cuando el esquema de decodificación se usa raramente. Al no permitir los espacios y superposiciones de las subimágenes, la complejidad del decodificador puede reducirse ya que no se requiere que el decodificador tenga en cuenta los posibles espacios y superposiciones al determinar los tamaños y ubicaciones de las subimágenes. Además, no permitir los espacios y superposiciones de subimágenes reduce la complejidad de los procesos de optimización de distorsión de tasa (RDO) en el codificador, ya que el codificador puede omitir considerar los casos de espacios y superposiciones al seleccionar una codificación para una secuencia de vídeo. Por consiguiente, evitar espacios y superposiciones puede reducir el uso de recursos de memoria y/o recursos de procesamiento en el codificador y el decodificador.

En un cuarto ejemplo, se puede señalizar una bandera en el SPS para indicar cuándo una subimagen es una subimagen restringida de movimiento temporal. Como se indicó anteriormente, algunos sistemas pueden configurar colectivamente todas las subimágenes para que sean subimágenes con restricción de movimiento temporal o no permitan por completo el uso de subimágenes con restricción de movimiento temporal. Dichas subimágenes restringidas por movimiento temporal proporcionan una funcionalidad de extracción independiente a costa de una menor eficiencia de codificación. Sin embargo, en la región de aplicaciones basadas en intereses, la región de interés debe codificarse para una extracción independiente, mientras que las regiones fuera de la región de interés no necesitan dicha funcionalidad. Por lo tanto, las subimágenes restantes sufren de una eficiencia de codificación reducida sin proporcionar ningún beneficio práctico. En consecuencia, la bandera permite una mezcla de subimágenes restringidas por movimiento temporal que proporcionan funcionalidad de extracción independiente y subimágenes no restringidas por movimiento para una mayor eficiencia de codificación cuando no se desea una extracción independiente. Por tanto, la bandera permite una mayor funcionalidad y/o una mayor eficiencia de codificación, lo que reduce el uso de recursos de red, recursos de memoria y/o recursos de procesamiento en el codificador y el decodificador.

En un quinto ejemplo, se señaliza un conjunto completo de IDs de subimagen en el SPS, y los encabezados de segmento incluyen un ID de subimagen que indica la subimagen que contiene los cortes correspondientes. Como se señaló anteriormente, algunos sistemas señalan las posiciones de las subimágenes en relación con otras subimágenes. Esto causa un problema si las subimágenes se pierden o se extraen por separado. Al designar cada subimagen mediante un ID, las subimágenes se pueden posicionar y dimensionar sin referencia a otras subimágenes. Esto, a su vez, admite la corrección de errores, así como aplicaciones que solo extraen algunas de las subimágenes y evitan la transmisión de otras subimágenes. Se puede enviar una lista completa de todas las identificaciones de subimagen en el SPS junto con la información de tamaño relevante. Cada encabezado de segmento puede contener un ID de subimagen que indica la subimagen que incluye el segmento correspondiente. De esta manera, las subimágenes y los segmentos correspondientes se pueden extraer y posicionar sin referencia a otras subimágenes. Por tanto, los IDs de subimagen

soportan una mayor funcionalidad y/o una mayor eficiencia de codificación, lo que reduce el uso de recursos de red, recursos de memoria y/o recursos de procesamiento en el codificador y el decodificador.

En un sexto ejemplo, los niveles se señalizan para cada subimagen. En algunos sistemas de codificación de vídeo, los niveles se señalizan mediante imágenes. Un nivel indica los recursos de hardware necesarios para decodificar la imagen. Como se señaló anteriormente, diferentes subimágenes pueden tener una funcionalidad diferente en algunos casos y, por lo tanto, pueden tratarse de manera diferente durante el proceso de codificación. Como tal, un nivel basado en imágenes puede no ser útil para decodificar algunas subimágenes. Por tanto, la presente divulgación incluye niveles para cada subimagen. De esta manera, cada subimagen puede codificarse independientemente de otras subimágenes sin sobrecargar innecesariamente al decodificador estableciendo requisitos de decodificación demasiado altos para las subimágenes codificadas de acuerdo con mecanismos menos complejos. La información de nivel de subimagen señalizada soporta una mayor funcionalidad y/o una mayor eficiencia de codificación, lo que reduce el uso de recursos de red, recursos de memoria y/o recursos de procesamiento en el codificador y el decodificador.

La FIG. 1 es un diagrama de flujo de un método operativo de ejemplo 100 para codificar una señal de vídeo. Específicamente, una señal de video se codifica en un codificador. El proceso de codificación comprime la señal de video empleando varios mecanismos para reducir el tamaño del archivo de video. Un tamaño de archivo más pequeño permite que el archivo de video comprimido se transmita hacia un usuario, al tiempo que reduce la sobrecarga de ancho de banda asociada. El decodificador luego decodifica el archivo de video comprimido para reconstruir la señal de video original para mostrarla a un usuario final. El proceso de decodificación generalmente refleja el proceso de codificación para permitir que el decodificador reconstruya consistentemente la señal de video.

En el paso 101, la señal de vídeo se introduce en el codificador. Por ejemplo, la señal de video puede ser un archivo de video sin comprimir almacenado en la memoria. Como otro ejemplo, el archivo de video puede ser capturado por un dispositivo de captura de video, como una cámara de video, y codificado para soportar la transmisión en vivo del video. El archivo de video puede incluir tanto un componente de audio como un componente de video. El componente de video contiene una serie de fotogramas de imagen que, cuando se ven en una secuencia, dan la impresión visual de movimiento. Los fotogramas contienen píxeles que se expresan en términos de luz, denominados en este documento componentes de luma (o muestras de luma), y color, que se denomina componentes de croma (o muestras de color). En algunos ejemplos, los fotogramas también pueden contener valores de profundidad para admitir la visualización tridimensional.

En el paso 103, el video se particiona en bloques. El particionamiento incluye subdividir los píxeles de cada fotograma en bloques cuadrados y / o rectangulares para la compresión. Por ejemplo, en codificación de video de alta eficiencia (HEVC) (también conocida como H.265 y MPEG-H Parte 2) el fotograma se puede dividir primero en unidades de árbol de codificación (CTU), que son bloques de un tamaño predefinido (por ejemplo, sesenta y cuatro píxeles por sesenta y cuatro píxeles). Las CTUs contienen muestras de luma y croma. Pueden emplearse árboles de codificación para dividir las CTUs en bloques y luego subdividir recursivamente los bloques hasta que se logren configuraciones que admitan una codificación adicional. Por ejemplo, los componentes de luma de un fotograma pueden subdividirse hasta que los bloques individuales contengan valores de iluminación relativamente homogéneos. Además, los componentes de croma de un fotograma pueden subdividirse hasta que los bloques individuales contengan valores de color relativamente homogéneos. En consecuencia, los mecanismos de partición varían según el contenido de los fotogramas de vídeo.

En el paso 105, se emplean varios mecanismos de compresión para comprimir los bloques de imagen particionados en el paso 103. Por ejemplo, se puede emplear la interpredicción y / o intrapredicción. La interpredicción está diseñada para aprovechar el hecho de que los objetos en una escena común tienden a aparecer en fotograma sucesivos. Por consiguiente, un bloque que representa un objeto en un fotograma de referencia no necesita describirse repetidamente en marcos adyacentes. Específicamente, un objeto, como una mesa, puede permanecer en una posición constante en varios fotogramas. Por lo tanto, la mesa se describe una vez y los fotogramas adyacentes pueden hacer referencia al fotograma de referencia. Pueden emplearse mecanismos de coincidencia de patrones para hacer coincidir objetos en múltiples fotograma. Además, los objetos en movimiento se pueden representar en varios fotogramas, por ejemplo, debido al movimiento del objeto o al movimiento de la cámara. Como ejemplo particular, un video puede mostrar un automóvil que se mueve por la pantalla en múltiples cuadros. Se pueden emplear vectores de movimiento para describir dicho movimiento. Un vector de movimiento es un vector bidimensional que proporciona un desplazamiento de las coordenadas de un objeto en un marco a las coordenadas del objeto en un marco de referencia. Como tal, la interpredicción puede codificar un bloque de imagen en un fotograma actual como un conjunto de vectores de movimiento que indican un desplazamiento de un bloque correspondiente en un fotograma de referencia.

La intrapredicción codifica bloques en un fotograma común. La intrapredicción aprovecha el hecho de que los componentes de luma y croma tienden a agruparse en un fotograma. Por ejemplo, un parche de verde en una porción de un árbol tiende a colocarse adyacente a parches de verde similares. La intrapredicción emplea múltiples modos de predicción direccional (por ejemplo, treinta y tres en HEVC), un modo plano y un modo actual directo (DC). Los modos direccionales indican que un bloque actual es similar / igual que las muestras de un bloque vecino en la dirección correspondiente. El modo plano indica que una serie de bloques a lo largo de una fila / columna (por ejemplo, un plano) se puede interpolar en función de los bloques vecinos en los bordes de la fila. El modo plano, en efecto, indica una

transición suave de luz / color a través de una fila / columna al emplear una pendiente relativamente constante en los valores cambiantes. El modo DC se emplea para suavizar los límites e indica que un bloque es similar / igual que un valor promedio asociado con muestras de todos los bloques vecinos asociados con las direcciones angulares de los modos de predicción direccional. Por consiguiente, los bloques de intrapredicción pueden representar bloques de imagen como varios valores de modo de predicción relacional en lugar de los valores reales. Además, los bloques de interpredicción pueden representar bloques de imagen como valores de vector de movimiento en lugar de valores reales. En cualquier caso, es posible que los bloques de predicción no representen exactamente los bloques de imágenes en algunos casos. Cualesquiera diferencias se almacenan en bloques residuales. Se pueden aplicar transformaciones a los bloques residuales para comprimir aún más el archivo.

En el paso 107, se pueden aplicar varias técnicas de filtrado. En HEVC, los filtros se aplican de acuerdo con un esquema de filtrado en bucle. La predicción basada en bloques discutida anteriormente puede resultar en la creación de imágenes en bloques en el decodificador. Además, el esquema de predicción basado en bloques puede codificar un bloque y luego reconstruir el bloque codificado para su uso posterior como bloque de referencia. El esquema de filtrado en bucle aplica de forma iterativa filtros de supresión de ruido, filtros de desbloqueo, filtros de bucle adaptativo y filtros de compensación adaptativa de muestra (SAO) a los bloques / fotogramas. Estos filtros mitigan tales artefactos de bloqueo para que el archivo codificado se pueda reconstruir con precisión. Además, estos filtros mitigan los artefactos en los bloques de referencia reconstruidos, de modo que es menos probable que los artefactos creen artefactos adicionales en los bloques posteriores que se codifican en función de los bloques de referencia reconstruidos.

Una vez que la señal de video ha sido particionada, comprimida y filtrada, los datos resultantes se codifican en un flujo de bits en el paso 109. El flujo de bits incluye los datos discutidos anteriormente, así como cualquier dato de señalización deseado para soportar la reconstrucción adecuada de la señal de video en el decodificador. Por ejemplo, tales datos pueden incluir datos de partición, datos de predicción, bloques residuales y varios indicadores que proporcionan instrucciones de codificación al decodificador. El flujo de bits puede almacenarse en la memoria para su transmisión hacia un decodificador cuando se solicite. El flujo de bits también se puede difundir y / o multidifundir hacia una pluralidad de decodificadores. La creación del flujo de bits es un proceso iterativo. Por consiguiente, los pasos 101, 103, 105, 107 y 109 pueden ocurrir de forma continua y / o simultánea en muchos fotogramas y bloques. El orden mostrado en la FIG. 1 se presenta para mayor claridad y facilidad de discusión, y no pretende limitar el proceso de codificación de video a un orden en particular.

El decodificador recibe el flujo de bits y comienza el proceso de decodificación en el paso 111. Específicamente, el decodificador emplea un esquema de decodificación por entropía para convertir el flujo de bits en la sintaxis y los datos de video correspondientes. El decodificador emplea los datos de sintaxis del flujo de bits para determinar las particiones para los fotogramas en el paso 111. La partición debe coincidir con los resultados de la partición de bloque en el paso 103. Ahora se describe la codificación / decodificación por entropía empleada en el paso 111. El codificador hace muchas elecciones durante el proceso de compresión, como seleccionar esquemas de partición de bloque entre varias opciones posibles basadas en el posicionamiento espacial de los valores en las imágenes de entrada. La señalización de las opciones exactas puede emplear una gran cantidad de contenedores. Como se usa en este documento, un contenedor es un valor binario que se trata como una variable (por ejemplo, un valor de bit que puede variar según el contexto). La codificación por entropía permite al codificador descartar cualquier opción que claramente no sea viable para un caso particular, dejando un conjunto de opciones permitidas. A cada opción permitida se le asigna una palabra de código. La longitud de las palabras de código se basa en el número de opciones permitidas (por ejemplo, un contenedor para dos opciones, dos contenedores para tres o cuatro opciones, etc.). El codificador luego codifica la palabra de código para la opción seleccionada. Este esquema reduce el tamaño de las palabras de código ya que las palabras de código son tan grandes como se desee para indicar de forma única una selección de un pequeño subconjunto de opciones permitidas en lugar de indicar de forma única la selección de un conjunto potencialmente grande de todas las opciones posibles. El decodificador luego decodifica la selección determinando el conjunto de opciones permitidas de una manera similar al codificador. Al determinar el conjunto de opciones permitidas, el decodificador puede leer la palabra de código y determinar la selección realizada por el codificador.

En el paso 113, el decodificador realiza la decodificación de bloques. Específicamente, el decodificador emplea transformaciones inversas para generar bloques residuales. Luego, el decodificador emplea los bloques residuales y los bloques de predicción correspondientes para reconstruir los bloques de imagen de acuerdo con la partición. Los bloques de predicción pueden incluir tanto bloques de intrapredicción como bloques de interpredicción como se generan en el codificador en el paso 105. A continuación, los bloques de imágenes reconstruidos se colocan en fotogramas de una señal de vídeo reconstruida de acuerdo con los datos de partición determinados en el paso 111. La sintaxis para el paso 113 también se puede señalar en el flujo de bits mediante la codificación por entropía como se discutió anteriormente.

En el paso 115, se realiza el filtrado en los fotogramas de la señal de video reconstruida de una manera similar al paso 107 en el codificador. Por ejemplo, se pueden aplicar filtros de supresión de ruido, filtros de desbloqueo, filtros de bucle adaptativo y filtros SAO a los fotogramas para eliminar los artefactos de bloque. Una vez que se filtran los fotogramas, la señal de vídeo se puede enviar a una pantalla en el paso 117 para visualización por un usuario final.

La FIG. 2 es un diagrama esquemático de un ejemplo de sistema de codificación y decodificación (códec) 200 para codificación de vídeo. Específicamente, el sistema de códec 200 proporciona funcionalidad para respaldar la

implementación del método operativo 100. El sistema de códec 200 está generalizado para representar los componentes empleados tanto en un codificador como en un decodificador. El sistema de códec 200 recibe y particiona una señal de video como se discutió con respecto a los pasos 101 y 103 en el método operativo 100, lo que da como resultado una señal de video 201 particionada. A continuación, el sistema de códec 200 comprime la señal de vídeo particionada 201 en un flujo de bits codificado cuando actúa como un codificador como se describe con respecto a los pasos 105, 107 y 109 en el método 100. Cuando actúa como un decodificador, el sistema de códec 200 genera una señal de vídeo de salida a partir del flujo de bits como se describe con respecto a los pasos 111, 113, 115 y 117 en el método operativo 100. El sistema de códec 200 incluye un componente de control de codificador general 211, un componente de cuantificación y de escala de transformación 213, un componente de estimación intraimagen 215, un componente de predicción intraimagen 217, un componente de compensación de movimiento 219, un componente de estimación de movimiento 221, un componente de transformación inversa y de escala 229, un componente de análisis de control de filtro 227, un componente de filtro en bucle 225, un componente de búfer/memoria intermedia de imágenes decodificadas 223 y un componente de codificación aritmética binaria adaptativa de contexto (CABAC) y de formato de encabezado 231. Dichos componentes se acoplan como se muestra. En la FIG. 2, las líneas negras indican el movimiento de los datos a codificar / decodificar, mientras que las líneas discontinuas indican el movimiento de los datos de control que controlan el funcionamiento de otros componentes. Todos los componentes del sistema de códec 200 pueden estar presentes en el codificador. El decodificador puede incluir un subconjunto de los componentes del sistema de códec 200. Por ejemplo, el decodificador puede incluir el componente de predicción intraimagen 217, el componente de compensación de movimiento 219, el componente de transformación inversa y de escala 229, el componente de filtro en bucle 225 y el componente de búfer de imágenes decodificadas 223. Estos componentes se describen ahora.

La señal de video particionada 201 es una secuencia de video capturada que ha sido particionada en bloques de píxeles por un árbol de codificación. Un árbol de codificación emplea varios modos de división para subdividir un bloque de píxeles en bloques de píxeles más pequeños. Estos bloques pueden luego subdividirse en bloques más pequeños. Los bloques pueden denominarse nodos en el árbol de codificación. Los nodos principales más grandes se dividen en nodos secundarios más pequeños. El número de veces que se subdivide un nodo se denomina profundidad del árbol de codificación / nodo. Los bloques divididos se pueden incluir en unidades de codificación (CUs) en algunos casos. Por ejemplo, una CU puede ser una subporción de una CTU que contiene un bloque de luma, un bloque(s) de croma (Cr) de diferencia de rojo y bloque(s) de croma de diferencia de azul (Cb) junto con las instrucciones de sintaxis correspondientes para la CU. Los modos de división pueden incluir un árbol binario (BT), árbol triple (TT) y un árbol cuádruple (QT) empleados para dividir un nodo en dos, tres o cuatro nodos secundarios, respectivamente, de diferentes formas dependiendo de los modos de división empleados. La señal de vídeo particionada 201 se reenvía al componente de control de codificador general 211, al componente de cuantificación y de escala de transformación 213, al componente de estimación intraimagen 215, al componente de análisis de control de filtro 227 y al componente de estimación de movimiento 221 para su compresión.

El componente de control de codificador general 211 está configurado para tomar decisiones relacionadas con la codificación de las imágenes de la secuencia de vídeo en el flujo de bits de acuerdo con las limitaciones de la aplicación. Por ejemplo, el componente de control de codificador general 211 gestiona la optimización del tamaño de tasa de bits / flujo de bits frente a la calidad de reconstrucción. Dichas decisiones se pueden tomar en función de la disponibilidad de espacio de almacenamiento / ancho de banda y las solicitudes de resolución de imagen. El componente de control de codificador general 211 también gestiona la utilización del búfer a la luz de la velocidad de transmisión para mitigar los problemas de saturación y falta de ejecución del búfer. Para gestionar estos problemas, el componente de control de codificador 211 general gestiona la partición, la predicción y el filtrado de los otros componentes. Por ejemplo, el componente de control de codificador general 211 puede aumentar dinámicamente la complejidad de la compresión para aumentar la resolución y aumentar el uso del ancho de banda o disminuir la complejidad de la compresión para disminuir la resolución y el uso del ancho de banda. Por tanto, el componente de control general de codificador 211 controla los otros componentes del sistema de códec 200 para equilibrar la calidad de reconstrucción de la señal de vídeo con las preocupaciones sobre la tasa de bits. El componente de control de codificador general 211 crea datos de control, que controlan el funcionamiento de los otros componentes. Los datos de control también se envían al componente de CABAC y de formato de encabezado 231 para ser codificados en el flujo de bits para señalizar parámetros para decodificar en el decodificador.

La señal de vídeo particionada 201 también se envía al componente de estimación de movimiento 221 y al componente de compensación de movimiento 219 para la interpredicción. Un fotograma o segmento de la señal de video particionada 201 puede dividirse en múltiples bloques de video. El componente de estimación de movimiento 221 y el componente de compensación de movimiento 219 realizan una codificación interpredictiva del bloque de vídeo recibido en relación con uno o más bloques en uno o más fotogramas de referencia para proporcionar predicción temporal. El sistema de códec 200 puede realizar múltiples pasadas de codificación, por ejemplo, para seleccionar un modo de codificación apropiado para cada bloque de datos de video.

El componente de estimación de movimiento 221 y el componente de compensación de movimiento 219 pueden estar muy integrados, pero se ilustran por separado con fines conceptuales. La estimación de movimiento, realizada por el componente de estimación de movimiento 221, es el proceso de generar vectores de movimiento, que estiman el movimiento para bloques de vídeo. Un vector de movimiento, por ejemplo, puede indicar el desplazamiento de un objeto codificado con respecto a un bloque predictivo. Un bloque predictivo es un bloque que se encuentra que coincide

estrechamente con el bloque a codificar, en términos de diferencia de píxeles. Un bloque predictivo también puede denominarse bloque de referencia. Dicha diferencia de píxeles puede determinarse mediante la suma de la diferencia absoluta (SAD), la suma de la diferencia cuadrada (SSD) u otras métricas de diferencia. HEVC emplea varios objetos codificados que incluyen una CTU, bloques de árbol de codificación (CTBs) y CU. Por ejemplo, una CTU se puede dividir en CTBs, que luego se pueden dividir en CBs para su inclusión en las CUs. Una CU puede codificarse como una unidad de predicción (PU) que contiene datos de predicción y / o una unidad de transformación (TU) que contiene datos residuales transformados para la CU. El componente de estimación de movimiento 221 genera vectores de movimiento, PUs y TUs utilizando un análisis de distorsión de tasa como parte de un proceso de optimización de distorsión de tasa. Por ejemplo, el componente de estimación de movimiento 221 puede determinar múltiples bloques de referencia, múltiples vectores de movimiento, etc. para un bloque / fotograma actual, y puede seleccionar los bloques de referencia, vectores de movimiento, etc. que tienen las mejores características de distorsión de tasa. Las mejores características de distorsión de tasa equilibran tanto la calidad de la reconstrucción de video (por ejemplo, la cantidad de datos perdidos por compresión) con la eficiencia de codificación (por ejemplo, el tamaño de la codificación final).

En algunos ejemplos, el sistema de códec 200 puede calcular valores para posiciones de píxeles sub-enteros de imágenes de referencia almacenadas en el componente de búfer de imágenes decodificadas 223. Por ejemplo, el sistema de códec de vídeo 200 puede interpolar valores de posiciones de un cuarto de píxel, posiciones de un octavo de píxel u otras posiciones de píxeles fraccionarios de la imagen de referencia. Por lo tanto, el componente de estimación de movimiento 221 puede realizar una búsqueda de movimiento en relación con las posiciones de píxeles completos y las posiciones de píxeles fraccionarios y generar un vector de movimiento con precisión de píxeles fraccionarios. El componente de estimación de movimiento 221 calcula un vector de movimiento para una PU de un bloque de vídeo en un segmento intercodificado comparando la posición de la PU con la posición de un bloque predictivo de una imagen de referencia. El componente de estimación de movimiento 221 envía el vector de movimiento calculado como datos de movimiento al componente de CABAC y de formato de encabezado 231 para codificación y movimiento al componente de compensación de movimiento 219.

La compensación de movimiento, realizada por el componente de compensación de movimiento 219, puede implicar buscar o generar el bloque predictivo en base al vector de movimiento determinado por el componente de estimación de movimiento 221. De nuevo, el componente de estimación de movimiento 221 y el componente de compensación de movimiento 219 pueden integrarse funcionalmente, en algunos ejemplos. Al recibir el vector de movimiento para la PU del bloque de vídeo actual, el componente de compensación de movimiento 219 puede localizar el bloque predictivo al que apunta el vector de movimiento. A continuación, se forma un bloque de vídeo residual restando los valores de píxeles del bloque predictivo de los valores de píxeles del bloque de vídeo actual que se está codificando, formando valores de diferencia de píxeles. En general, el componente de estimación de movimiento 221 realiza una estimación de movimiento con relación a los componentes de luma, y el componente de compensación de movimiento 219 usa vectores de movimiento calculados en base a los componentes de luma tanto para los componentes de croma como para los componentes de luma. El bloque predictivo y el bloque residual se envían al componente de cuantificación y de escala de transformación 213.

La señal de vídeo particionada 201 también se envía al componente de estimación intraimagen 215 y al componente de predicción intraimagen 217. Al igual que con el componente de estimación de movimiento 221 y el componente de compensación de movimiento 219, el componente de estimación intraimagen 215 y el componente de predicción intraimagen 217 pueden estar muy integrados, pero se ilustran por separado con fines conceptuales. El componente de estimación intraimagen 215 y el componente de predicción intraimagen 217 intrapredicen un bloque actual en relación con los bloques en un fotograma actual, como alternativa a la interpredicción realizada por el componente de estimación de movimiento 221 y el componente de compensación de movimiento 219 entre fotogramas, como se describió anteriormente. En particular, el componente de estimación intraimagen 215 determina un modo de intrapredicción a usar para codificar un bloque actual. En algunos ejemplos, el componente de estimación intraimagen 215 selecciona un modo de intrapredicción apropiado para codificar un bloque actual a partir de múltiples modos de intrapredicción probados. Los modos de intrapredicción seleccionados se reenvían luego al componente de CABAC y de formato de encabezado 231 para su codificación.

Por ejemplo, el componente de estimación intraimagen 215 calcula los valores de distorsión de tasa usando un análisis de distorsión de tasa para los diversos modos de intrapredicción probados y selecciona el modo de intrapredicción que tiene las mejores características de distorsión de tasa entre los modos probados. El análisis de distorsión de tasa generalmente determina una cantidad de distorsión (o error) entre un bloque codificado y un bloque no codificado original que fue codificado para producir el bloque codificado, así como una tasa de bits (por ejemplo, una cantidad de bits) utilizada para producir el bloque codificado. El componente de estimación de intraimagen 215 calcula las relaciones a partir de las distorsiones y tasas para los diversos bloques codificados para determinar qué modo de intrapredicción presenta el mejor valor de distorsión de tasa para el bloque. Además, el componente de estimación intraimagen 215 puede configurarse para codificar bloques de profundidad de un mapa de profundidad utilizando un modo de modelado de profundidad (DMM) basado en la optimización de la distorsión de tasa (RDO).

El componente de predicción intraimagen 217 puede generar un bloque residual a partir del bloque predictivo basado en los modos de intrapredicción seleccionados determinados por el componente de estimación intraimagen 215 cuando se implementa en un codificador o leer el bloque residual del flujo de bits cuando se implementa en un

decodificador. El bloque residual incluye la diferencia de valores entre el bloque predictivo y el bloque original, representado como una matriz. A continuación, el bloque residual se envía al componente de cuantificación y de escala de transformación 213. El componente de estimación intraimagen 215 y el componente de predicción intraimagen 217 pueden operar tanto en componentes de luma como de croma.

El componente de cuantificación y de escala de transformación 213 está configurado para comprimir más el bloque residual. El componente de cuantificación y de escala de transformación 213 aplica una transformada, tal como una transformada de coseno discreta (DCT), una transformada de seno discreta (DST) o una transformada conceptualmente similar, al bloque residual, produciendo un bloque de video que comprende valores de coeficiente de transformación residuales. También se podrían utilizar transformadas wavelet, transformadas de enteros, transformaciones de subbandas u otros tipos de transformaciones. La transformación puede convertir la información residual de un dominio de valor de píxel en un dominio de transformación, tal como un dominio de frecuencia. El componente de cuantificación y de escala de transformación 213 también está configurado para escalar la información residual transformada, por ejemplo, basándose en la frecuencia. Tal escala implica aplicar un factor de escala a la información residual para que la información de frecuencia diferente se cuantifique en diferentes granularidades, lo que puede afectar la calidad visual final del video reconstruido. El componente de cuantificación y de escala de transformación 213 también está configurado para cuantificar los coeficientes de transformación para reducir aún más la tasa de bits. El proceso de cuantificación puede reducir la profundidad de bits asociada con algunos o todos los coeficientes. El grado de cuantificación se puede modificar ajustando un parámetro de cuantificación. En algunos ejemplos, el componente de cuantificación y de escala de transformación 213 puede entonces realizar una exploración de la matriz que incluye los coeficientes de transformación cuantificados. Los coeficientes de transformación cuantificados se envían al componente de CABAC y de formato de encabezado 231 para ser codificados en el flujo de bits.

El componente de transformación inversa y de escala 229 aplica una operación inversa del componente de cuantificación y de escala de transformación 213 para soportar la estimación del movimiento. El componente de transformación inversa y de escala 229 aplica escalamiento inverso, transformación y / o cuantificación para reconstruir el bloque residual en el dominio de píxeles, por ejemplo, para uso posterior como un bloque de referencia que puede convertirse en un bloque predictivo para otro bloque actual. El componente de estimación de movimiento 221 y / o el componente de compensación de movimiento 219 pueden calcular un bloque de referencia añadiendo el bloque residual de nuevo a un bloque predictivo correspondiente para su uso en la estimación de movimiento de un bloque / cuadro posterior. Filtros se aplican a los bloques de referencia reconstruidos para mitigar los artefactos creados durante el escalado, la cuantificación y la transformación. De lo contrario, tales artefactos podrían causar una predicción inexacta (y crear artefactos adicionales) cuando se predicen bloques posteriores.

El componente de análisis de control de filtro 227 y el componente de filtro en bucle 225 aplican los filtros a los bloques residuales y / o a los bloques de imagen reconstruidos. Por ejemplo, el bloque residual transformado del componente de transformación inversa y de escala 229 puede combinarse con un bloque de predicción correspondiente del componente de predicción intraimagen 217 y / o el componente de compensación de movimiento 219 para reconstruir el bloque de imagen original. A continuación, los filtros se pueden aplicar al bloque de imagen reconstruido. En algunos ejemplos, los filtros se pueden aplicar en cambio a los bloques residuales. Como ocurre con otros componentes de la FIG. 2, el componente de análisis de control de filtro 227 y el componente de filtro en bucle 225 están altamente integrados y pueden implementarse juntos, pero se representan por separado con fines conceptuales. Los filtros aplicados a los bloques de referencia reconstruidos se aplican a regiones espaciales particulares e incluyen múltiples parámetros para ajustar cómo se aplican dichos filtros. El componente de análisis de control de filtro 227 analiza los bloques de referencia reconstruidos para determinar dónde deberían aplicarse dichos filtros y establece los parámetros correspondientes. Dichos datos se envían al componente de CABAC y de formato de encabezado 231 como datos de control de filtro para la codificación. El componente de filtro en bucle 225 aplica dichos filtros basándose en los datos de control del filtro. Los filtros pueden incluir un filtro de desbloqueo, un filtro de supresión de ruido, un filtro SAO y un filtro de bucle adaptativo. Dichos filtros se pueden aplicar en el dominio espacial / de píxeles (por ejemplo, en un bloque de píxeles reconstruido) o en el dominio de frecuencia, según el ejemplo.

Cuando funciona como un codificador, el bloque de imagen reconstruido filtrado, el bloque residual y / o el bloque de predicción se almacenan en el componente de búfer de imágenes decodificadas 223 para su uso posterior en la estimación de movimiento como se discutió anteriormente. Cuando funciona como decodificador, el componente de búfer de imágenes decodificadas 223 almacena y reenvía los bloques reconstruidos y filtrados hacia una pantalla como parte de una señal de vídeo de salida. El componente de búfer de imágenes decodificadas 223 puede ser cualquier dispositivo de memoria capaz de almacenar bloques de predicción, bloques residuales y / o bloques de imágenes reconstruidas.

El componente de CABAC y de formato de encabezado 231 recibe los datos de los diversos componentes del sistema de códec 200 y codifica dichos datos en un flujo de bits codificado para su transmisión hacia un decodificador. Específicamente, el componente de CABAC y de formato de encabezado 231 genera varios encabezados para codificar datos de control, tales como datos de control general y datos de control de filtro. Además, los datos de predicción, incluidos los datos de intrapredicción y de movimiento, así como los datos residuales en forma de datos de coeficientes de transformación cuantificados, están todos codificados en el flujo de bits. El flujo de bits final incluye toda la información deseada por el decodificador para reconstruir la señal de video particionada original 201. Dicha información también

puede incluir tablas de índice de modo de intrapredicción (también denominadas tablas de mapeo de palabras de código), definiciones de contextos de codificación para varios bloques, indicaciones de los modos de intrapredicción más probables, una indicación de información de partición, etc. Tales datos pueden ser codificados empleando codificación por entropía. Por ejemplo, la información puede codificarse empleando codificación de longitud variable adaptativa al contexto (CAVLC), CABAC, codificación aritmética binaria adaptativa al contexto basada en sintaxis (SBAC), codificación por entropía de partición de intervalo de probabilidad (PIPE) u otra técnica de codificación por entropía. Después de la codificación por entropía, el flujo de bits codificado puede transmitirse a otro dispositivo (por ejemplo, un decodificador de vídeo) o archivarse para su posterior transmisión o recuperación.

La FIG. 3 es un diagrama de bloques que ilustra un codificador de vídeo de ejemplo 300. Puede emplearse el codificador de vídeo 300 para implementar las funciones de codificación del sistema de códec 200 y / o implementar los pasos 101, 103, 105, 107 y / o 109 del método operativo 100. El codificador 300 particiona una señal de video de entrada, dando como resultado una señal de video particionada 301, que es sustancialmente similar a la señal de video particionada 201. La señal de video particionada 301 es luego comprimida y codificada en un flujo de bits por componentes del codificador 300.

Específicamente, la señal de vídeo dividida 301 se reenvía a un componente de predicción intraimagen 317 para la intrapredicción. El componente de predicción intraimagen 317 puede ser sustancialmente similar al componente de estimación intraimagen 215 y al componente de predicción intraimagen 217. La señal de vídeo particionada 301 también se envía a un componente de compensación de movimiento 321 para la interpredicción basada en bloques de referencia en un componente de búfer de imágenes decodificadas 323. El componente de compensación de movimiento 321 puede ser sustancialmente similar al componente de estimación de movimiento 221 y al componente de compensación de movimiento 219. Los bloques de predicción y los bloques residuales del componente de predicción intraimagen 317 y el componente de compensación de movimiento 321 se envían a un componente de transformación y cuantificación 313 para la transformación y cuantificación de los bloques residuales. El componente de transformación y cuantificación 313 puede ser sustancialmente similar al componente de transformación y cuantificación 213. Los bloques residuales transformados y cuantificados y los bloques de predicción correspondientes (junto con los datos de control asociados) se envían a un componente de codificación por entropía 331 para codificar en un flujo de bits. El componente de codificación por entropía 331 puede ser sustancialmente similar al componente de CABAC y de formato de encabezado 231.

Los bloques residuales transformados y cuantificados y / o los correspondientes bloques de predicción también se envían desde el componente de transformación y cuantificación 313 a un componente de transformación inversa y cuantificación 329 para la reconstrucción en bloques de referencia para su uso por el componente de compensación de movimiento 321. El componente de transformación y cuantificación inversa 329 puede ser sustancialmente similar al componente de transformación inversa y de escala 229. Los filtros en bucle en un componente de filtro en bucle 325 también se aplican a los bloques residuales y / o bloques de referencia reconstruidos, según el ejemplo. El componente de filtro en bucle 329 puede ser sustancialmente similar al componente de análisis de control de filtro 227 y al componente de filtro en bucle 225. El componente de filtro en bucle 325 puede incluir múltiples filtros como se describe con respecto al componente de filtro en bucle 225. A continuación, los bloques filtrados se almacenan en un componente de búfer de imágenes decodificadas 323 para su uso como bloques de referencia por el componente de compensación de movimiento 321. El componente de búfer de imágenes decodificadas 323 puede ser sustancialmente similar al componente de búfer de imágenes decodificadas 223.

La FIG. 4 es un diagrama de bloques que ilustra un decodificador de vídeo de ejemplo 400. El decodificador 400 de video se puede emplear para implementar las funciones de decodificación del sistema de códec 200 y/o implementar los pasos 111, 113, 115 y / o 117 del método operativo 100. El decodificador 400 recibe un flujo de bits, por ejemplo, de un codificador 300, y genera una señal de vídeo de salida reconstruida basada en el flujo de bits para su visualización a un usuario final.

El flujo de bits es recibido por un componente de decodificación por entropía 433. El componente de decodificación por entropía 433 está configurado para implementar un esquema de decodificación de entropía, tal como CAVLC, CABAC, SBAC, codificación PIPE u otras técnicas de codificación de entropía. Por ejemplo, el componente de decodificación por entropía 433 puede emplear información de encabezado para proporcionar un contexto para interpretar datos adicionales codificados como palabras de código en el flujo de bits. La información decodificada incluye cualquier información deseada para decodificar la señal de video, tal como datos de control general, datos de control de filtro, información de partición, datos de movimiento, datos de predicción y coeficientes de transformación cuantificados de bloques residuales. Los coeficientes de transformación cuantificados se envían a un componente de transformación y cuantificación inversa 429 para su reconstrucción en bloques residuales. El componente de cuantificación y transformación inversa 429 puede ser similar al componente de cuantificación y transformación inversa 329.

Los bloques residuales reconstruidos y / o los bloques de predicción se envían al componente de predicción intraimagen 417 para su reconstrucción en bloques de imagen basados en operaciones de intrapredicción. El componente de predicción intraimagen 417 puede ser similar al componente de estimación intraimagen 215 y al componente de predicción intraimagen 217. Específicamente, el componente de predicción intraimagen 417 emplea modos de predicción para localizar un bloque de referencia en el fotograma y aplica un bloque residual al resultado para

reconstruir bloques de imágenes intrapredichos. Los bloques de imagen intrapredichos reconstruidos y / o los bloques residuales y los datos de interpredicción correspondientes se envían a un componente de búfer de imágenes decodificadas 423 a través de un componente de filtro en bucle 425, que puede ser sustancialmente similar al componente de búfer de imágenes decodificadas 223 y al componente de filtro de bucle 225, respectivamente. El componente de filtro en bucle 425 filtra los bloques de imágenes reconstruidas, los bloques residuales y / o los bloques de predicción, y dicha información se almacena en el componente de búfer de imágenes decodificadas 423. Los bloques de imágenes reconstruidas del componente de búfer de imágenes decodificadas 423 se envían a un componente de compensación de movimiento 421 para la interpredicción. El componente de compensación de movimiento 421 puede ser sustancialmente similar al componente de estimación de movimiento 221 y/o al componente de compensación de movimiento 219. Específicamente, el componente de compensación de movimiento 421 emplea vectores de movimiento de un bloque de referencia para generar un bloque de predicción y aplica un bloque residual al resultado para reconstruir un bloque de imagen. Los bloques reconstruidos resultantes también pueden enviarse a través del componente de filtro en bucle 425 al componente de búfer de imágenes decodificadas 423. El componente de búfer de imágenes decodificadas 423 continúa almacenando bloques de imágenes reconstruidas adicionales, que pueden reconstruirse en fotogramas a través de la información de partición. Estos fotogramas también se pueden colocar en una secuencia. La secuencia se envía a una pantalla como una señal de vídeo de salida reconstruida.

La FIG. 5 es un diagrama esquemático que ilustra un flujo de bits 500 y un subflujo de bits 501 extraídos del flujo de bits 500 de ejemplo. Por ejemplo, el flujo de bits 500 se puede generar mediante un sistema de códec 200 y / o un codificador 300 para decodificar mediante un sistema de códec 200 y / o un decodificador 400. Como otro ejemplo, el flujo de bits 500 puede ser generado por un codificador en el paso 109 del método 100 para ser utilizado por un decodificador en el paso 111.

El flujo de bits 500 incluye un conjunto de parámetros de secuencia (SPS) 510, una pluralidad de conjuntos de parámetros de imagen (PPS) 512, una pluralidad de encabezados de segmento 514, datos de imagen 520, y uno o más mensajes SEI 515. Un SPS 510 contiene datos de secuencia comunes a todas las imágenes de la secuencia de vídeo contenida en el flujo de bits 500. Dichos datos pueden incluir el tamaño de imagen, la profundidad de bits, los parámetros de la herramienta de codificación, las restricciones de tasa de bits, etc. El PPS 512 contiene parámetros que son específicos de una o más imágenes correspondientes. Por tanto, cada imagen de una secuencia de vídeo puede referirse a un PPS 512. El PPS 512 puede indicar herramientas de codificación disponibles para mosaicos en imágenes correspondientes, parámetros de cuantificación, compensaciones, parámetros de herramientas de codificación específicas de imagen (por ejemplo, controles de filtro), etc. El encabezado de segmento 514 contiene parámetros que son específicos para uno o más segmentos correspondientes 524 en una imagen. Por tanto, cada segmento 524 en una secuencia de vídeo puede referirse a encabezado de segmento 514. El encabezado de segmento 514 puede contener información de tipo de segmento, conteos/recuentos de orden de imagen (POCs), listas de imágenes de referencia, pesos/ponderaciones de predicción, puntos de entrada de mosaico, parámetros de desbloqueo, etc. En algunos ejemplos, los segmentos 524 pueden denominarse grupos de mosaicos. En tal caso, el encabezado de segmento 514 puede denominarse encabezado de grupo de mosaicos. Los mensajes SEI 515 son mensajes opcionales que contienen metadatos que no son necesarios para la decodificación de bloques, pero que pueden emplearse para fines relacionados, como indicar el tiempo de salida de la imagen, la configuración de la pantalla, la detección de pérdida, la ocultación de pérdida, etc.

Los datos de imagen 520 contienen datos de vídeo codificados de acuerdo con la interpredicción y / o intrapredicción, así como los correspondientes datos residuales transformados y cuantificados. Dichos datos de imagen 520 se clasifican de acuerdo con una partición utilizada para particionar la imagen antes de la codificación. Por ejemplo, la secuencia de video se divide en imágenes 521. Las imágenes 521 pueden dividirse además en subimágenes 522, que se dividen en segmentos 524. Los segmentos 524 se pueden dividir además en mosaicos y/o CTUs. Las CTUs se dividen además en bloques de codificación basados en árboles de codificación. A continuación, los bloques de codificación se pueden codificar / decodificar de acuerdo con los mecanismos de predicción. Por ejemplo, una imagen 521 puede contener una o más subimágenes 522. Una subimagen 522 contener uno o más segmentos 524. La imagen 521 se refiere al PPS 512 y los segmentos 524 se refieren al encabezado de segmento 514. Las subimágenes 522 pueden particionarse de forma coherente en una secuencia de vídeo completa (también conocida como segmento) y, por tanto, pueden referirse al SPS 510. Cada segmento 524 puede contener uno o más mosaicos. Cada segmento 524, y por tanto la imagen 521 y la subimagen 522, también pueden contener una pluralidad de CTUs.

Cada imagen 521 puede contener un conjunto completo de datos visuales asociados con una secuencia de video para un instante correspondiente en el tiempo. Sin embargo, es posible que determinadas aplicaciones deseen mostrar sólo una parte de una imagen 521 en algunos casos. Por ejemplo, un sistema de realidad virtual (VR) puede mostrar una región de la imagen 521 seleccionada por el usuario, que crea la sensación de estar presente en la escena representada en la imagen 521. La región que un usuario puede desear ver no se conoce cuando el flujo de bits 500 está codificado. Por consiguiente, la imagen 521 puede contener cada región posible que un usuario puede ver potencialmente como subimágenes 522, que pueden decodificarse y visualizarse por separado basándose en la entrada del usuario. Otras aplicaciones pueden mostrar por separado una región de interés. Por ejemplo, un televisor con una imagen en una imagen puede desear mostrar una región particular, y por lo tanto una subimagen 522, de una secuencia de video sobre una imagen 521 de una secuencia de video no relacionada. En otro ejemplo más, los sistemas de teleconferencia pueden mostrar una imagen completa 521 de un usuario que está hablando en ese momento y una

subimagen 522 de un usuario que no está hablando en ese momento. Por consiguiente, una subimagen 522 puede contener una región definida de la imagen 521. Una subimagen 522 que está temporalmente restringida de movimiento puede decodificarse por separado del resto de la imagen 521. Específicamente, una subimagen restringida de movimiento temporal se codifica sin referencia a muestras fuera de la subimagen restringida de movimiento temporal y, por tanto, contiene información suficiente para la decodificación completa sin referencia al resto de la imagen 521.

Cada segmento 524 puede ser un rectángulo definido por una CTU en una esquina superior izquierda y una CTU en una esquina inferior derecha. En algunos ejemplos, un segmento 524 incluye una serie de mosaicos y/o CTUs en un orden de exploración de trama que procede de izquierda a derecha y de arriba a abajo. En otros ejemplos, un segmento 524 es un segmento rectangular. Un segmento rectangular puede no atravesar todo el ancho de una imagen de acuerdo con un orden de exploración de trama. En su lugar, un segmento rectangular puede contener una región rectangular y/o cuadrada de una imagen 521 y/o una subimagen 522 definida en términos de una CTU y/o filas de mosaicos y una CTU y/o columnas de mosaicos. Un segmento 524 es la unidad más pequeña que un decodificador puede mostrar por separado. Por tanto, los segmentos 524 de una imagen 521 pueden asignarse a diferentes subimágenes 522 para representar por separado las regiones deseadas de una imagen 521.

Un decodificador puede mostrar una o más subimágenes 523 de la imagen 521. Las subimágenes 523 son un subgrupo de subimágenes 522 seleccionado por el usuario o un subgrupo predefinido. Por ejemplo, una imagen 521 se puede dividir en nueve subimágenes 522, pero el decodificador puede mostrar solo una única subimagen 523 del grupo de subimágenes 522. Las subimágenes 523 contienen segmentos 525, que son un subgrupo de segmentos 524 seleccionado o predefinido. Para permitir la visualización separada de las subimágenes 523, puede extraerse un subflujo de bits 501 del flujo de bits 500. La extracción 529 puede ocurrir en el lado del codificador de modo que el decodificador solo reciba el subflujo de bits 501. En otros casos, todo el flujo de bits 500 se transmite al decodificador y el decodificador extrae 529 el subflujo de bits 501 para una decodificación separada. Cabe señalar que el subflujo de bits 501 también puede denominarse generalmente como un flujo de bits en algunos casos. Un subflujo de bits 501 incluye el SPS 510, el PPS 512, las subimágenes 523 seleccionadas, así como los encabezados de segmento 514 y los mensajes SEI 515 que son relevantes para las subimágenes 523 y/o los segmentos 525.

La presente divulgación señala varios datos para soportar una codificación eficaz de las subimágenes 522 para la selección y visualización de las subimágenes 523 en el decodificador. El SPS 510 incluye un tamaño de subimagen 531, una ubicación de subimagen 532 e IDs de subimagen 533 relacionados con el conjunto completo de subimágenes 522. El tamaño de subimagen 531 incluye una altura de subimagen en muestras de luma y un ancho de subimagen en muestras de luma para una subimagen correspondiente 522. La ubicación de subimagen 532 incluye una distancia de desplazamiento entre una muestra superior izquierda de una subimagen correspondiente 522 y una muestra superior izquierda de la imagen 521. La ubicación de subimagen 532 y el tamaño de subimagen 531 define un diseño de la subimagen correspondiente 552. El ID de subimagen 533 contiene datos que identifican de forma única una subimagen correspondiente 522. El ID de subimagen 533 puede ser un índice de exploración de trama de la subimagen 522 u otro valor definido. Por tanto, un decodificador puede leer el SPS 510 y determinar el tamaño, la ubicación y el ID de cada subimagen 522. En algunos sistemas de codificación de vídeo, los datos relacionados con las subimágenes 522 pueden incluirse en el PPS 512 porque una subimagen 522 está particionada a partir de una imagen 521. Sin embargo, las particiones utilizadas para crear subimágenes 522 pueden ser utilizadas por aplicaciones, tales como aplicaciones basadas en ROI, aplicaciones de VR/RV, etc., que dependen de particiones de subimagen 522 consistentes sobre una secuencia/segmento de vídeo. Como tal, las particiones de subimagen 522 generalmente no en una base de imagen por imagen. Colocar información de diseño para subimágenes 522 en el SPS 510 asegura que el diseño solo se señalice una vez para una secuencia/segmento en lugar de señalizarse de forma redundante para cada PPS 512 (que puede señalizarse para cada imagen 521 en algunos casos). Además, señalizar información de subimagen 522, en lugar de depender del decodificador para derivar dicha información, reduce la posibilidad de error en caso de pérdida de paquetes y soporta una funcionalidad adicional en términos de extracción de subimágenes 523. Por consiguiente, señalizar el diseño de subimagen 522 en el SPS 510 mejora la funcionalidad de un codificador y/o decodificador.

El SPS 510 también contiene banderas/indicadores 534 de subimágenes restringidas de movimiento relacionados con el conjunto completo de subimágenes 522. Las banderas/indicadores 534 de subimágenes restringidas de movimiento indican si cada subimagen 522 es una subimagen restringida de movimiento temporal. Por tanto, el decodificador puede leer las banderas 534 de subimágenes restringidas de movimiento y determinar cuál de las subimágenes 522 se puede extraer y visualizar por separado sin decodificar otras subimágenes 522. Esto permite que las subimágenes 522 seleccionadas se codifiquen como subimágenes restringidas de movimiento temporal mientras que permite codificar otras subimágenes 522 sin tales restricciones para una mayor eficiencia de codificación.

Los ID de subimagen 533 también se incluyen en los encabezados de segmento 514. Cada encabezado de segmento 514 contiene datos relevantes para un conjunto correspondiente de segmentos 524. Por consiguiente, el encabezado de segmento 514 contiene solo los ID de subimagen 533 correspondientes a los segmentos 524 asociados con el encabezado de segmento 514. Como tal, un decodificador puede recibir un segmento 524, obtener un ID de subimagen 533 del encabezado de segmento 514 y determinar qué subimagen 522 contiene el segmento 524. El decodificador también puede usar el ID de subimagen 533 del encabezado de segmento 514 para correlacionar con los datos relacionados en el SPS 510. Como tal, el decodificador puede determinar cómo colocar las subimágenes 522/523 y los segmentos 524/525 leyendo el SPS 510 y los encabezados 514 de los segmentos relevantes. Esto permite

decodificar las subimágenes 523 y los segmentos 525 incluso si algunas subimágenes 522 se pierden en la transmisión o se omiten intencionalmente para aumentar la eficacia de la codificación.

El mensaje SEI 515 también puede contener un nivel de subimagen 535. El nivel de subimagen 535 indica los recursos de hardware necesarios para decodificar una subimagen correspondiente 522. De esta manera, cada subimagen 522 puede codificarse independientemente de otras subimágenes 522. Esto asegura que se pueda asignar a cada subimagen 522 la cantidad correcta de recursos de hardware en el decodificador. Sin tal nivel de subimagen 535, a cada subimagen 522 se le asignarían suficientes recursos para decodificar la subimagen más compleja 522. Por tanto, el nivel de subimagen 535 evita que el decodificador sobreasigne recursos de hardware si las subimágenes 522 están asociadas con requisitos de recursos de hardware variables.

La FIG. 6 es un diagrama esquemático que ilustra un ejemplo de imagen 600 particionada en subimágenes 622. Por ejemplo, una imagen 600 puede codificarse y decodificarse a partir de un flujo de bits 500, por ejemplo, mediante un sistema de códec 200, un codificador 300 y/o un decodificador 400. Además, la imagen 600 puede particionarse y/o incluirse en un subflujo de bits 501 para soportar la codificación y decodificación de acuerdo con el método 100.

La imagen 600 puede ser sustancialmente similar a una imagen 521. Además, la imagen 600 puede particionarse en subimágenes 622, que son sustancialmente similares a las subimágenes 522. Cada una de las subimágenes 622 incluye un tamaño de subimagen 631, que puede incluirse en un flujo de bits 500 como un tamaño de subimagen 531. El tamaño de subimagen 631 incluye un ancho de subimagen 631a y una altura de subimagen 631b. El ancho de subimagen 631a es el ancho de una subimagen correspondiente 622 en unidades de muestras de luma. La altura de subimagen 631b es la altura de una subimagen correspondiente 622 en unidades de muestras de luma. Cada una de las subimágenes 622 incluye un ID de subimagen 633, que puede estar incluido en un flujo de bits 500 como un ID de subimagen 633. El ID de subimagen 633 puede ser cualquier valor que identifique de forma única cada subimagen 622. En el ejemplo mostrado, el ID de subimagen 633 es un índice de subimagen 622. Cada una de las subimágenes 622 incluye una ubicación 632, que puede estar incluida en un flujo de bits 500 como una ubicación de subimagen 532. La ubicación 632 se expresa como un desplazamiento entre la muestra superior izquierda de una subimagen correspondiente 622 y una muestra superior izquierda 642 de la imagen 600.

También como se muestra, algunas subimágenes 622 pueden ser subimágenes restringidas de movimiento temporal 634 y otras subimágenes 622 pueden no serlo. En el ejemplo mostrado, la subimagen 622 con un ID de subimagen 633 de cinco es una subimagen 634 con limitación de movimiento temporal. Esto indica que la subimagen 622 identificada como cinco está codificada sin referencia a ninguna otra subimagen 622 y, por lo tanto, puede extraerse y decodificarse por separado sin considerar los datos de las otras subimágenes 622. Una indicación de qué subimágenes 622 son subimágenes restringidas de movimiento temporal 634 puede ser señalizada en un flujo de bits 500 en banderas de subimágenes restringidas de movimiento 534.

Como se muestra, las subimágenes 622 se pueden restringir para cubrir una imagen 600 sin un espacio o superposición. Un espacio/hueco es una región de una imagen 600 que no está incluida en ninguna subimagen 622. Una superposición es una región de una imagen 600 que se incluye en más de una subimagen 622. En el ejemplo mostrado en la FIG. 6, las subimágenes 622 están particionadas a partir la imagen 600 para evitar tanto espacios como superposiciones. Los espacios hacen que las muestras de imagen 600 se queden fuera de las subimágenes 622. Las superposiciones hacen que los segmentos asociados se incluyan en múltiples subimágenes 622. Por lo tanto, los espacios y superposiciones pueden hacer que las muestras se vean afectadas por el tratamiento diferencial cuando las subimágenes 622 se codifican de forma diferente. Si esto está permitido en el codificador, un decodificador debe soportar tal esquema de codificación incluso cuando el esquema de decodificación se usa raramente. Al no permitir los espacios y superposiciones de la subimagen 622, la complejidad del decodificador puede reducirse ya que no se requiere que el decodificador tenga en cuenta los posibles huecos y superposiciones al determinar los tamaños 631 y las ubicaciones 632 de subimagen. Además, no permitir los espacios y superposiciones de la subimagen 622 reduce la complejidad de los procesos RDO en el codificador. Esto se debe a que el codificador puede omitir considerar los casos de espacios y superposición al seleccionar una codificación para una secuencia de video. Por consiguiente, evitar espacios y superposiciones puede reducir el uso de recursos de memoria y/o recursos de procesamiento en el codificador y el decodificador.

La FIG. 7 es un diagrama esquemático que ilustra un mecanismo de ejemplo 700 para relacionar los segmentos 724 con un diseño de subimagen 722. Por ejemplo, el mecanismo 700 puede aplicarse a la imagen 600. Además, el mecanismo 700 se puede aplicar basándose en datos en un flujo de bits 500, por ejemplo mediante un sistema de códec 200, un codificador 300 y/o un decodificador 400. Además, el mecanismo 700 se puede emplear para soportar la codificación y decodificación de acuerdo con el método 100.

El mecanismo 700 se puede aplicar a los segmentos 724 en una subimagen 722, tales como los segmentos 524/525 y las subimágenes 522/523, respectivamente. En el ejemplo mostrado, la subimagen 722 incluye un primer segmento 724a, un segunda segmento 724b y un tercer segmento 724c. Los encabezados de segmento para cada uno de los segmentos 724 incluyen un ID de subimagen 733 para la subimagen 722. El decodificador puede hacer coincidir el ID de subimagen 733 del encabezado de segmento con el ID de subimagen 733 en el SPS. El decodificador puede entonces determinar la ubicación 732 y el tamaño de subimagen 722 a partir del SPS basándose en el ID de la subimagen

733. Usando la ubicación 732, la subimagen 722 puede colocarse con relación a la muestra superior izquierda en la esquina superior izquierda 742 de la imagen. El tamaño se puede usar para establecer la altura y el ancho de la subimagen 722 en relación con la ubicación 732. Los segmentos 724 pueden incluirse entonces en la subimagen 722. Por consiguiente, los segmentos 724 se pueden colocar en la subimagen 722 correcta en base al ID de subimagen 733 sin referencia a otras subimágenes. Esto admite la corrección de errores ya que otras subimágenes perdidas no alteran la decodificación de la subimagen 722. Esto también admite aplicaciones que solo extraen una subimagen 722 y evita la transmisión de otras subimágenes. Por tanto, los IDs de subimagen 733 soportan una mayor funcionalidad y/o una mayor eficiencia de codificación, lo que reduce el uso de recursos de red, recursos de memoria y/o recursos de procesamiento en el codificador y el decodificador.

La FIG. 8 es un diagrama esquemático que ilustra otra imagen de ejemplo 800 particionada en subimágenes 822. La imagen 800 puede ser sustancialmente similar a la imagen 600. Además, una imagen 800 puede codificarse y descodificarse a partir de un flujo de bits 500, por ejemplo, mediante un sistema de códec 200, un codificador 300 y/o un decodificador 400. Además, la imagen 800 se puede dividir y/o incluir en un subflujo de bits 501 para soportar la codificación y decodificación de acuerdo con el método 100 y/o el mecanismo 700.

La imagen 800 incluye subimágenes 822, que pueden ser sustancialmente similares a las subimágenes 522, 523, 622 y/o 722. Las subimágenes 822 se dividen en una pluralidad de CTUs 825. Un CTU 825 es una unidad de codificación básica en sistemas de codificación de video estandarizados. Una CTU 825 se subdivide mediante un árbol de codificación en bloques de codificación, que se codifican de acuerdo con interpredicción o la intrapredicción. Como se muestra, algunas subimágenes 822a están restringidas para incluir anchos de subimagen y alturas de subimagen que son múltiplos del tamaño de CTU 825. En el ejemplo que se muestra, las subimágenes 822a tienen una altura de seis CTUs 825 y un ancho de cinco CTUs 825. Esta restricción se elimina para las subimágenes 822b colocadas en el borde derecho 801 de las imágenes y para las subimágenes 822c ubicadas en el borde inferior de las imágenes 802. En el ejemplo que se muestra, las subimágenes 822b tienen un ancho de entre cinco y seis CTUs 825. Sin embargo, las subimágenes 822b que no están colocadas en el borde inferior 802 de las imágenes todavía están limitadas para mantener una altura de la subimagen que es un múltiplo del tamaño de CTU 825. En el ejemplo que se muestra, las subimágenes 822c tienen una altura de entre seis y siete CTUs 825. Sin embargo, las subimágenes 822c que no están colocadas en el borde derecho 801 de las imágenes todavía están restringidas para mantener un ancho de subimagen que es un múltiplo del tamaño de CTU 825.

Como se señaló anteriormente, algunos sistemas de video pueden limitar las subimágenes 822 para incluir alturas y anchos que sean múltiplos del tamaño de CTU 825. Esto puede evitar que las subimágenes 822 funcionen correctamente con muchas disposiciones de imagen, por ejemplo, con una imagen 800 que contiene un ancho o alto total que no es un múltiplo del tamaño CTU 825. Al permitir que las subimágenes inferiores 822c y las subimágenes derechas 822b incluyan alturas y anchos, respectivamente, que no sean múltiplos del tamaño de CTU 825, las subimágenes 822 pueden usarse con cualquier imagen 800 sin provocar errores de decodificación. Esto da como resultado un aumento de la funcionalidad del codificador y decodificador. Además, la mayor funcionalidad permite que un codificador codifique imágenes de manera más eficiente, lo que reduce el uso de recursos de red, recursos de memoria y/o recursos de procesamiento en el codificador y el decodificador.

Como se describe en el presente documento, la presente divulgación describe diseños para la división de imágenes basada en subimágenes en la codificación de vídeo. Una subimagen es un área rectangular dentro de una imagen que se puede decodificar de forma independiente mediante un proceso de decodificación similar al que se utiliza para una imagen. La presente divulgación se refiere a la señalización de subimágenes en una secuencia de vídeo codificada y/o flujo de bits, así como al proceso de extracción de subimágenes. Las descripciones de las técnicas se basan en VVC por la JVET de ITU-T e ISO/IEC. Sin embargo, las técnicas también se aplican a otras especificaciones de códecs de video. Las siguientes son modalidades de ejemplo descritas en el presente documento. Tales modalidades se pueden aplicar individualmente o en combinación.

La información relacionada con las subimágenes que pueden estar presentes en la secuencia de vídeo codificada (CVS) puede indicarse en un conjunto de parámetros de nivel de secuencia, como un SPS. Dicha señalización puede incluir la siguiente información. El número de subimágenes que están presentes en cada imagen del CVS puede señalizarse en el SPS. En el contexto del SPS o un CVS, las subimágenes colocadas para todas las unidades de acceso (AUs) pueden denominarse colectivamente como una secuencia de subimágenes. También se puede incluir en el SPS un lazo/bucle para especificar más información que describe las propiedades de cada subimagen. Esta información puede comprender la identificación de la subimagen, la ubicación de subimagen (por ejemplo, la distancia de desplazamiento entre la muestra de luma de la esquina superior izquierda de la subimagen y la muestra de luma de la esquina superior izquierda de la imagen), y el tamaño de subimagen. Además, el SPS puede indicar si cada subimagen es una subimagen restringida de movimiento (que contiene la funcionalidad de un MCTS). La información de perfil, grado y nivel para cada subimagen también puede indicarse o derivarse en el decodificador. Dicha información puede emplearse para determinar información de perfil, grado y nivel para un flujo de bits creado extrayendo subimágenes del flujo de bits original. El perfil y el grado de cada subimagen puede derivarse para que sea el mismo que el perfil y el grado de todo el flujo de bits. El nivel de cada subimagen puede indicarse explícitamente. Tal señalización puede estar presente en el bucle contenido en el SPS. Los parámetros del decodificador de referencia hipotético (HRD) de nivel de

secuencia se pueden señalizar en la sección de información de usabilidad de video (VUI) del SPS para cada subimagen (o, de manera equivalente, cada secuencia de subimagen).

Cuando una imagen no está particionada en dos o más subimágenes, las propiedades de la subimagen (por ejemplo, ubicación, tamaño, etc.), excepto el ID de la subimagen, pueden no estar presentes/señalizadas en el flujo de bits. Cuando se extrae una subimagen de imágenes en un CVS, es posible que cada unidad de acceso en el nuevo flujo de bits no contenga subimágenes. En este caso, la imagen de cada AU en el nuevo flujo de bits no se particiona en múltiples subimágenes. Por tanto, no hay necesidad de señalar las propiedades de subimagen, como la ubicación y el tamaño, en el SPS, ya que dicha información puede derivarse de las propiedades de la imagen. Sin embargo, la identificación de subimagen todavía se puede señalizar, ya que el ID puede ser denominada por unidades/grupos de mosaicos VCL NAL que se incluyen en la subimagen extraída. Esto puede permitir que los IDs de las subimágenes permanezcan iguales al extraer la subimagen.

La ubicación de una subimagen en la imagen (desplazamiento xy desplazamiento y) se puede señalizar en unidades de muestras de luma. La ubicación representa la distancia entre la muestra de luma de la esquina superior izquierda de la subimagen y la muestra de luma de la esquina superior izquierda de la imagen. Alternativamente, la ubicación de una subimagen en la imagen se puede señalizar en unidades del tamaño mínimo del bloque de codificación de luma (MinCbSizeY). Alternativamente, la unidad de desplazamientos de ubicación de subimagen puede indicarse explícitamente mediante un elemento de sintaxis en un conjunto de parámetros. La unidad puede ser CtbSizeY, MinCbSizeY, muestra de luma u otros valores.

El tamaño de una subimagen (ancho de subimagen y altura de subimagen) se puede señalizar en unidades de muestras de luma. Alternativamente, el tamaño de una subimagen se puede señalizar en unidades del tamaño mínimo de bloque de codificación de luma (MinCbSizeY). Alternativamente, la unidad de los valores de tamaño de subimagen puede indicarse explícitamente mediante un elemento de sintaxis en un conjunto de parámetros. La unidad puede ser CtbSizeY, MinCbSizeY, muestra de luma u otros valores. Cuando el borde derecho de una subimagen no coincide con el borde derecho de la imagen, es posible que se requiera que el ancho de la subimagen sea un múltiplo entero del tamaño de CTU de luma (CtbSizeY). Asimismo, cuando el borde inferior de una subimagen no coincide con el borde inferior de la imagen, es posible que se requiera que la altura de la subimagen sea un múltiplo entero del tamaño de CTU de luma (CtbSizeY). Si el ancho de una subimagen no es un múltiplo entero del tamaño de CTU de luma, es posible que se requiera que la subimagen esté ubicada en la posición más a la derecha de la imagen. Asimismo, si la altura de una subimagen no es un múltiplo entero del tamaño de CTU de luma, es posible que se requiera que la subimagen esté ubicada en la posición más inferior de la imagen. En algunos casos, el ancho de una subimagen se puede señalar en unidades de tamaño de CTU de luma, pero el ancho de una subimagen no es un múltiplo entero del tamaño de CTU de luma. En este caso, el ancho real en las muestras de luma se puede derivar en función de la ubicación de desplazamiento de la subimagen. El ancho de la subimagen se puede derivar en función del tamaño de CTU de luma y la altura de la imagen se puede derivar en función de las muestras de luma. Asimismo, la altura de una subimagen puede indicarse en unidades de tamaño de CTU de luma, pero la altura de la subimagen no es un múltiplo entero del tamaño de CTU de luma. En tal caso, la altura real en la muestra de luma se puede derivar en función de la ubicación de desplazamiento de la subimagen. La altura de la subimagen se puede derivar en función del tamaño de CTU de luma y la altura de la imagen se puede derivar en función de las muestras de luma.

Para cualquier subimagen, el ID de la subimagen puede ser diferente del índice de subimagen. El índice de subimagen puede ser el índice de la subimagen como se indica en un bucle de subimágenes en el SPS. El ID de subimagen puede ser el índice de la subimagen en el orden de exploración de trama de subimagen en la imagen. Cuando el valor del ID de subimagen de cada subimagen es el mismo que el índice de la subimagen, el ID de la subimagen se puede señalizar o derivar. Cuando el ID de subimagen de cada subimagen es diferente del índice de subimagen, el ID de subimagen se señaliza explícitamente. El número de bits para la señalización de IDs de subimagen puede indicarse en el mismo conjunto de parámetros que contiene propiedades de subimagen (por ejemplo, en el SPS). Es posible que algunos valores de ID de subimagen se reserven para determinados fines. Por ejemplo, cuando los encabezados de grupo de mosaicos contienen ID de subimagen para especificar qué subimagen contiene un grupo de mosaicos, el valor cero puede reservarse y no usarse para subimágenes para garantizar que los primeros bits de un encabezado de grupo de mosaicos no sean todos ceros para evitar la inclusión accidental de un código de prevención de emulación. En casos opcionales en los que las subimágenes de una imagen no cubren toda el área de la imagen sin espacios y sin superposición, se puede reservar un valor (por ejemplo, el valor uno) para grupos de mosaicos que no forman parte de ninguna subimagen. Alternativamente, el ID de subimagen del área restante se señaliza explícitamente. El número de bits para señalizar el ID de la subimagen puede limitarse de la siguiente manera. El rango de valores debe ser suficiente para identificar de forma única todas las subimágenes en una imagen, incluidos los valores reservados de la identificación de la subimagen. Por ejemplo, el número mínimo de bits para el ID de subimagen puede ser el valor de Ceil(Log2 (número de subimágenes en un número de imagen del ID de subimagen reservado).

Puede estar restringido que la unión de subimágenes debe cubrir la imagen completa sin espacios y sin superposición. Cuando se aplica esta restricción, para cada subimagen, puede estar presente una bandera para especificar si la subimagen es una subimagen restringida de movimiento, lo que indica que la subimagen se puede extraer. Alternativamente, es posible que la unión de subimágenes no cubra la imagen completa, pero es posible que no se permitan superposiciones.

Los ID de subimagen pueden estar presentes inmediatamente después del encabezado de la unidad NAL para ayudar al proceso de extracción de la subimagen sin requerir que el extractor analice el resto de los bits de la unidad NAL. Para las unidades VCL NAL, el ID de subimagen puede estar presente en los primeros bits de los encabezados del grupo de mosaicos. Para unidades que no son VCL NAL, se puede aplicar lo siguiente. Para SPS, no es necesario que el ID de subimagen esté presente inmediatamente después del encabezado de la unidad NAL. Para PPS, si todos los grupos de mosaicos de la misma imagen están restringidos para hacer referencia al mismo PPS, no es necesario que el ID de la subimagen esté presente inmediatamente después de su encabezado de unidad NAL. Si se permite que los grupos de mosaicos de la misma imagen se refieran a diferentes PPS, el ID de la subimagen puede estar presente en los primeros bits de PPS (por ejemplo, inmediatamente después del encabezado de la unidad NAL). En este caso, se puede permitir que cualquier grupo de mosaicos de una imagen comparta el mismo PPS. Alternativamente, cuando se permite que los grupos de mosaicos de la misma imagen se refieran a diferentes PPS, y también se permite que diferentes grupos de mosaicos de la misma imagen compartan el mismo PPS, no puede haber ID de subimagen presente en la sintaxis de PPS. Alternativamente, cuando se permite que los grupos de mosaicos de la misma imagen se refieran a diferentes PPS, y también se permite que diferentes grupos de mosaicos de la misma imagen compartan el mismo PPS, una lista de ID de subimagen puede estar presente en la sintaxis de PPS. La lista indica las subimágenes a las que se aplica el PPS. Para otras unidades NAL no VCL, si la unidad no VCL se aplica al nivel de imagen o superior (p. Ej., delimitador de la unidad de acceso, fin de secuencia, fin de flujo de bits, etc.), es posible que el ID de subimagen no esté presente inmediatamente después del encabezado de unidad NAL. De lo contrario, el ID de subimagen puede estar presente inmediatamente después del encabezado de unidad NAL.

Con la señalización de SPS anterior, la partición de mosaicos dentro de subimágenes individuales se puede señalizar en el PPS. Se puede permitir que los grupos de mosaicos dentro de la misma imagen se refieran a diferentes PPS. En este caso, la agrupación de mosaicos solo puede estar dentro de cada subimagen. El concepto de agrupación de mosaicos consiste en particionar una subimagen en mosaicos.

Alternativamente, se define un conjunto de parámetros para describir la partición de mosaicos dentro de subimágenes individuales. Dicho conjunto de parámetros puede denominarse Conjunto de parámetros de subimagen (SPPS). El SPPS se refiere a SPS. Un elemento de sintaxis que hace referencia al ID de SPS está presente en SPPS. El SPPS puede contener un ID de subimagen. Para fines de extracción de subimagen, el elemento de sintaxis que hace referencia al ID de subimagen es el primer elemento de sintaxis en SPPS. El SPPS contiene una estructura de mosaicos (por ejemplo, varias columnas, varias filas, espaciado de mosaico uniforme, etc.) El SPPS puede contener una bandera para indicar si un filtro de bucle está habilitado o no a través de los límites de la subimagen asociados. Alternativamente, las propiedades de la subimagen para cada subimagen pueden indicarse en el SPPS en lugar de en el SPS. La partición de mosaicos dentro de subimágenes individuales aún se puede señalar en el PPS. Los grupos de mosaicos dentro de la misma imagen pueden hacer referencia a diferentes PPS. Una vez que se activa un SPPS, el SPPS dura una secuencia de AUs consecutivas en orden de decodificación. Sin embargo, el SPPS puede desactivarse/activarse en una AU que no es el inicio de un CVS. En cualquier momento durante el proceso de decodificación de un flujo de bits de una sola capa con múltiples subimágenes en algunas AUs, pueden estar activos múltiples SPPS. Un SPPS puede ser compartido por diferentes subimágenes de una AU. Alternativamente, SPPS y PPS se pueden combinar en un conjunto de parámetros. En tal caso, es posible que no se requiera que todos los grupos de mosaicos de la misma imagen se refieran al mismo PPS. Puede aplicarse una restricción de modo que todos los grupos de mosaicos en la misma subimagen puedan hacer referencia al mismo conjunto de parámetros resultante de la fusión entre SPPS y PPS.

El número de bits utilizados para señalizar el ID de subimagen puede indicarse en un encabezado de unidad NAL. Cuando está presente en un encabezado de unidad NAL, dicha información puede ayudar a los procesos de extracción de subimagen a analizar el valor de ID de subimagen al comienzo de la carga útil de una unidad NAL (por ejemplo, los primeros bits inmediatamente después del encabezado de unidad NAL). Para tal señalización, algunos de los bits reservados (por ejemplo, siete bits reservados) en un encabezado de unidad NAL pueden usarse para evitar aumentar la longitud del encabezado de unidad NAL. El número de bits para tal señalización puede cubrir el valor de sub-picture-ID-bit-len. Por ejemplo, cuatro bits de los siete bits reservados de un encabezado de unidad NAL de VVC pueden usarse para este propósito.

Al decodificar una subimagen, la ubicación de cada bloque de árbol de codificación (por ejemplo, xCtb e yCtb) puede ajustarse a una ubicación de muestra de luma real en la imagen en lugar de una ubicación de muestra de luma en la subimagen. De esta manera, se puede evitar la extracción de una subimagen co-ubicada de cada imagen de referencia ya que el bloque de árbol de codificación se decodifica con referencia a la imagen en lugar de a la subimagen. Para ajustar la ubicación de un bloque de árbol de codificación, las variables SubpictureXOffset y SubpictureYOffset se pueden derivar en función de la posición de la subimagen (subpic_x_offset y subpic_y_offset). Los valores de las variables se pueden sumar a los valores de las coordenadas x e y de la ubicación de la muestra de luma, respectivamente, de cada bloque de árbol de codificación en la subimagen.

Un proceso de extracción de subimágenes se puede definir de la siguiente manera. La entrada al proceso es la subimagen de destino que se va a extraer. Esto puede ser en forma de ID de subimagen o ubicación de subimagen. Cuando la entrada es la ubicación de una subimagen, el ID de subimagen asociado se puede resolver analizando la información de la subimagen en el SPS. Para unidades que no son VCL NAL, se aplica lo siguiente. Los elementos de

sintaxis en el SPS relacionados con el tamaño y el nivel de la imagen pueden actualizarse con la información de tamaño y nivel de la subimagen. Las siguientes unidades NAL que no son VCL se mantienen sin cambios: PPS, delimitador de unidad de acceso (AUD), fin de secuencia (EOS), fin de flujo de bits (EOB) y cualquier otra unidad NAL no VCL que sea aplicable al nivel de imagen o superior. Las restantes unidades NAL no VCL con ID de subimagen no igual al ID de subimagen destino/objetivo pueden eliminarse. Las unidades VCL NAL con un ID de subimagen diferente al ID de subimagen del objetivo también pueden eliminarse.

Puede usarse un mensaje SEI de anidamiento de subimagen de nivel de secuencia para anidar mensajes SEI de nivel AU o nivel de subimagen para un conjunto de subimágenes. Esto puede incluir un período de almacenamiento en búfer, temporización de imagen y mensajes SEI no HRD. La sintaxis y semántica de este mensaje SEI de anidamiento de subimagen puede ser la siguiente. Para operaciones de sistemas, como en entornos de formato de medios omnidireccionales (OMAF), el reproductor OMAF puede solicitar y decodificar un conjunto de secuencias de subimágenes que cubren una ventana de visualización ("viewport"). Por lo tanto, el mensaje SEI de nivel de secuencia se usa para transportar información de un conjunto de secuencias de subimagen que cubren colectivamente una región de imagen rectangular. Los sistemas pueden utilizar la información, y la información es indicativa de la capacidad de decodificación requerida, así como de la tasa de bits del conjunto de secuencias de subimagen. La información indica el nivel del flujo de bits que incluye solo el conjunto de secuencias de subimágenes. Esta información también indica la tasa de bits del flujo de bits que contiene solo el conjunto de secuencias de subimágenes. Opcionalmente, puede especificarse un proceso de extracción de subflujo de bits para un conjunto de secuencias de subimagen. El beneficio de hacer esto es que el flujo de bits que incluye solo un conjunto de secuencias de subimagen también puede ser conforme. Una desventaja es que al considerar las diferentes posibilidades de tamaño de la ventana de visualización, puede haber muchos de estos conjuntos además de los ya grandes números posibles de secuencias de subimágenes individuales.

En una modalidad de ejemplo, uno o más de los ejemplos descritos pueden implementarse como sigue. Una subimagen puede definirse como una región rectangular de uno o más grupos de mosaicos dentro de una imagen. Un proceso de división binaria permitido puede definirse como sigue. Las entradas a este proceso son: un modo de división binaria btSplit, un ancho de bloque de codificación cbWidth, una altura de bloque de codificación cbHeight, una ubicación (x0, y0) de la muestra de luma superior izquierda del bloque de codificación considerado en relación con la parte superior izquierda luma muestra de la imagen, una profundidad de árbol de múltiples tipos mttDepth, una profundidad máxima de árbol de múltiples tipos con desplazamiento maxMttDepth, un tamaño máximo de árbol binario maxBtSize y un índice de partición partIdx. El resultado de este proceso es la variable allowBtSplit.

	btSplit = = SPLIT_BT_VER	btSplit = = SPLIT_BT_HOR
parallelTtSplit	SPLIT_TT_VER	SPLIT_TT_HOR
cbSize	cbWidth	cbHeight

Especificación de parallelTtSplit y cbSize basado en btSplit

Las variables parallelTtSplit y cbSize se derivan como se especificó anteriormente. La variable allowBtSplit puede derivarse como sigue. Si una o más de las siguientes condiciones son verdaderas, allowBtSplit se establece igual a FALSE: cbSize es menor o igual que MinBtSizeY, cbWidth es mayor que maxBtSize, cbHeight es mayor que maxBtSize y mttDepth es mayor o igual que maxMttDepth. De lo contrario, si se cumplen todas las condiciones siguientes, allowBtSplit se establece igual a FALSE: btSplit es igual a SPLIT_BT_VER y y0 + cbHeight es mayor que SubPicBottomBorderInPic. De lo contrario, si se cumplen todas las condiciones siguientes, allowBtSplit se establece igual a FALSE, btSplit es igual a SPLIT_BT_HOR, x0+cbWidth es mayor que SubPicRightBorderInPic y y0+cbHeight es menor o igual que SubPicBottomBorderInPic. De lo contrario, si todas las condiciones siguientes son verdaderas, allowBtSplit se establece igual a FALSE: mttDepth es mayor que cero, partIdx es igual a uno y MttSplitMode[x0] y0] [mttDepth - 1] es igual a parallelTtSplit. De lo contrario, si se cumplen todas las condiciones siguientes, allowBtSplit se establece igual a FALSE: btSplit es igual a SPLIT_BT_VER, cbWidth es menor o igual a MaxTbSizeY y cbHeight es mayor que MaxTbSizeY. De lo contrario, si se cumplen todas las condiciones siguientes, allowBtSplit se establece igual a FALSE: btSplit es igual a SPLIT_BT_HOR, cbWidth es mayor que MaxTbSizeY y cbHeight es menor o igual que MaxTbSizeY. De lo contrario, allowBtSplit se establece igual a TRUE.

Un proceso de división ternaria permitido puede definirse como sigue. Las entradas para este proceso son: un modo de división ternaria ttSplit, un ancho de bloque de codificación cbWidth, una altura de bloque de codificación cbHeight, una ubicación (x0, y0) de la muestra de luma superior izquierda del bloque de codificación considerado en relación con la luma superior izquierda muestra de la imagen, una profundidad de árbol de múltiples tipos mttDepth, una profundidad máxima de árbol de múltiples tipos con desplazamiento maxMttDepth y un tamaño máximo de árbol binario maxTtSize. El resultado de este proceso es la variable allowTtSplit.

	ttSplit = = SPLIT_TT_VER	ttSplit = = SPLIT_TT_HOR
cbSize	cbWidth	cbHeight

Especificación de cbSize basada en ttSplit.

La variable cbSize se deriva como se especificó anteriormente. La variable allowTtSplit puede derivarse como sigue. Si una o más de las siguientes condiciones son verdaderas, allowTtSplit se establece igual a FALSE: cbSize es menor o igual a 2 * MinTtSizeY, cbWidth es mayor que Min (MaxTbSizeY, maxTtSize), cbHeight es mayor que Min (MaxTbSizeY, maxTtSize) , mttDepth es mayor o igual que maxMttDepth, x0+cbWidth es mayor que SubPicRightBorderInPic y y0+cbHeight es mayor que SubPicBottomBorderInPic. De lo contrario, allowTtSplit se establece igual a TRUE.

La sintaxis y la semántica del conjunto de parámetros de secuencia de RBSP son las siguientes.

seq_parameter_set_rbsp( ) {	Descriptor
sps_seq_parameter_set_id	ue(v)
pic_width_in_luma_samples	ue(v)
pic_height_in_luma_samples	ue(v)
num_subpic_minus1	ue(v)
subpic_id_len_minus1	ue(v)
for (i = 0; i <= num_subpic_minus1; i++) {	
subpic_id[i]	u(v)
if(num_subpic_minus1 > 0) {	
subpic_level_idc[i]	u(8)
subpic_x_offset[i]	ue(v)
subpic_y_offset[i]	ue(v)
subpic_width_in_luma_samples[i]	ue(v)
subpic_height_in_luma_samples[i]	ue(v)
subpic_motion_constrained_flag[i]	u(1)
}	
}	
...	
}	

Pic_width_in_luma_samples especifica el ancho de cada imagen decodificada en unidades de muestras de luma. pic_width_in_luma_samples no debe ser igual a cero y debe ser un múltiplo entero de MinCbSizeY. Pic_height_in_luma_samples especifica la altura de cada imagen decodificada en unidades de muestras de luma. pic_height_in_luma_samples no debe ser igual a cero y debe ser un múltiplo entero de MinCbSizeY. Num_subpicture_minus1 más 1 especifica el número de subimágenes divididas en imágenes codificadas que pertenecen a la secuencia de vídeo codificada. El subpic_id_len_minus1 más 1 especifica el número de bits utilizados para representar el elemento de sintaxis subpic_id [i] en SPS, spps_subpic_id en SPPS en referencia a SPS y tile_group_subpic_id en encabezados de grupo de mosaicos que se refieren a SPS. El valor de subpic_id_len_minus1 deberá estar comprendido entre Ceil(Log2(num_subpic_minus1 + 2) a ocho, inclusive. El subpic_id [i] especifica el ID de subimagen de la subimagen i-ésima de imágenes que se refieren al SPS. La longitud de subpic_id [i] es subpic_id_len_minus1+1 bits. El valor de subpic_id [i] será mayor que cero. El subpic_level_idc [i] indica un nivel para el cual el CVS resultó de la extracción de las i-ésimas subimágenes que se ajustan a los requisitos de recursos especificados. Los flujos de bits no deben contener valores de subpic_level_idc [i] distintos de los especificados. Otros valores de subpic_level_idc [i] están reservados. Cuando no está presente, se infiere que el valor de subpic_level_idc[i] es igual al valor de general_level_idc.

El subpic_x_offset [i] especifica el desplazamiento horizontal de la esquina superior izquierda de la subimagen i-ésima en relación con la esquina superior izquierda de la imagen. Cuando no está presente, se infiere que el valor de subpic_x_offset[i] es igual a 0. El valor del desplazamiento x de la subimagen se obtiene de la siguiente manera: SubpictureXOffset[i] = subpic_x_offset[i]. El subpic_y_offset[i] especifica el desplazamiento vertical de la esquina superior izquierda de la subimagen i-ésima en relación con la esquina superior izquierda de la imagen. Cuando no está presente, se infiere que el valor de subpic_y_offset[i] es igual a cero. El valor del desplazamiento y de la subimagen se deriva de la siguiente manera: SubpictureYOffset [i] = subpic_y_offset[i]. El subpic_width_in_luma_samples [i] especifica el ancho de la i-ésima subimagen decodificada para la que este SPS es el SPS activo. Cuando la suma de SubpictureXOffset [i] y subpic_width_in_luma_samples[i] es menor que pic_width_in_luma_samples, el valor de subpic_width_in_luma_samples[i] será un múltiplo entero de CtbSizeY. Cuando no está presente, se infiere que el valor de subpic_width_in_luma_samples[i] es igual al valor de pic_width_in_luma_samples. El subpic_height_in_luma_samples[i] especifica la altura de la i-ésima subimagen decodificada para la que este SPS es el SPS activo. Cuando la suma de SubpictureYOffset[i] y subpic_height_in_luma_samples[i] es menor que pic_height_in_luma_samples, el valor de subpic_height_in_luma_samples[i] será un múltiplo entero de CtbSizeY. Cuando no está presente, se infiere que el valor de subpic_height_in_luma_samples[i] es igual al valor de pic_height_in_luma_samples.

Es un requisito de la conformidad del flujo de bits que la unión de subimágenes cubra toda el área de una imagen sin superposiciones ni espacios. El subpic_motion_constrained_flag[i] igual a uno especifica que la i-ésima subimagen es una subimagen restringida de movimiento temporal. El subpic_motion_constrained_flag[i] igual a cero especifica que

la i-ésima subimagen puede ser o no una subimagen restringida de movimiento temporal. Cuando no está presente, se infiere que el valor de subpic_motion_constrained_flag es igual a cero.

Las variables SubpicWidthInCtbsY, SubpicHeightInCtbsY, SubpicSizeInCtbsY, SubpicWidthInMinCbsY, SubpicHeightInMinCbsY, SubpicSizeInMinCbsY, SubpicSizeInSamplesY, SubpicWidthInSamplesC y SubpicHeightIn se derivan de la siguiente manera:

SubpicWidthInLumaSamples[i] = subpic_width_in_luma_samples[i]

SubpicHeightInLumaSamples[i] = subpic_height_in_luma_samples[i]

SubPicRightBorderInPic[i] = SubpictureXOffset[i] + PicWidthInLumaSamples[i]

SubPicBottomBorderInPic[i] = SubpictureYOffset[i] + PicHeightInLumaSamples[i]

SubpicWidthInCtbsY[i] = Ceil(SubpicWidthInLumaSamples[i] ÷ CtbSizeY)

SubpicHeightInCtbsY[i] = Ceil(SubpicHeightInLumaSamples[i] ÷ CtbSizeY)

SubpicSizeInCtbsY[i] = SubpicWidthInCtbsY[i] * SubpicHeightInCtbsY[i]

SubpicWidthInMinCbsY[i] = SubpicWidthInLumaSamples[i] / MinCbSizeY

SubpicHeightInMinCbsY[i] = SubpicHeightInLumaSamples[i] / MinCbSizeY

SubpicSizeInMinCbsY[i] = SubpicWidthInMinCbsY[i] * SubpicHeightInMinCbsY[i]

SubpicSizeInSamplesY[i] = SubpicWidthInLumaSamples[i] * SubpicHeightInLumaSamples[i]

SubpicWidthInSamplesC[i] = SubpicWidthInLumaSamples[i] / SubWidthC

SubpicHeightInSamplesC[i] = SubpicHeightInLumaSamples[i] / SubHeightC

La sintaxis y la semántica del conjunto de parámetros RBSP son las siguientes.

sub_pic_parameter_set_rbsp( ) {	Descriptor
spps_subpic_id	u(v)
spps_subpic_parameter_set_id	ue(v)
spps_seq_parameter_set_id	ue(v)
single_tile_in_subpic_flag	u(1)
if(!single_tile_in_subpic_flag) {	
num_tile_columns_minus1	ue(v)
num_tile_rows_minus1	ue(v)
uniform_tile_spacing_flag	u(1)
if(!uniform_tile_spacing_flag) {	
for(i = 0; i < num_tile_columns_minus1; i++)	
tile_column_width_minus1[i]	ue(v)
for(i = 0; i < num_tile_rows_minus1; i++)	
tile_row_height_minus1[i]	ue(v)
}	
loop_filter_across_tiles_enabled_flag	u(1)
}	
if(loop_filter_across_tiles_enabled_flag)	
loop_filter_across_subpic_enabled_flag	u(1)
rbsp_trailing_bits( )	
}	

Spps_subpic_id identifica la subimagen a la que pertenece el SPPS. La longitud de spps_subpic_id es subpic_id_len_minus1 + 1 bits. Spps_subpic_parameter_set_id identifica el SPPS como referencia mediante otros elementos de sintaxis. El valor de spps_subpic_parameter_set_id deberá estar comprendido entre cero y sesenta y tres, ambos inclusive. Spps_seq_parameter_set_id especifica el valor de sps_seq_parameter_set_id para el SPS activo. El valor de spps_seq_parameter_set_id deberá estar comprendido entre cero y quince, ambos inclusive. El single_tile_in_subpic_flag igual a uno especifica que solo hay un mosaico en cada subimagen que se refiere al SPPS. El

single_tile_in_subpic_flag igual a cero especifica que hay más de un mosaico en cada subimagen que se refiere al SPPS. El num_tile_columns_minus1 más 1 especifica el número de columnas de mosaicos que particionan la subimagen. El num_tile_columns_minus1 deberá estar en el rango de cero a PicWidthInCtbsY[spps_subpic_id] - 1, inclusive. Cuando no está presente, se infiere que el valor de num_tile_columns_minus1 es igual a cero. El num_tile_rows_minus1 menos 1 especifica el número de filas de mosaicos que particionan la subimagen. El num_tile_rows_minus1 deberá estar en el rango de cero a PicHeightInCtbsY[spps_subpic_id] - 1, inclusive. Cuando no está presente, se infiere que el valor de num_tile_rows_minus1 es igual a cero. La variable NumTilesInPic se establece igual a (num_tile_columns_minus1 + 1) * (num_tile_rows_minus1 + 1).

Cuando single_tile_in_subpic_flag es igual a cero, NumTilesInPic debe ser mayor que cero. El indicador uniform_tile_spacing_flag igual a uno especifica que los límites de las columnas de mosaicos y, del mismo modo, los límites de las filas de mosaicos se distribuyen uniformemente en la subimagen. El indicador uniform_tile_spacing_flag igual a cero especifica que los límites de las columnas de mosaicos y, del mismo modo, los límites de las filas de mosaicos no se distribuyen uniformemente a través de la subimagen, sino que se señalizan explícitamente utilizando los elementos sintácticos tile_column_width_minus1[i] y tile_row_height_minus1[i]. Cuando no está presente, se infiere que el valor de uniform_tile_spacing_flag es igual a uno. El valor de tile_column_width_minus1[i] más 1 especifica el ancho de la i-ésima columna de mosaicos en unidades de CTBs. El valor de tile_row_height_minus1[i] más 1 especifica la altura de la i-ésima fila de mosaicos en unidades de CTBs.

Las siguientes variables se derivan invocando el proceso de conversión de exploración de trama y mosaico de CTB: la lista ColWidth [i] para i que va de cero a num_tile_columns_minus1, inclusive, especificando el ancho de la i-ésima columna de mosaico en unidades de CTBs; la lista RowHeight[j] para j que va de cero a num_tile_rows_minus1, inclusive, especificando la altura de la j-ésima fila de mosaicos en unidades de CTBs; la lista ColBd[i] para i que va de cero a num_tile_columns_minus1+1, inclusive, especificando la ubicación del i-ésimo límite de columna de mosaico en unidades de CTBs; la lista RowBd [j] para j que va de cero a num_tile_rows_minus1+1, inclusive, especificando la ubicación del j-ésimo límite de fila de mosaicos en unidades de CTBs; la lista CtbAddrRsToTs[ctbAddrRs] para ctbAddrRs que va desde cero hasta PicSizeInCtbsY - 1, inclusive, especificando la conversión de una dirección de CTB en la exploración de trama de CTB de una imagen a una dirección de CTB en la exploración de mosaicos; la lista CtbAddrTsToRs[ctbAddrTs] para ctbAddrTs que va desde cero hasta PicSizeInCtbsY - 1, inclusive, especificando la conversión de una dirección de CTB en la exploración de mosaicos a una dirección de CTB en la exploración de trama de CTB de una imagen; la lista TileId[ctbAddrTs] para ctbAddrTs que va desde cero hasta PicSizeInCtbsY - 1, inclusive, especificando la conversión de una dirección de CTB en la exploración de mosaicos a un ID de mosaico; la lista NumCtusInTile[tileIdx] para tileIdx que va desde cero hasta PicSizeInCtbsY - 1, inclusive, especificando la conversión de un índice de mosaico al número de CTU en el mosaico; la lista FirstCtbAddrTs[tileIdx] para tileIdx que va de cero a NumTilesInPic - 1, inclusive, especificando la conversión de un ID de mosaico a la dirección de CTB en la exploración de mosaico del primer CTB en el mosaico; la lista ColumnWidthInLumaSamples[i] para i que va de cero a num_tile_columns_minus1, inclusive, especificando el ancho de la i-ésima columna de mosaico en unidades de muestras de luma; y la lista RowHeightInLumaSamples[j] para j que va de cero a num_tile_rows_minus1, inclusive, especificando la altura de la j-ésima fila de mosaicos en unidades de muestras de luma. Los valores de ColumnWidthInLumaSamples[i] para i que van de cero a num_tile_columns_minus1, inclusive, y RowHeightInLumaSamples[j] para j que van de cero a num_tile_rows_minus1, inclusive, serán todos mayores que cero.

El loop_filter_across_tiles_enabled_flag igual a uno especifica que las operaciones de filtrado de bucle pueden realizarse a través de los límites de los mosaicos en subimágenes que se refieren al SPPS. El loop_across_tiles_enabled_flag igual a cero especifica que las operaciones de filtrado de bucle no se realizan a través de los límites de los mosaicos en las subimágenes que se refieren al SPPS. Las operaciones de filtrado de bucle incluyen el filtro de desbloqueo, el filtro de desplazamiento adaptativo de muestra y las operaciones de filtro de bucle adaptativo. Cuando no está presente, se infiere que el valor de loop_filter_across_tiles_enabled_flag es igual a uno. El loop_across_subpic_enabled_flag igual a uno especifica que las operaciones de filtrado de bucle pueden realizarse a través de los límites de las subimágenes en las subimágenes que se refieren al SPPS. El loop_filter_across_subpic_enabled_flag igual a cero especifica que las operaciones de filtrado de bucle no se realizan a través de los límites de las subimágenes en las subimágenes que se refieren al SPPS. Las operaciones de filtrado de bucle incluyen el filtro de desbloqueo, el filtro de desplazamiento adaptativo de muestra y las operaciones de filtro de bucle adaptativo. Cuando no está presente, se infiere que el valor de loop_filter_across_subpic_enabled_flag es igual al valor de loop_filter_across_tiles_enabled_flag.

La sintaxis y la semántica general del encabezado del grupo de mosaicos son las siguientes.

tile_group_header( ) {	Descriptor
tile_group_subpic_id	u(v)
tile_group_subpic_parameter_set_id	u(v)
...	
}	

El valor del elemento de sintaxis del encabezado del grupo de mosaicos tile_group_pic_parameter_set_id y tile_group_pic_order_cnt_lsb será el mismo en todos los encabezados del grupo de mosaicos de una imagen codificada. El valor del elemento de sintaxis del encabezado del grupo de mosaicos tile_group_subpic_id será el mismo en todos los encabezados del grupo de mosaicos de una subimagen codificada. El Tile_group_subpic_id identifica la subimagen a la que pertenece el grupo de mosaicos. El length of tile_group_subpic_id es subpic_id_len_minus1 + 1 bits. El tile_group_subpic_parameter_set_id especifica el valor de spps_subpic_parameter_set_id para el SPPS en uso. El valor de tile_group_spps_parameter_set_id estará comprendido entre cero y sesenta y tres, ambos inclusive.

Las siguientes variables se derivan y anulan las respectivas variables derivadas del SPS activo:

PicWidthInLumaSamples = SubpicWidthInLumaSamples[tile_group_subpic_id]

PicHeightInLumaSamples = PicHeightInLumaSamples[tile_group_subpic_id]

SubPicRightBorderInPic = SubPicRightBorderInPic[tile_group_subpic_id]

SubPicBottomBorderInPic = SubPicBottomBorderInPic[tile_group_subpic_id]

PicWidthInCtbsY = SubPicWidthInCtbsY[tile_group_subpic_id]

PicHeightInCtbsY = SubPicHeightInCtbsY[tile_group_subpic_id]

PicSizeInCtbsY = SubPicSizeInCtbsY[tile_group_subpic_id]

PicWidthInMinCbsY = SubPicWidthInMinCbsY[tile_group_subpic_id]

PicHeightInMinCbsY = SubPicHeightInMinCbsY[tile_group_subpic_id]

PicSizeInMinCbsY = SubPicSizeInMinCbsY[tile_group_subpic_id]

PicSizeInSamplesY = SubPicSizeInSamplesY[tile_group_subpic_id]

PicWidthInSamplesC = SubPicWidthInSamplesC[tile_group_subpic_id]

PicHeightInSamplesC = SubPicHeightInSamplesC[tile_group_subpic_id]

La sintaxis de la unidad del árbol de codificación es como sigue.

coding_tree_unit( ) {	Descriptor
xCtb = (CtbAddrInRs % PicWidthInCtbsY) << CtbLog2SizeY + SubpictureXOffset	
yCtb = (CtbAddrInRs / PicWidthInCtbsY) << CtbLog2SizeY + SubpictureYOffset	
...	
}	
dual_tree_implicit_qt_split(x0, y0, log2CbSize, cqtDepth) {	Descriptor
if(log2CbSize > 6) {	
x1 = x0 + (1 << (log2CbSize – 1))	
y1 = y0 + (1 << (log2CbSize – 1))	
dual_tree_implicit_qt_split(x0, y0, log2CbSize – 1, cqtDepth + 1)	
if(x1 < SubPicRightBorderInPic)	
dual_tree_implicit_qt_split(x1, y0, log2CbSize – 1, cqtDepth + 1)	
if(y1 < SubPicBottomBorderInPic)	
dual_tree_implicit_qt_split(x0, y1, log2CbSize – 1, cqtDepth + 1)	
if(x1 < SubPicRightBorderInPic && y1 < SubPicBottomBorderInPic)	
dual_tree_implicit_qt_split(x1, y1, log2CbSize – 1, cqtDepth + 1)	
} else {	
coding_quadtree(x0, y0, log2CbSize, cqtDepth, DUAL_TREE_LUMA)	
coding_quadtree(x0, y0, log2CbSize, cqtDepth, DUAL_TREE_CHROMA)	

coding_tree_unit( ) {	Descriptor
}	
}	

La sintaxis de codificación de árbol cuádruple y la semántica son las siguientes.

coding_quadtree(x0, y0, log2CbSize, cqtDepth, treeType) {	Descriptor
minQtSize = (treeType = = DUAL_TREE_CHROMA) ? MinQtSizeC : MinQtSizeY	
maxBtSize = (treeType = = DUAL_TREE_CHROMA) ? MaxBtSizeC : MaxBtSizeY	
if(((x0 + (1 << log2CbSize) <= PicWidthInLumaSamples) ? 1 : 0) + ((y0 + (1 << log2CbSize) <= PicHeightInLumaSamples) ? 1 : 0) + (((1 << log2CbSize) <= maxBtSize) ? 1 : 0)) >= 2 && (1 << log2CbSize) > minQtSize)	
qt_split_cu_flag[x0][y0]	ae(v)
if(cu_qp_delta_enabled_flag && cqtDepth <= diff_cu_qp_delta_depth) {	
IsCuQpDeltaCoded = 0	
CuQpDeltaVal = 0	
CuQgTopLeftX = x0	
CuQgTopLeftY = y0	
}	
if(qt_split_cu_flag[x0][y0]) {	
x1 = x0 + (1 << (log2CbSize – 1))	
y1 = y0 + (1 << (log2CbSize – 1))	
coding_quadtree(x0, y0, log2CbSize – 1, cqtDepth + 1, treeType)	
if(x1 < SubPicRightBorderInPic)	
coding_quadtree(x1, y0, log2CbSize – 1, cqtDepth + 1, treeType)	
if(y1 < SubPicBottomBorderInPic)	
coding_quadtree(x0, y1, log2CbSize – 1, cqtDepth + 1, treeType)	
if(x1 < SubPicRightBorderInPic && y1 < SubPicBottomBorderInPic)	
coding_quadtree(x1, y1, log2CbSize – 1, cqtDepth + 1, treeType)	
} else	
multi_type_tree(x0, y0, 1 << log2CbSize, 1 << log2CbSize, cqtDepth, 0, 0, 0, treeType)	
}	

Qt_split_cu_flag [x0] [y0] especifica si una unidad de codificación se divide en unidades de codificación con un tamaño medio horizontal y vertical. Los índices de arreglo x0, y0 especifican la ubicación (x0, y0) de la muestra de luma superior izquierda del bloque de codificación considerado en relación con la muestra de luma superior izquierda de la imagen. Cuando qt_split_cu_flag [x0] [y0] no está presente, se aplica lo siguiente: Si una o más de las siguientes condiciones son verdaderas, se infiere que el valor de qt_split_cu_flag [x0] [y0] es igual a uno. x0 + (1 << log2CbSize) es mayor que SubPicRightBorderInPic y (1 << log2CbSize) es mayor que MaxBtSizeC si treeType es igual a DUAL_TREE_CHROMA o mayor que MaxBtSizeY en caso contrario. y0 + (1 << log2CbSize) es mayor que SubPicBottomBorderInPic y (1 << log2CbSize) es mayor que MaxBtSizeC si treeType es igual a DUAL_TREE_CHROMA o mayor que MaxBtSizeY en caso contrario.

De lo contrario, si se cumplen todas las condiciones siguientes, se infiere que el valor de qt_split_cu_flag [x0] [y0] es igual a 1: x0 + (1 << log2CbSize) es mayor que SubPicRightBorderInPic, y0 + (1 << log2CbSize) es mayor que SubPicBottomBorderInPic, y (1 << log2CbSize) es mayor que MinQtSizeC si treeType es igual a DUAL_TREE_CHROMA o mayor que MinQtSizeY en caso contrario. En caso contrario, se infiere que el valor de qt_split_cu_flag[x0][y0] es igual a cero.

La sintaxis y la semántica del árbol de múltiples tipos son las siguientes.

multi_type_tree(x0, y0, cbWidth, cbHeight, cqtDepth, mttDepth, depthOffset, partIdx, treeType) {	Descriptor
if((allowSplitBtVer || allowSplitBtHor || allowSplitTtVer || allowSplitTtHor) && (x0 + cbWidth <= SubPicRightBorderInPic) && (y0 + cbHeight <= SubPicBottomBorderInPic))	
mtt_split_cu_flag	ae(v)
if(cu_qp_delta_enabled_flag && (cqtDepth + mttDepth) <= diff_cu_qp_delta_depth) {	

multi_type_tree(x0, y0, cbWidth, cbHeight, cqtDepth, mttDepth, depthOffset, partIdx, treeType) {	Descriptor
IsCuQpDeltaCoded = 0	
CuQpDeltaVal = 0	
CuQgTopLeftX = x0	
CuQgTopLeftY = y0	
}	
if(mtt_split_cu_flag) {	
if((allowSplitBtHor || allowSplitTtHor) && (allowSplitBtVer || allowSplitTtVer))	
mtt_split_cu_vertical_flag	ae(v)
if((allowSplitBtVer && allowSplitTtVer && mtt_split_cu_vertical_flag) || (allowSplitBtHor && allowSplitTtHor && !mtt_split_cu_vertical_flag))	
mtt_split_cu_binary_flag	ae(v)
if(MttSplitMode[x0][y0][mttDepth] == SPLIT_BT_VER) {	
depthOffset += (x0 + cbWidth > SubPicRightBorderInPic) ? 1 : 0	
x1 = x0 + (cbWidth / 2)	
multi_type_tree(x0, y0, cbWidth / 2, cbHeight, cqtDepth, mttDepth + 1, depthOffset, 0, treeType)	
if(x1 < SubPicRightBorderInPic)	
multi_type_tree(x1, y0, cbWidth / 2, cbHeightY, cqtDepth, mttDepth + 1, depthOffset, 1, treeType)	
} else if(MttSplitMode[x0][y0][mttDepth] == SPLIT_BT_HOR) {	
depthOffset += (y0 + cbHeight > SubPicBottomBorderInPic) ? 1 : 0	
y1 = y0 + (cbHeight / 2)	
multi_type_tree(x0, y0, cbWidth, cbHeight / 2, cqtDepth, mttDepth + 1, depthOffset, 0, treeType)	
if(y1 < SubPicBottomBorderInPic)	
multi_type_tree(x0, y1, cbWidth, cbHeight / 2, cqtDepth, mttDepth + 1, depthOffset, 1, treeType)	
} else if(MttSplitMode[x0][y0][mttDepth] == SPLIT_TT_VER) {	
x1 = x0 + (cbWidth / 4)	
x2 = x0 + (3 * cbWidth / 4)	
multi_type_tree(x0, y0, cbWidth / 4, cbHeight, cqtDepth, mttDepth + 1, depthOffset, 0, treeType)	
multi_type_tree(x1, y0, cbWidth / 2, cbHeight, cqtDepth, mttDepth + 1, depthOffset, 1, treeType)	
multi_type_tree(x2, y0, cbWidth / 4, cbHeight, cqtDepth, mttDepth + 1, depthOffset, 2, treeType)	
} else { /* SPLIT_TT_HOR */	
y1 = y0 + (cbHeight / 4)	
y2 = y0 + (3 * cbHeight / 4)	
multi_type_tree(x0, y0, cbWidth, cbHeight / 4, cqtDepth, mttDepth + 1, depthOffset, 0, treeType)	
multi_type_tree(x0, y1, cbWidth, cbHeight / 2, cqtDepth, mttDepth + 1, depthOffset, 1, treeType)	
multi_type_tree(x0, y2, cbWidth, cbHeight / 4, cqtDepth, mttDepth + 1, depthOffset, 2 , treeType)	
}	
} else	
coding_unit(x0, y0, cbWidth, cbHeight, treeType)	
}	

El mtt_split_cu_flag igual a cero especifica que una unidad de codificación no se divide. El mtt_split_cu_flag igual a uno especifica que una unidad de codificación se divide en dos unidades de codificación usando una división binaria o en tres unidades de codificación usando una división ternaria como lo indica el elemento de sintaxis mtt_split_cu_binary_flag. La división binaria o ternaria puede ser vertical u horizontal, como lo indica el elemento de sintaxis mtt_split_cu_vertical_flag. Cuando mtt_split_cu_flag no está presente, el valor de mtt_split_cu_flag se infiere de la siguiente manera. Si se cumple una o más de las siguientes condiciones, se infiere que el valor de mtt_split_cu_flag

es igual a 1: x0 + cbWidth es mayor que SubPicRightBorderInPic y y0 + cbHeight es mayor que SubPicBottomBorderInPic. En caso contrario, se infiere que el valor de mtt_split_cu_flag es igual a cero.

El proceso de derivación para la predicción de vectores de movimiento de luma temporal es como sigue. Los resultados de este proceso son: la predicción del vector de movimiento mvLXCol en una precisión de muestra fraccionaria de 1/16 y la bandera de disponibilidad availableFlagLXCol. La variable currCb especifica el bloque de codificación de luma actual en la ubicación de luma (xCb, yCb). Las variables mvLXCol y availableFlagLXCol se derivan de la siguiente manera. Si tile_group_temporal_mvp_enabled_flag es igual a cero, o si la imagen de referencia es la imagen actual, ambos componentes de mvLXCol se establecen en cero y availableFlagLXCol se establece en cero. De lo contrario (tile_group_temporal_mvp_enabled_flag es igual a uno y la imagen de referencia no es la imagen actual), se aplican los siguientes pasos ordenados. El vector de movimiento ubicado en la parte inferior derecha se deriva de la siguiente manera:

xColBr = xCb + cbWidth (8-355)

yColBr = yCb + cbHeight (8-356)

Si yCb >> CtbLog2SizeY es igual a yColBr >> CtbLog2SizeY, yColBr es menor que SubPicBottomBorderInPic y xColBr es menor que SubPicRightBorderInPic, se aplica lo siguiente. La variable colCb especifica el bloque de codificación de luma que cubre la ubicación modificada dada por ((xColBr >> 3) << 3, (yColBr >> 3) << 3) dentro de la imagen colocada especificada por ColPic. La ubicación de luma (xColCb, yColCb) se establece igual a la muestra superior izquierda del bloque de codificación de luma co-localizado/colocados especificado por colCb en relación con la muestra de luma superior izquierda de la imagen co-localizada especificada por ColPic. El proceso de derivación para vectores de movimiento co-localizados se invoca con currCb, colCb, (xColCb, yColCb), refIdxLX y sbFlag establecidos en cero como entradas, y la salida se asigna a mvLXCol y availableFlagLXCol. De lo contrario, ambos componentes de mvLXCol se establecen en cero y availableFlagLXCol se establece en cero.

El proceso de derivación para candidatos de fusión de triángulo temporal es la siguiente. Las variables mvLXColC0, mvLXColC1, availableFlagLXColC0 y availableFlagLXColC1 se derivan de la siguiente manera. Si tile_group_temporal_mvp_enabled_flag es igual a cero, ambos componentes de mvLXColC0 y mvLXColC1 se establecen en cero y availableFlagLXColC0 y availableFlagLXColC1 se establecen en cero. De lo contrario (tile_group_temporal_mvp_enabled_flag es igual a 1), se aplican los siguientes pasos ordenados. El vector de movimiento ubicado abajo a la derecha mvLXColC0 se deriva de la siguiente manera:

xColBr = xCb + cbWidth (8-392)

yColBr = yCb + cbHeight (8-393)

Si yCb >> CtbLog2SizeY es igual a yColBr >> CtbLog2SizeY, yColBr es menor que SubPicBottomBorderInPic y xColBr es menor que SubPicRightBorderInPic, se aplica lo siguiente. La variable colCb especifica el bloque de codificación de luma que cubre la ubicación modificada dada por ((xColBr >> 3) << 3, (yColBr >> 3) << 3) dentro de la imagen colocada especificada por ColPic. La ubicación de luma (xColCb, yColCb) se establece igual a la muestra superior izquierda del bloque de codificación de luma co-localizado/colocados especificado por colCb en relación con la muestra de luma superior izquierda de la imagen co-localizada especificada por ColPic. El proceso de derivación para vectores de movimiento co-localizados se invoca con currCb, colCb, (xColCb, yColCb), refIdxLXC0 y sbFlag establecidos en cero como entradas, y la salida se asigna a mvLXColC0 y availableFlagLXColC0. De lo contrario, ambos componentes de mvLXColC0 se establecen en cero y availableFlagLXColC0 se establece en cero.

El proceso de derivación para candidatos de fusión de vector de movimiento de punto de control afín construido es como sigue. El cuarto vector de movimiento del punto de control (ubicado en la parte inferior derecha) cpMvLXCorner [3], el índice de referencia refIdxLXCorner [3], la bandera de utilización de la lista de predicción predFlagLXCorner [3] y la bandera de disponibilidad availableFlagCorner [3] con X siendo 0 y 1 se derivan de la siguiente manera . Los índices de referencia para el candidato de fusión temporal, refIdxLXCorner [3], con X siendo cero o uno, se establecen en cero. Las variables mvLXCol y availableFlagLXCol, con X siendo cero o uno, se derivan de la siguiente manera. Si tile_group_temporal_mvp_enabled_flag es igual a cero, ambos componentes de mvLXCol se establecen en cero y availableFlagLXCol se establece en cero. De lo contrario (tile_group_temporal_mvp_enabled_flag es igual a uno), se aplica lo siguiente:

xColBr = xCb + cbWidth (8-566)

yColBr = yCb + cbHeight (8-567)

Si yCb >> CtbLog2SizeY es igual a yColBr >> CtbLog2SizeY, yColBr es menor que SubPicBottomBorderInPic y xColBr es menor que SubPicRightBorderInPic, se aplica lo siguiente. La variable colCb especifica el bloque de codificación de luma que cubre la ubicación modificada dada por ((xColBr >> 3) << 3, (yColBr >> 3) << 3) dentro de la imagen colocada especificada por ColPic. La ubicación de luma (xColCb, yColCb) se establece igual a la muestra

superior izquierda del bloque de codificación de luma co-localizado/colocados especificado por colCb en relación con la muestra de luma superior izquierda de la imagen co-localizada especificada por ColPic. El proceso de derivación para vectores de movimiento co-localizados se invoca con currCb, colCb, (xColCb, yColCb), refIdxLX y sbFlag establecidos en cero como entradas, y la salida se asigna a mvLXCol y availableFlagLXCol. De lo contrario, ambos componentes de mvLXCol se establecen en 0 y availableFlagLXCol se establece en cero. Reemplace todas las apariciones de pic_width_in_luma_samples con PicWidthInLumaSamples. Reemplace todas las apariciones de pic_height_in_luma_samples con PicHeightInLumaSamples.

En un segundo ejemplo de modalidad, la sintaxis y la semántica de RBSP del conjunto de parámetros de secuencia son las siguientes.

seq_parameter_set_rbsp( ) {	Descriptor
sps_seq_parameter_set_id	ue(v)
pic_width_in_luma_samples	ue(v)
pic_height_in_luma_samples	ue(v)
num_subpic_minus1	ue(v)
subpic_id_len_minus1	ue(v)
for (i = 0; i <= num_subpic_minus1; i++) {	
subpic_id[i]	u(v)
if(num_subpic_minus1 > 0) {	
subpic_level_idc[i]	u(8)
subpic_x_offset[i]	ue(v)
subpic_y_offset[i]	ue(v)
subpic_width_in_luma_samples[i]	ue(v)
subpic_height_in_luma_samples[i]	ue(v)
}	
}	
...	
}	

El subpic_id_len_minus1 más uno especifica el número de bits usados para representar el elemento de sintaxis subpic_id [i] en SPS, spps_subpic_id en SPPS refiriéndose al SPS, y tile_group_subpic_id en encabezados de grupo de mosaicos referidos al SPS. El valor de subpic_id_len_minus1 deberá estar en el rango de Ceil(Log2(num_subpic_minus1 + 3) a ocho, inclusive. Es un requisito de la conformidad del flujo de bits que no haya superposición entre la subimagen[i] para i de 0 a num_subpic_minus1, inclusive. Cada subimagen puede ser una subimagen restringida de movimiento temporal.

La semántica general de la cabecera del grupo de mosaicos es la siguiente. El Tile_group_subpic_id identifica la subimagen a la que pertenece el grupo de mosaicos. El length of tile_group_subpic_id es subpic_id_len_minus1 + 1 bits. El tile_group_subpic_id igual a uno indica que el grupo de mosaicos no pertenece a ninguna subimagen.

En un tercer ejemplo de modalidad, la sintaxis de encabezado de unidad NAL y la semántica son las siguientes.

nal_unit_header( ) {	Descriptor
forbidden_zero_bit	f(1)
nal_unit_type	u(5)
nuh_temporal_id_plus1	u(3)
nuh_subpicture_id_len	u(4)
nuh_reserved_zero_4bits	u(3)
}	

El Nuh_subpicture_id_len especifica el número de bits utilizados para representar el elemento de sintaxis que especifica el ID de subimagen. Cuando el valor de nuh_subpicture_id_len es mayor que cero, los primeros nuh_subpicture_id_len-th bits después de nuh_reserved_zero_4bits especifican el ID de la subimagen a la que pertenece la carga útil de la unidad NAL. Cuando nuh_subpicture_id_len es mayor que cero, el valor de nuh_subpicture_id_len será igual al valor de subpic_id_len_minus1 en el SPS activo. El valor de nuh_subpicture_id_len para unidades NAL no VCL está restringido de la siguiente manera. Si nal_unit_type es igual a SPS_NUT o PPS_NUT, nuh_subpicture_id_len será igual a cero. Nuh_reserved_zero_3bits será igual a '000'. Los decodificadores ignorarán (por ejemplo, eliminarán del flujo de bits y descartarán) las unidades NAL con valores de nuh_reserved_zero_3bits no iguales a '000'.

En un cuarto ejemplo de modalidad, la sintaxis de anidamiento de subimágenes es la siguiente.

sub-picture_nesting(payloadSize) {	Descriptor
all_sub_pictures_flag	u(1)
if(!all_sub_pictures_flag) {	
nesting_num_sub_pictures_minus1	ue(v)
for(i = 0; i <= nesting_num_sub_pictures_minus1; i++)	
nesting_sub_picture_id[i]	u(v)
}	
while(!byte_aligned( ))	
sub_picture_nesting_zero_bit /* igual a 0 */	u(1)
do	
sei_message( )	
while(more_rbsp_data( ))	
}	

El all_sub_pictures_flag igual a uno especifica que los mensajes SEI anidados se aplican a todas las subimágenes. all_sub_pictures_flag igual a uno especifica que las subimágenes a las que se aplican los mensajes SEI anidados son señalizadas explícitamente por los elementos de sintaxis subsiguientes. El nesting_num_sub_pictures_minus1 más 1 especifica el número de subimágenes a las que se aplican los mensajes SEI anidados. El nesting_sub_picture_id[i] indica el ID de subimagen de la i-ésima subimagen a la que se aplican los mensajes SEI anidados. El elemento de sintaxis nesting_sub_picture_id[i] está representado por Ceil (Log2 (nesting_num_sub_pictures_minus1 + 1)) bits. El sub_picture_nesting_zero_bit será igual a cero.

La FIG. 9 es un diagrama esquemático de un dispositivo de codificación de vídeo de ejemplo 900. El dispositivo de codificación de video 900 es adecuado para implementar las modalidades/ejemplos descritos en este documento. El dispositivo de codificación de vídeo 900 comprende puertos descendentes 920, puertos ascendentes 950 y/o unidades transceptoras (Tx / Rx) 910, que incluyen transmisores y/o receptores para comunicar datos ascendentes y/o descendentes a través de una red. El dispositivo de codificación de video 900 también incluye un procesador 930 que incluye una unidad lógica y/o unidad central de procesamiento (CPU) para procesar los datos y una memoria 932 para almacenar los datos. El dispositivo de codificación de video 900 también puede comprender componentes eléctricos, ópticos a eléctricos (OE), componentes eléctricos a ópticos (EO) y/o componentes de comunicación inalámbrica acoplados a los puertos ascendentes 950 y/o descendentes 920 para la comunicación de datos a través de redes de comunicación eléctricas, ópticas o inalámbricas. El dispositivo de codificación de vídeo 900 también puede incluir dispositivos de entrada y/o salida (E/S / I/O) 960 para comunicar datos hacia y desde un usuario. Los dispositivos de E/S 960 pueden incluir dispositivos de salida tales como una pantalla para mostrar datos de video, altavoces para emitir datos de audio, etc. Los dispositivos de E/S 960 también pueden incluir dispositivos de entrada, tales como un teclado, mouse, trackball, etc. y/o interfaces correspondientes para interactuar con dichos dispositivos de salida.

El procesador 930 se implementa mediante hardware y software. El procesador 930 puede implementarse como uno o más chips de CPU, núcleos (por ejemplo, como un procesador de múltiples núcleos), matrices de puertas programables en campo (FPGA), circuitos integrados específicos de aplicación (ASIC) y procesadores de señales digitales (DSP). El procesador 930 está en comunicación con los puertos descendentes 920, Tx/Rx 910, los puertos ascendentes 950 y la memoria 932. El procesador 930 comprende un módulo de codificación 914. El módulo de codificación 914 implementa las modalidades descritas anteriormente, como los métodos 100, 1000, 1100, y/o el mecanismo 700, que pueden emplear un flujo de bits 500, una imagen 600, y/o una imagen 800. El módulo de codificación 914 también puede implementar cualquier otro método/mecanismo descrito en este documento. Además, el módulo de codificación 914 puede implementar un sistema de códec 200, un codificador 300 y/o un decodificador 400. Por ejemplo, el módulo de codificación 914 puede emplearse para señalizar y/u obtener ubicaciones y tamaños de subimagen en un SPS. En otro ejemplo, el módulo de codificación 914 puede restringir los anchos de las subimágenes y las alturas de las subimágenes para que sean múltiplos del tamaño de la CTU, a menos que dichas subimágenes se coloquen en el borde derecho de la imagen o en el borde inferior de la imagen, respectivamente. En otro ejemplo, el módulo de codificación 914 puede restringir las subimágenes para cubrir una imagen sin espacios ni superposiciones. En otro ejemplo, el módulo de codificación 914 puede emplearse para señalizar y/u obtener datos que indiquen que algunas subimágenes son subimágenes restringidas de movimiento temporal y otras subimágenes no. En otro ejemplo, el módulo de codificación 914 puede señalizar un conjunto completo de IDs de subimagen en el SPS e incluir un ID de subimagen en cada encabezado de segmento para indicar la subimagen que contiene los segmentos correspondientes. En otro ejemplo, el módulo de codificación 914 puede señalizar niveles para cada subimagen. Como tal, el módulo de codificación 914 hace que el dispositivo de codificación de video 900 proporcione funcionalidad adicional, evite cierto procesamiento para reducir la sobrecarga de procesamiento y/o aumente la eficiencia de codificación al particionar y codificar datos de video. Acordemente, el módulo de codificación 914 mejora la funcionalidad del dispositivo de codificación de video 900 así como también resuelve problemas que son específicos de las artes de codificación de video.

Además, el módulo de codificación 914 efectúa una transformación del dispositivo de codificación de vídeo 900 a un estado diferente. Alternativamente, el módulo de codificación 914 puede implementarse como instrucciones almacenadas en la memoria 932 y ejecutadas por el procesador 930 (por ejemplo, como un producto de programa de computadora almacenado en un medio no transitorio).

La memoria 932 comprende uno o más tipos de memoria, como discos, unidades de cinta, unidades de estado sólido, memoria de solo lectura (ROM), memoria de acceso aleatorio (RAM), memoria flash, memoria ternaria direccionable por contenido (TCAM), memoria aleatoria estática. memoria de acceso (SRAM), etc. La memoria 932 puede usarse como un dispositivo de almacenamiento de datos de desbordamiento, para almacenar programas cuando dichos programas se seleccionan para su ejecución, y para almacenar instrucciones y datos que se leen durante la ejecución del programa.

La FIG. 10 es un diagrama de flujo de un método de ejemplo 1000 de codificación de un diseño de subimagen en un flujo de bits, como el flujo de bits 500, de imágenes para soportar la extracción de subimágenes, como las subimágenes 522, 523, 622, 722 y/o 822. El método 1000 puede ser empleado por un codificador, como un sistema de códec 200, un codificador 300, y/o un dispositivo de codificación de vídeo 900 al realizar el método 100.

El método 1000 puede comenzar cuando un codificador recibe una secuencia de video que incluye una pluralidad de imágenes y determina codificar esa secuencia de video en un flujo de bits, por ejemplo en base a la entrada del usuario. La secuencia de vídeo se particiona en fotografías/imágenes/fotogramas para su posterior división antes de la codificación. En el paso 1001, una imagen se particiona en una pluralidad de subimágenes que incluyen una subimagen actual indicada en lo sucesivo como la subimagen. La subimagen se codifica en un flujo de bits en el paso 1003.

En el paso 1005, un tamaño de subimagen y una ubicación de subimagen de la subimagen se codifican en un SPS en el flujo de bits. La ubicación de subimagen incluye una distancia de desplazamiento entre una muestra superior izquierda de la subimagen y una muestra superior izquierda de la imagen. El tamaño de subimagen incluye una altura de subimagen en las muestras de luma y un ancho de subimagen en las muestras de luma. También se puede codificar una bandera en el SPS para indicar que la subimagen es una subimagen restringida de movimiento. En tal caso, el tamaño de subimagen y la ubicación de subimagen indican un diseño de la subimagen restringida de movimiento.

En el paso 1007, las IDs de subimágenes se codifican en el SPS para cada una de las subimágenes divididas de la imagen. Varias de las subimágenes divididas de la imagen también pueden codificarse en el SPS. En el paso 1009, el flujo de bits se almacena para su comunicación hacia un decodificador. El flujo de bits puede entonces ser transmitido hacia el decodificador como se desee. En algunos ejemplos, se puede extraer un subflujo de bits del flujo de bits codificado. En tal caso, el flujo de bits transmitido es un subflujo de bits. En otros ejemplos, el flujo de bits codificado puede transmitirse para la extracción del subflujo de bits en el decodificador. En otros ejemplos más, el flujo de bits codificado se puede decodificar y visualizar sin extracción de subflujo de bits. En cualquiera de estos ejemplos, el tamaño, la ubicación, el ID, el número de subimagen y/o la bandera de subimagen restringida de movimiento se pueden utilizar para señalizar eficazmente el diseño de subimagen a un decodificador.

La FIG. 11 es un diagrama de flujo de un método de ejemplo 1100 de decodificación de un flujo de bits, como el flujo de bits 500 y/o el subflujo de bits 501, de subimágenes, como las subimágenes 522, 523, 622, 722 y/o 822, basadas en un diseño de subimagen señalizado. El método 1100 puede ser empleado por un decodificador, como un sistema de códec 200, un decodificador 400, y/o un dispositivo de codificación de vídeo 900 al realizar el método 100. Por ejemplo, el método 1100 se puede aplicar para decodificar un flujo de bits creado como resultado del método 1000.

El método 1100 puede comenzar cuando un decodificador comienza a recibir un flujo de bits que contiene subimágenes. El flujo de bits puede incluir una secuencia de vídeo completa o el flujo de bits puede ser un subflujo de bits que contiene un conjunto reducido de subimágenes para extracción por separado. En el paso 1101, se recibe un flujo de bits. El flujo de bits comprende una subimagen dividida en una imagen. El flujo de bits también comprende un SPS. El SPS comprende un tamaño de imagen secundaria y una ubicación de imagen secundaria. En algunos ejemplos, la subimagen es una subimagen restringida de movimiento temporal. En tales casos, el tamaño de subimagen y la ubicación de subimagen indican un diseño de subimagen con limitación de movimiento. En algunos ejemplos, el SPS puede comprender además IDs de subimagen para cada subimagen dividida en la imagen.

En el paso 1103, se analiza el SPS para obtener el tamaño de subimagen y la ubicación de subimagen. El tamaño de subimagen puede incluir una altura de subimagen en muestras de luma y un ancho de subimagen en muestras de luma. La ubicación de subimagen puede incluir una distancia de desplazamiento entre una muestra superior izquierda de la subimagen y una muestra superior izquierda de la imagen. La subimagen también se puede analizar para obtener otros datos relacionados con la subimagen, tales como un indicador de subimagen restringida de movimiento temporal y/o IDs de subimagen.

En el paso 1105, se puede determinar el tamaño de subimagen en relación con el tamaño de una pantalla basándose en el tamaño de subimagen. Además, se puede determinar una posición de la subimagen con relación a la pantalla basándose en la ubicación de subimagen. El decodificador también puede determinar si la subimagen se puede

decodificar independientemente en base a la bandera de subimagen restringida de movimiento temporal. Por lo tanto, el decodificador puede determinar el diseño de la subimagen basándose en los datos analizados del SPS y/o los datos correspondientes de los encabezados de segmento asociados con los segmentos contenidos en la subimagen.

En el paso 1107, la subimagen se decodifica basándose en el tamaño de subimagen, la ubicación de subimagen y/u otra información obtenida del SPS, un PPS, encabezado(s) de segmento, mensaje(s) SEI, etc. La subimagen se decodifica para crear una secuencia de video. La secuencia de vídeo se puede reenviar luego para su visualización en el paso 1109.

La FIG. 12 es un diagrama esquemático de un sistema de ejemplo 1200 para señalizar un diseño de subimagen, como un diseño para subimágenes 522, 523, 622, 722 y/o 822, a través de un flujo de bits, como el flujo de bits 500 y/o subflujo de bits 501. El sistema 1200 puede implementarse mediante un codificador y un decodificador, tal como un sistema de códec 200, un codificador 300, un decodificador 400 y / o un dispositivo de codificación de vídeo 900. Además, el sistema 1200 puede emplearse al implementar el método 100, 1000 y/o 1100.

El sistema 1200 incluye un codificador de video 1202. El codificador de video 1202 comprende un módulo de partición 1201 para particionar una imagen en una pluralidad de subimágenes que incluyen una subimagen actual. El codificador de video 1202 comprende además un módulo de codificación 1203 para codificar la subimagen, particionada a partir de la imagen, en un flujo de bits, y codificar un tamaño de subimagen y una ubicación de subimagen de la subimagen en un SPS en el flujo de bits. El codificador de video 1202 comprende además un módulo de almacenamiento 1205 para almacenar el flujo de bits para la comunicación hacia un decodificador. El codificador de video 1202 comprende además un módulo de transmisión 1207 para transmitir el flujo de bits que incluye la subimagen, el tamaño de subimagen y la ubicación de subimagen hacia el decodificador. El codificador de video 1202 puede configurarse además para realizar cualquiera de los pasos del método 1000.

El sistema 1200 también incluye un decodificador de video 1210. El decodificador de video 1210 comprende un módulo de recepción 1211 para recibir un flujo de bits que comprende una subimagen particionada a partir de una imagen y un SPS que comprende un tamaño de subimagen de la subimagen y una ubicación de subimagen de la subimagen. El decodificador de vídeo 1210 comprende además un módulo de análisis 1213 para analizar el SPS para obtener el tamaño de subimagen y la ubicación de subimagen. El decodificador de video 1210 comprende además un módulo de decodificación 1215 para decodificar la subimagen en base al tamaño de subimagen y la ubicación de subimagen para crear una secuencia de video. El decodificador de vídeo 1110 comprende además un módulo de reenvío 1217 para reenviar la secuencia de vídeo para su visualización. El decodificador de vídeo 1210 puede configurarse además para realizar cualquiera de los pasos del método 1100.

Un primer componente se acopla directamente a un segundo componente cuando no hay componentes intermedios, excepto por una línea, una traza u otro medio entre el primer componente y el segundo componente. El primer componente se acopla indirectamente al segundo componente cuando hay componentes intermedios distintos de una línea, una traza u otro medio entre el primer componente y el segundo componente. El término "acoplado" y sus variantes incluyen tanto acoplados directamente como acoplados indirectamente. El uso del término "aproximadamente" significa un rango que incluye ± 10% del número siguiente, a menos que se indique lo contrario.

También debe entenderse que los pasos de los métodos ejemplares expuestos en este documento no se requiere necesariamente que se realicen en el orden descrito, y el orden de los pasos de dichos métodos debe entenderse como meramente ejemplar. Asimismo, se pueden incluir pasos adicionales en tales métodos, y se pueden omitir o combinar ciertos pasos, en métodos consistentes con diversas modalidades de la presente divulgación.

Si bien se han proporcionado varias modalidades en la presente divulgación, se puede entender que los sistemas y métodos descritos se pueden realizar en muchas otras formas específicas sin apartarse del espíritu o alcance de la presente divulgación. Los presentes ejemplos deben considerarse ilustrativos y no restrictivos, y la intención no se limita a los detalles que se dan en el presente documento. Por ejemplo, los distintos elementos o componentes pueden combinarse o integrarse en otro sistema o pueden omitirse ciertas características o no implementarse.

Además, las técnicas, sistemas, subsistemas y métodos descritos e ilustrados en las diversas modalidades como discretos o separados pueden combinarse o integrarse con otros sistemas, componentes, técnicas o métodos sin apartarse del alcance de la presente divulgación. Un experto en la técnica puede comprobar otros ejemplos de cambios, sustituciones y alteraciones y pueden realizarse sin apartarse del espíritu y alcance descritos en este documento.

REIVINDICACIONES

1. Un método implementado en un decodificador, el método caracterizado porque comprende:

recibir (1101), por un receptor del decodificador, un flujo de bits que comprende subimágenes particionadas de una imagen, y un conjunto de parámetros de secuencia (SPS) que comprende parámetros, en donde los parámetros comprenden un tamaño de subimagen para cada una de las subimágenes y una ubicación de subimagen para cada una de las subimágenes, en donde el SPS comprende además un identificador (ID) de subimagen para cada una de las subimágenes;

analizar (1103), mediante el procesador, el SPS para obtener el ID de subimagen, el tamaño de subimagen y la ubicación de subimagen para cada una de las subimágenes;

generar (1107), mediante el procesador, las subimágenes en base al ID de subimagen, el tamaño de subimagen y la ubicación de subimagen para cada una de las subimágenes para crear una secuencia de vídeo.

2. El método de la reivindicación 1, caracterizado porque la ubicación de subimagen incluye una distancia de desplazamiento entre una muestra superior izquierda de una correspondiente subimagen y una muestra superior izquierda de la imagen.

3. El método de la reivindicación 1 o 2, caracterizado porque el tamaño de subimagen incluye una altura de subimagen en las muestras de luma y un ancho de subimagen en las muestras de luma.

4. Un método implementado en un codificador, el método caracterizado porque comprende:

codificar (1003), mediante un procesador del codificador, subimágenes, particionadas a partir de una imagen, en un flujo de bits;

codificar (1005), por el procesador, un identificador (ID) de subimagen, un tamaño de subimagen y una ubicación de subimagen para cada una de las subimágenes en un conjunto de parámetros de secuencia (SPS) en el flujo de bits.

5. El método de la reivindicación 4, el método caracterizado porque comprende, además:

almacenar (1009), en una memoria del codificador, el flujo de bits para la comunicación hacia un decodificador.

6. Un decodificador (1210), caracterizado porque comprende:

una unidad de recepción (1211), configurada para recibir un flujo de bits que comprende subimágenes particionadas de una imagen, y un conjunto de parámetros de secuencia (SPS) que comprende parámetros, en donde los parámetros comprenden un tamaño de subimagen para cada una de las subimágenes y una ubicación de subimagen para cada una de las subimágenes, en donde el SPS comprende además un identificador (ID) de subimagen para cada una de las subimágenes;

una unidad de análisis (1213), configurada para analizar el SPS para obtener el ID de subimagen, el tamaño de subimagen y la ubicación de subimagen para cada una de las subimágenes;

una unidad de generación (1215), configurada para generar las subimágenes basadas en el ID de subimagen, el tamaño de subimagen y la ubicación de subimagen para cada una de las subimágenes para crear una secuencia de vídeo.

7. Un codificador (1202), caracterizado porque comprende:

una unidad de codificación (1203), configurada para:

codificar subimágenes, particionadas a partir de una imagen, en un flujo de bits; y

codificar un identificador (ID) de subimagen, un tamaño de subimagen y una ubicación de subimagen para cada una de las subimágenes en un conjunto de parámetros de secuencia (SPS) en el flujo de bits.

8. El codificador (1202) de la reivindicación 7, el codificador caracterizado porque comprende, además:

una unidad de almacenamiento (1205), configurada para almacenar el flujo de bits para la comunicación hacia un decodificador.

9. Un flujo de bits, el flujo de bits que comprende subimágenes particionadas de una imagen actual y un conjunto de parámetros de secuencia (SPS) que comprende parámetros, en donde los parámetros comprenden un identificador (ID) de subimagen, un tamaño de subimagen y una ubicación de subimagen para cada una de las subimágenes.

10. Un producto de programa de computadora que comprende un código de programa para realizar el método de acuerdo con cualquiera de las reivindicaciones 1 a 5 cuando se ejecuta en una computadora o un procesador.

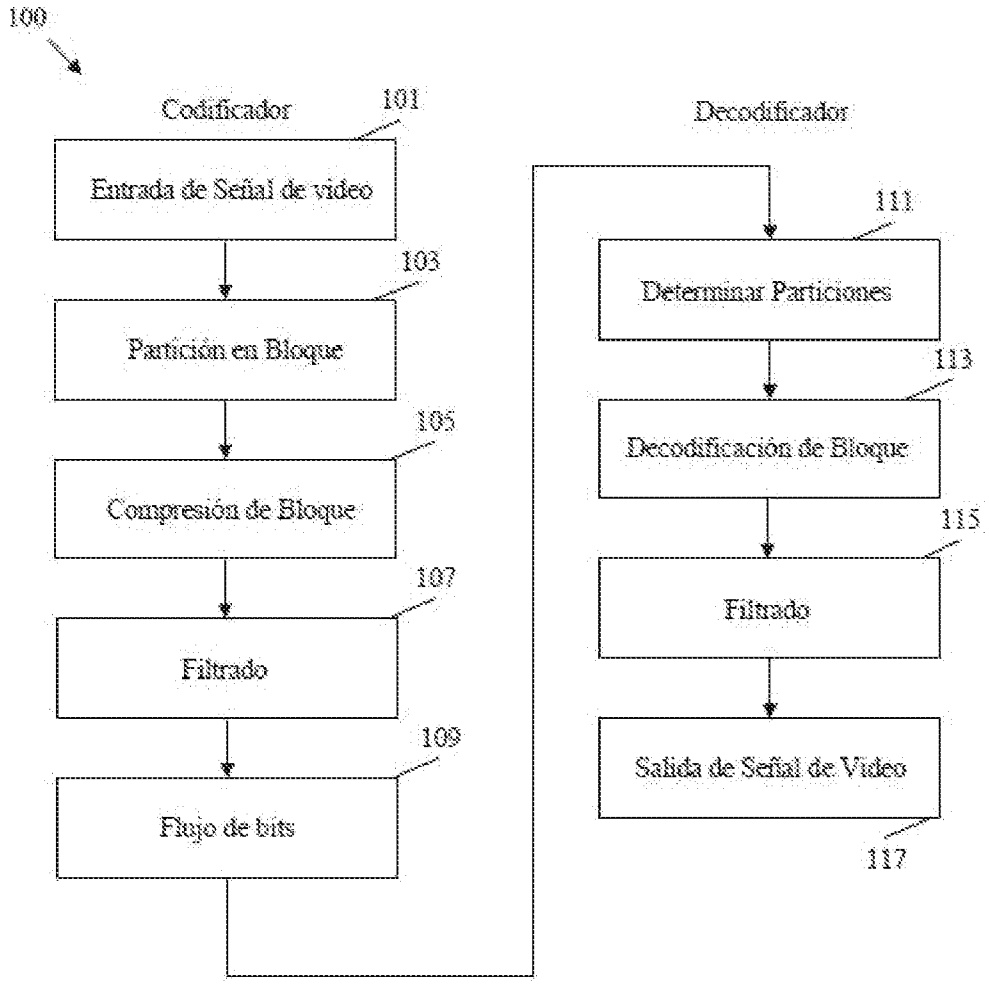
100
Codificador
101
Entrada de Señal de video
103
Partición en Bloque
105
Compresión de Bloque
107
Filtrado
109
Flujo de bits
Decodificador
111
Determinar Particiones
113
Decodificación de Bloque
115
Filtrado
Salida de Señal de Video
117

FIG. 1

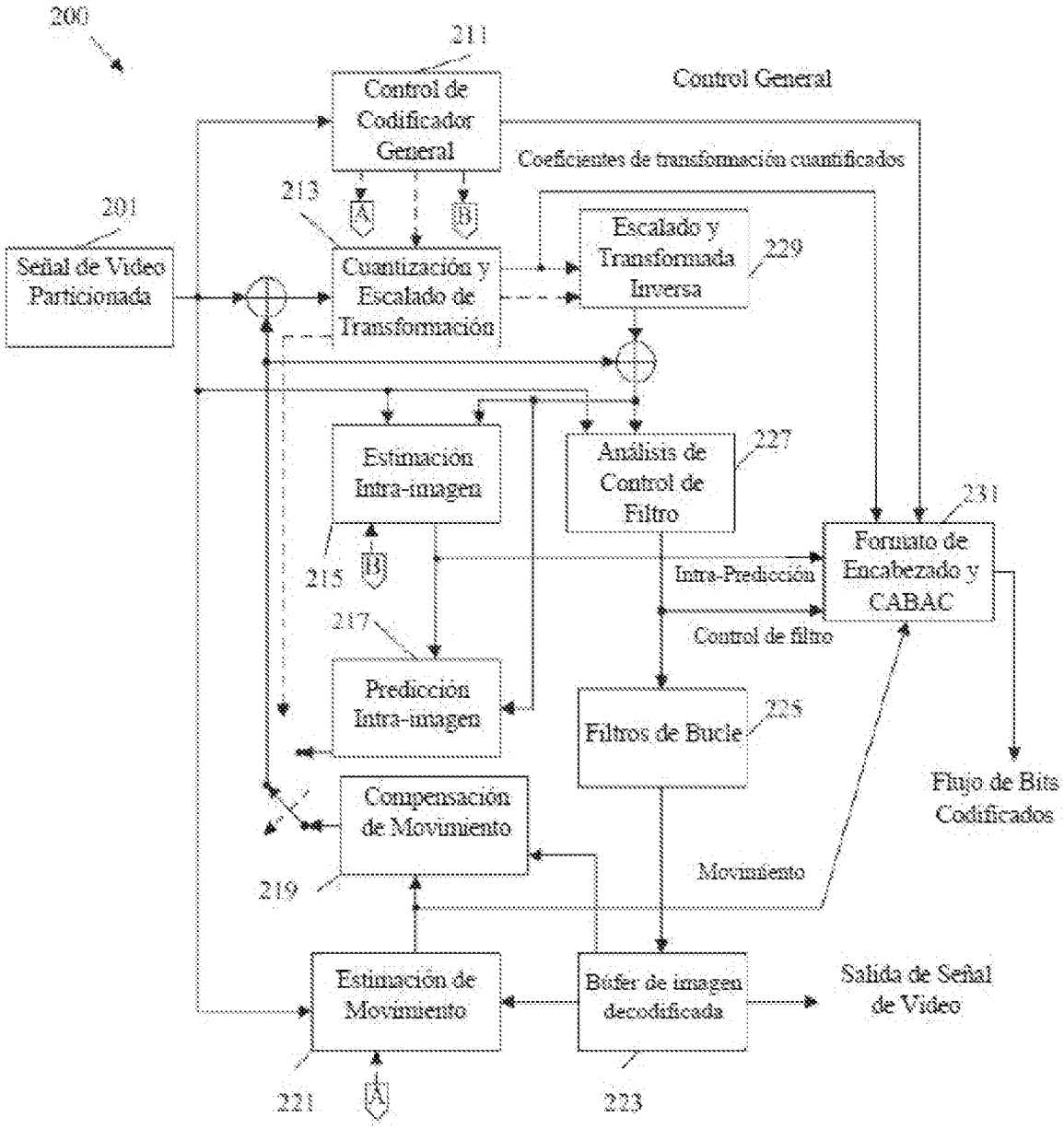
200
211
Control de Codificador General
Control General
Coeficientes de transformación cuantificados
213
A
B
201
Señal de Video Particionada
Cuantización y Escalado de Transformación
Escalado y Transformada Inversa
229
Estimación Intra-imagen
Análisis de Control de Filtro
227
231
Formato de Encabezado y CABAC
215
B
Intra-Predicción
Control de filtro
217
Predicción Intra-imagen
Filtros de Bucle
225
Flujo de Bits Codificados
Compensación de Movimiento
219
Movimiento
Estimación de Movimiento
Búfer de imagen decodificada
Salida de Señal de Video
221
A
223

FIG. 2

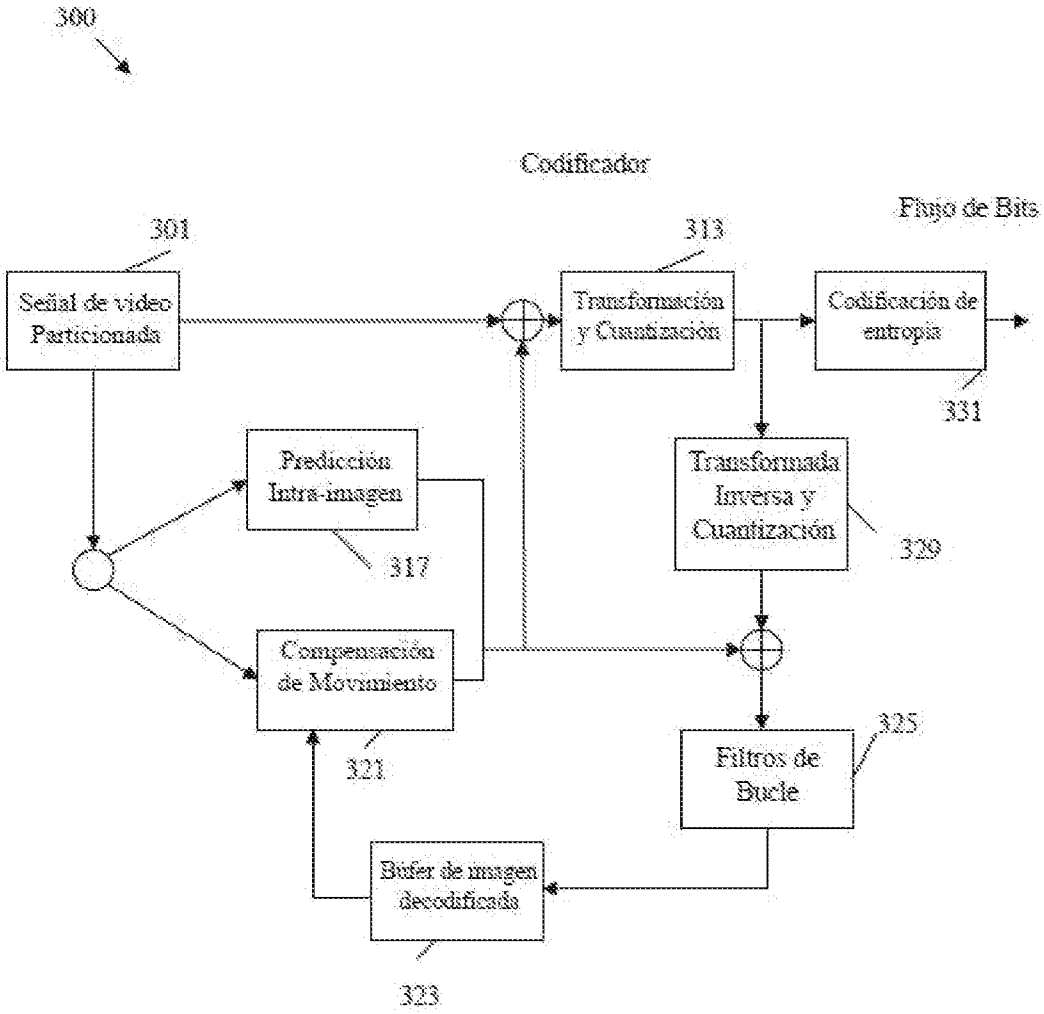
300
Codificador
Flujo de Bits
301
313
Señal de video Particionada
Transformación y Cuantización
Codificación de entropía
331
Predicción Intra-imagen
317
Transformada Inversa y Cuantización
329
Compensación de Movimiento
321
325
Filtros de Bucle
Búfer de imagen decodificada
323

FIG. 3

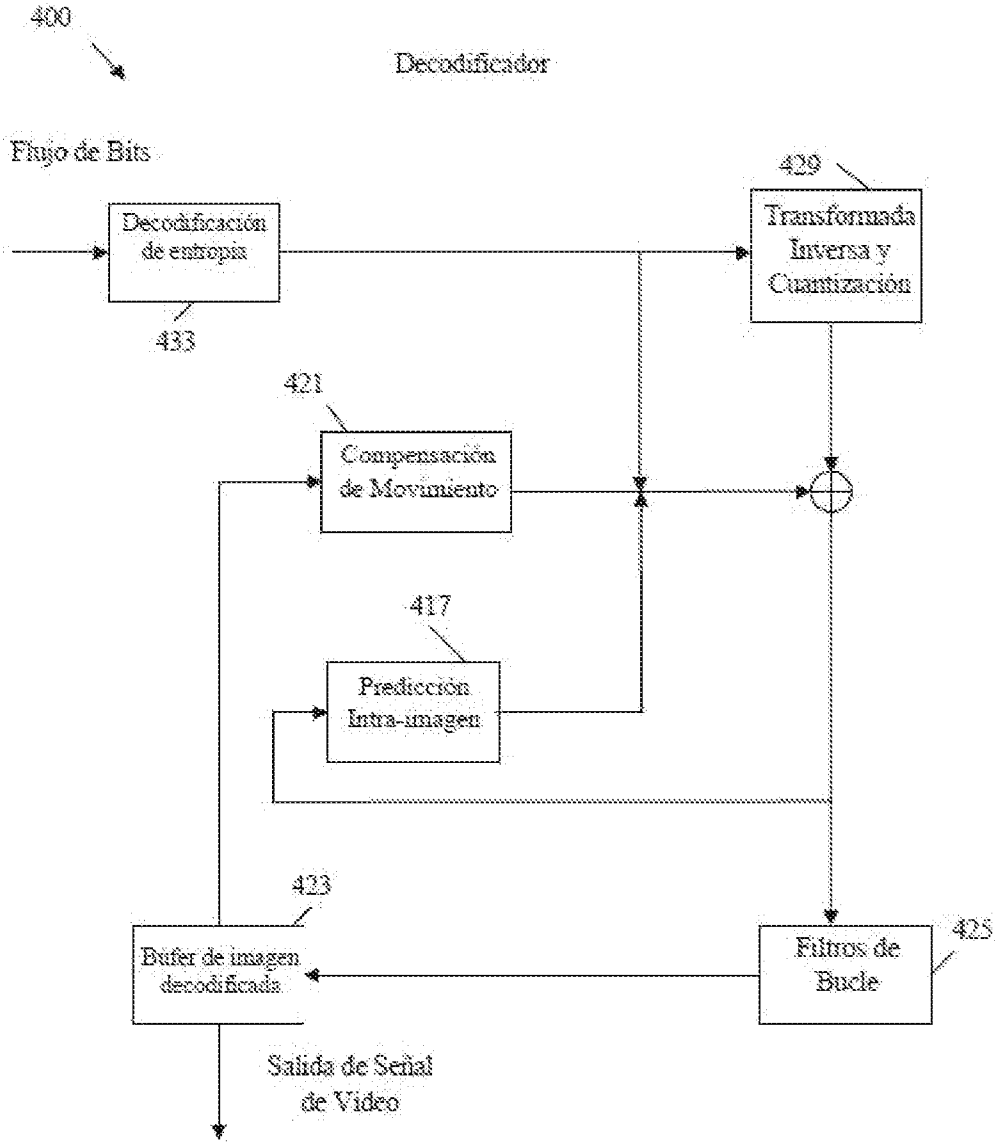
400
Decodificador
Flujo de Bits
Decodificación de entropía
433
429
Transformada Inversa y Cuantización
421
Compensación de Movimiento
417
Predicción Intra-imagen
423
Búfer de imagen decodificada
425
Filtros de Bucle
Salida de Señal de Video

FIG. 4

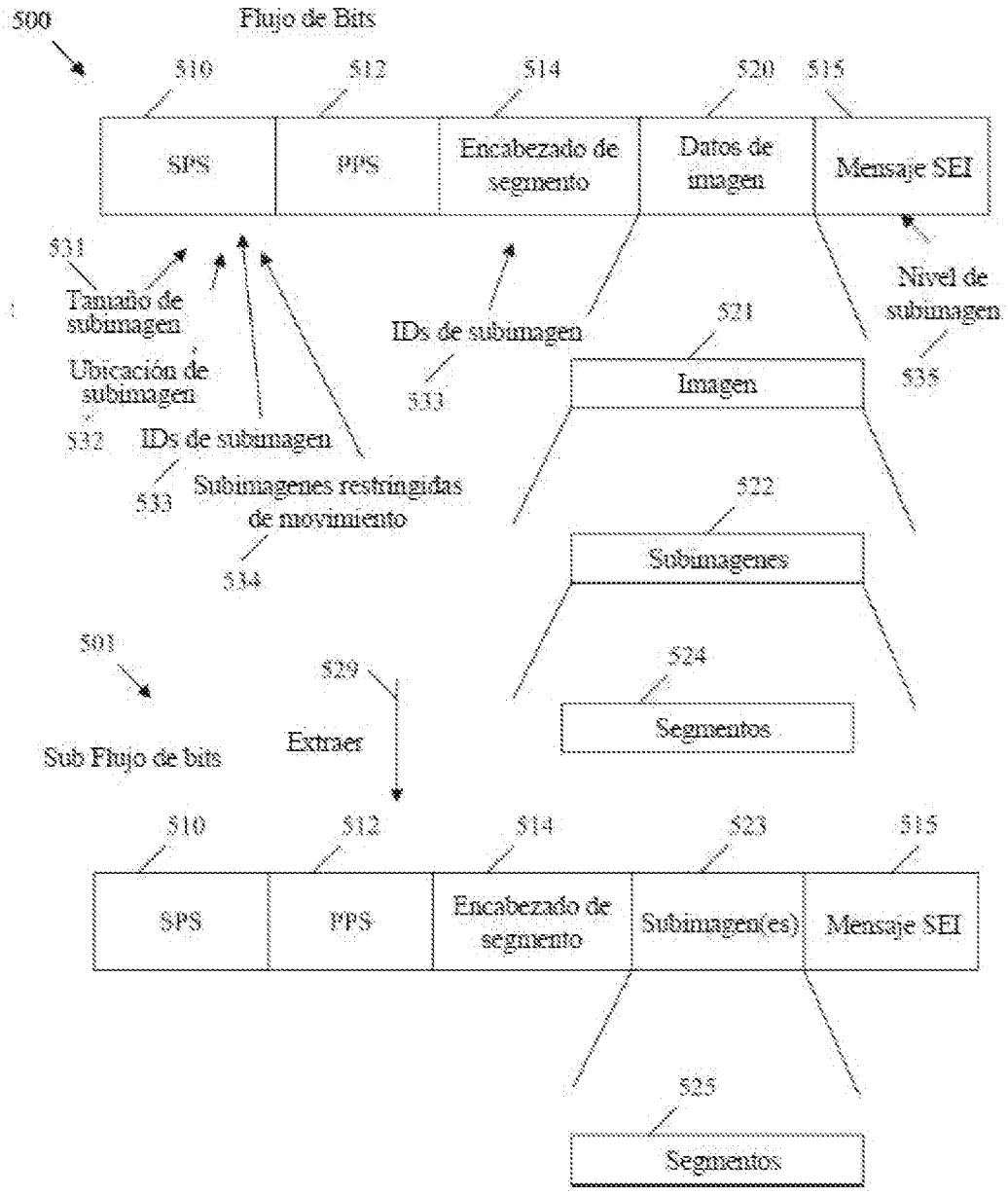
500
Flujo de Bits
510
512
514
520
515
SPS
PPS
Encabezado de segmento
Datos de imagen
Mensaje SEI
531
Tamaño de subimagen
Ubicación de subimagen
532
IDs de subimagen
533
Subimagenes restringidas de movimiento
534
IDs de subimagen
533
521
Imagen
Nivel de subimagen
535
522
Subimagenes
524
Segmentos
501
Sub Flujo de bits
529
Extraer
510
512
514
523
515
SPS
PPS
Encabezado de segmento
Subimagen(es)
Mensaje SEI
525
Segmentos

FIG. 5

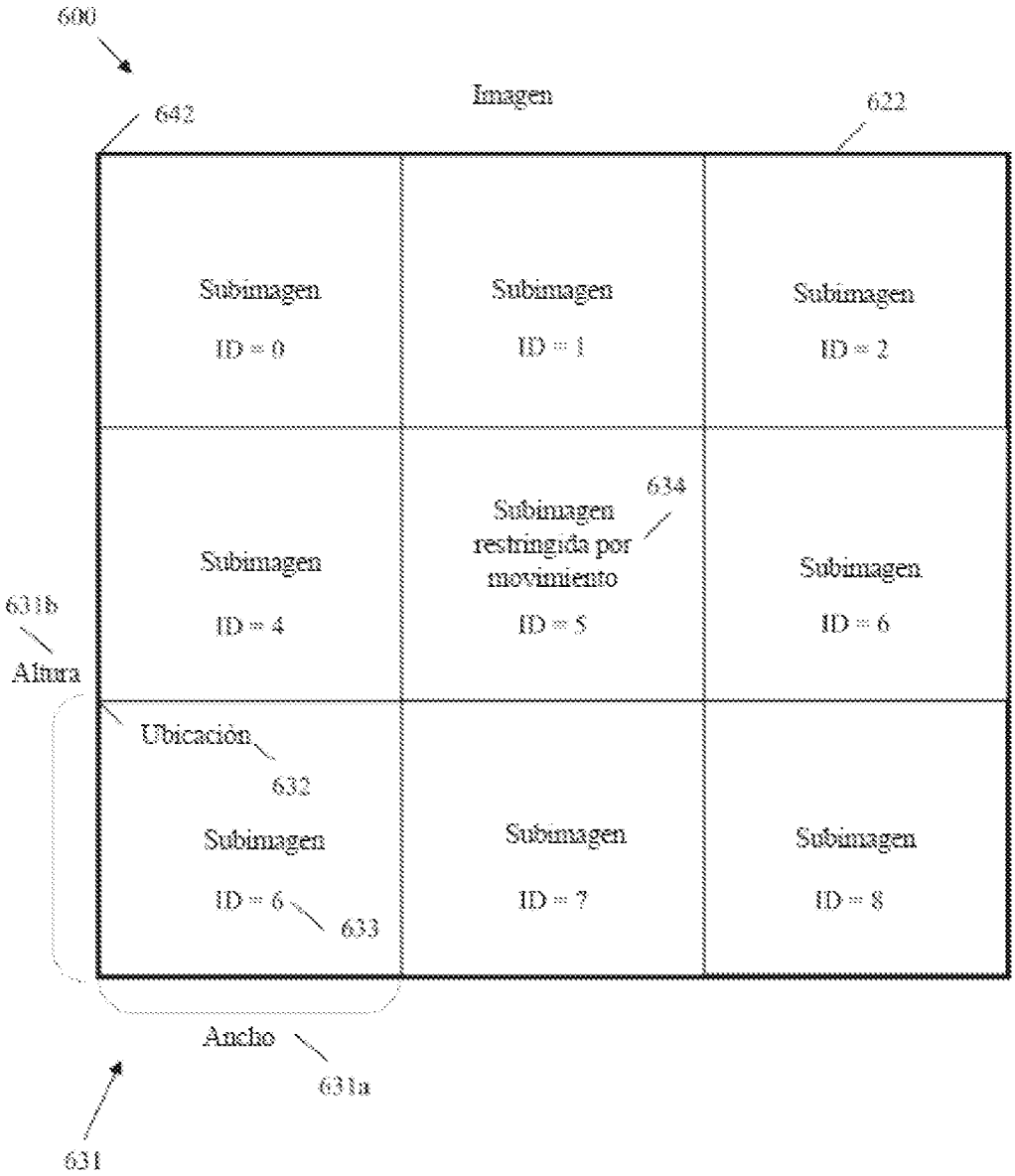
600
Imagen
642
622
Subimagen
ID = 0
Subimagen
ID = 1
Subimagen
ID = 2
Subimagen
ID = 4
634
Subimagen
restringida por
movimiento
ID = 5
Subimagen
ID = 6
631b
Altura
Ubicación
632
Subimagen
ID = 6
633
Subimagen
ID = 7
Subimagen
ID = 8
Ancho
631a
631

FIG. 6

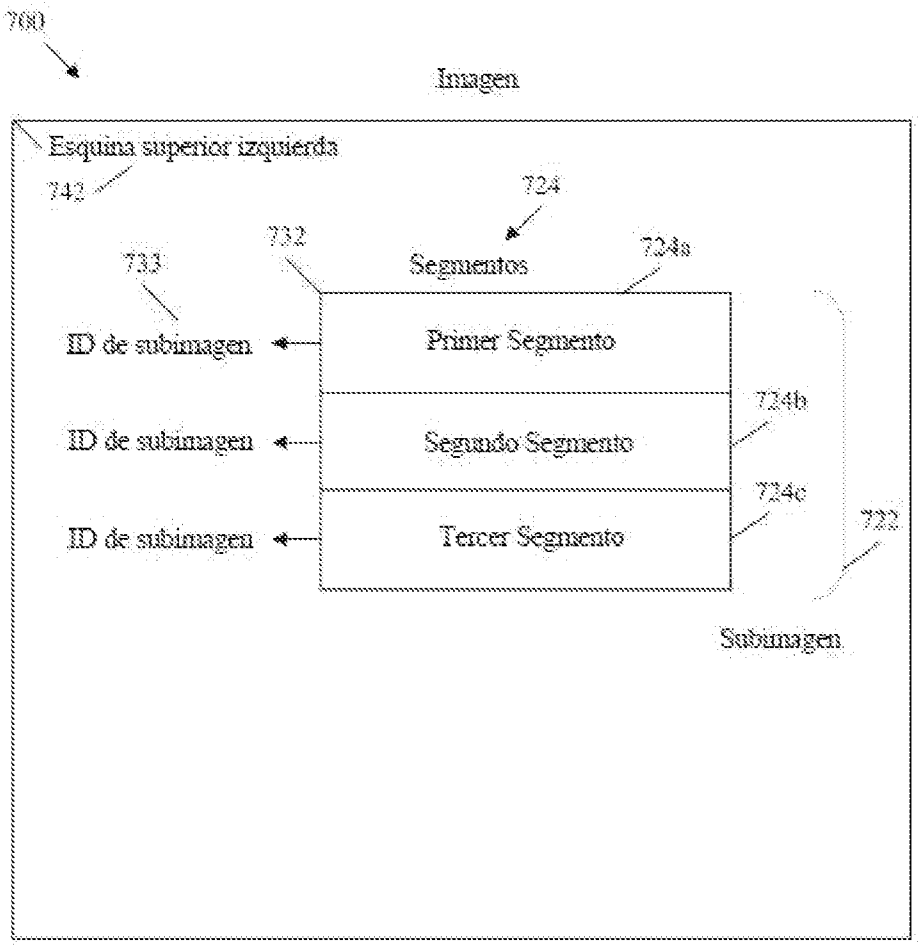
700
Imagen
Esquina superior izquierda
742
724
732
Segmentos
724a
733
ID de subimagen
Primer Segmento
724b
ID de subimagen
Segundo Segmento
724c
ID de subimagen
Tercer Segmento
722
Subimagen

FIG. 7

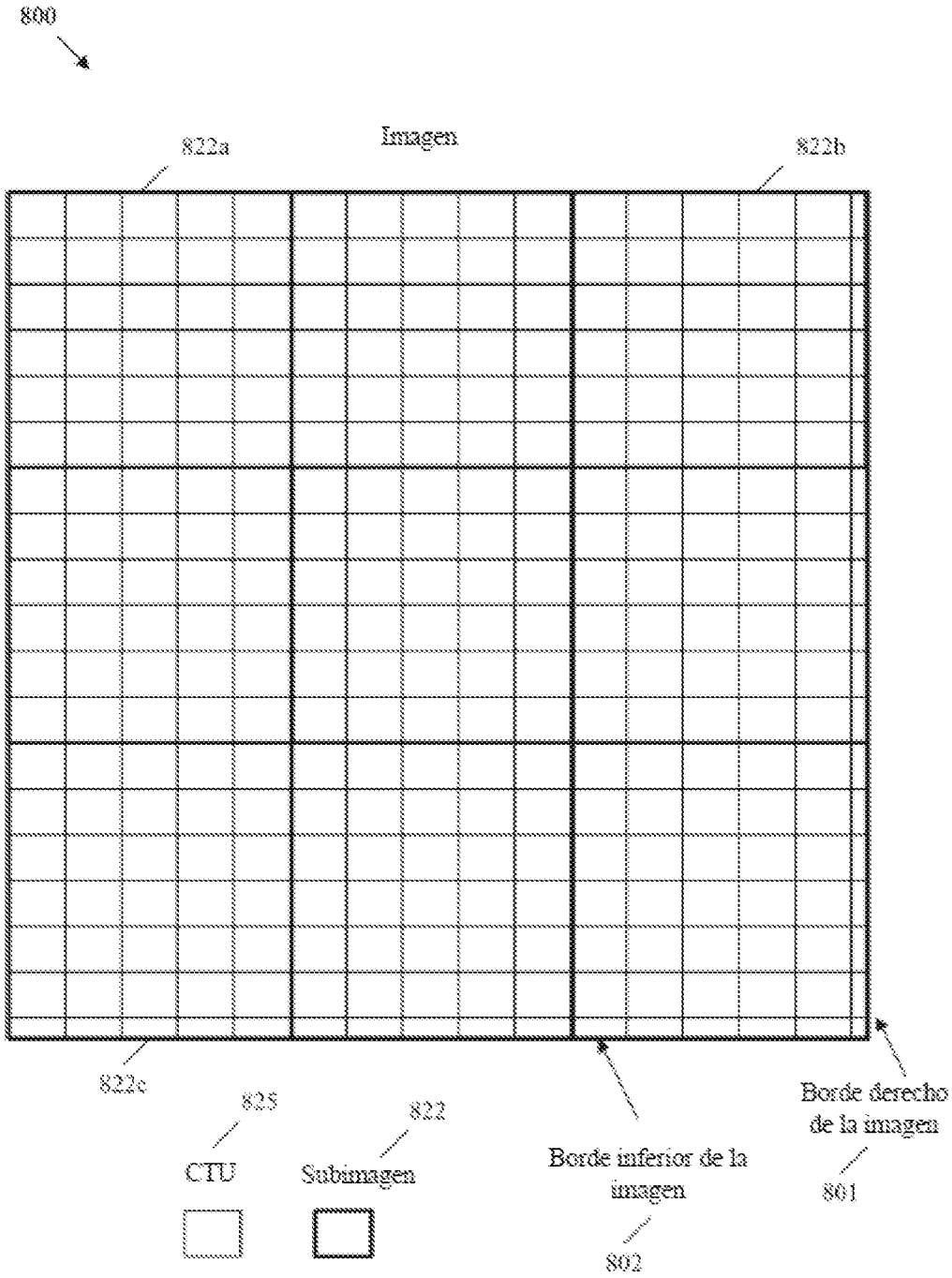
800
822a
Imagen
822b
822c
825
CTU
822
Subimagen
Borde inferior de la imagen
802
Borde derecho de la imagen
801

FIG. 8

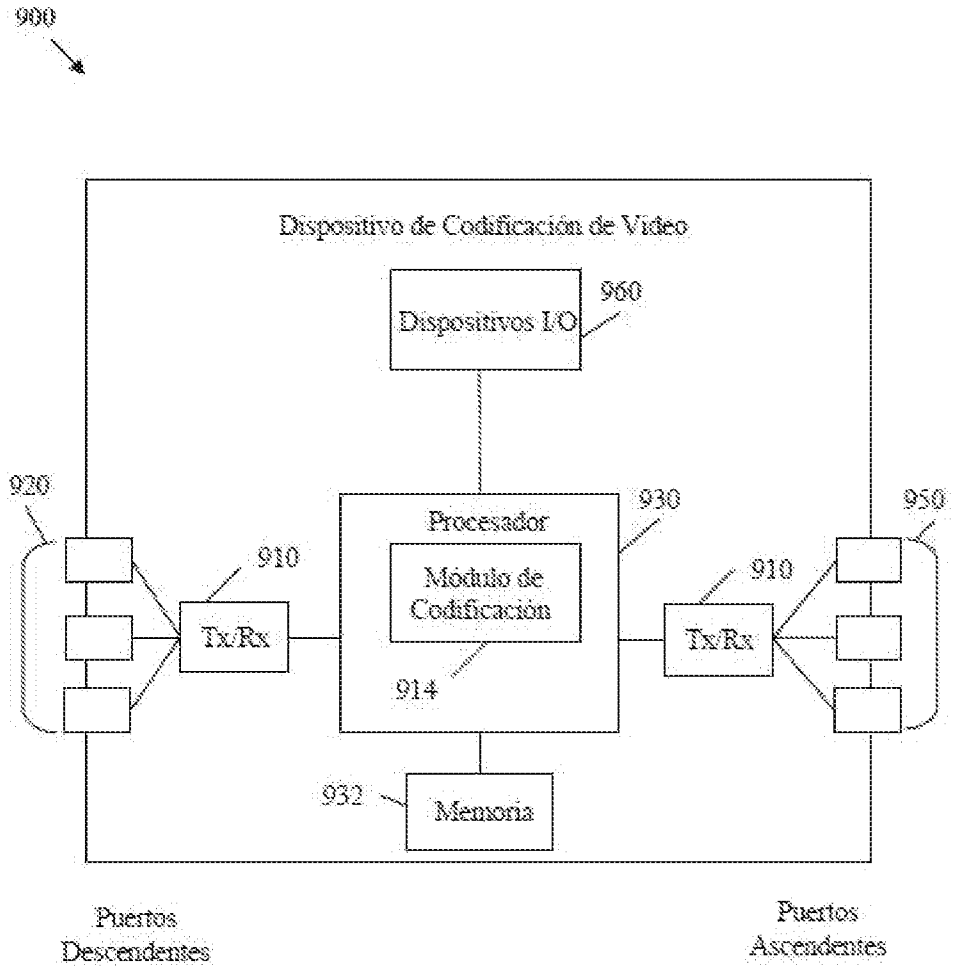
900
Dispositivo de Codificación de Video
Dispositivos I/O
960
920
Procesador
930
950
910
Módulo de Codificación
910
Tx/Rx
Tx/Rx
914
932
Memoria
Puertos Descendentes
Puertos Ascendentes

FIG. 9

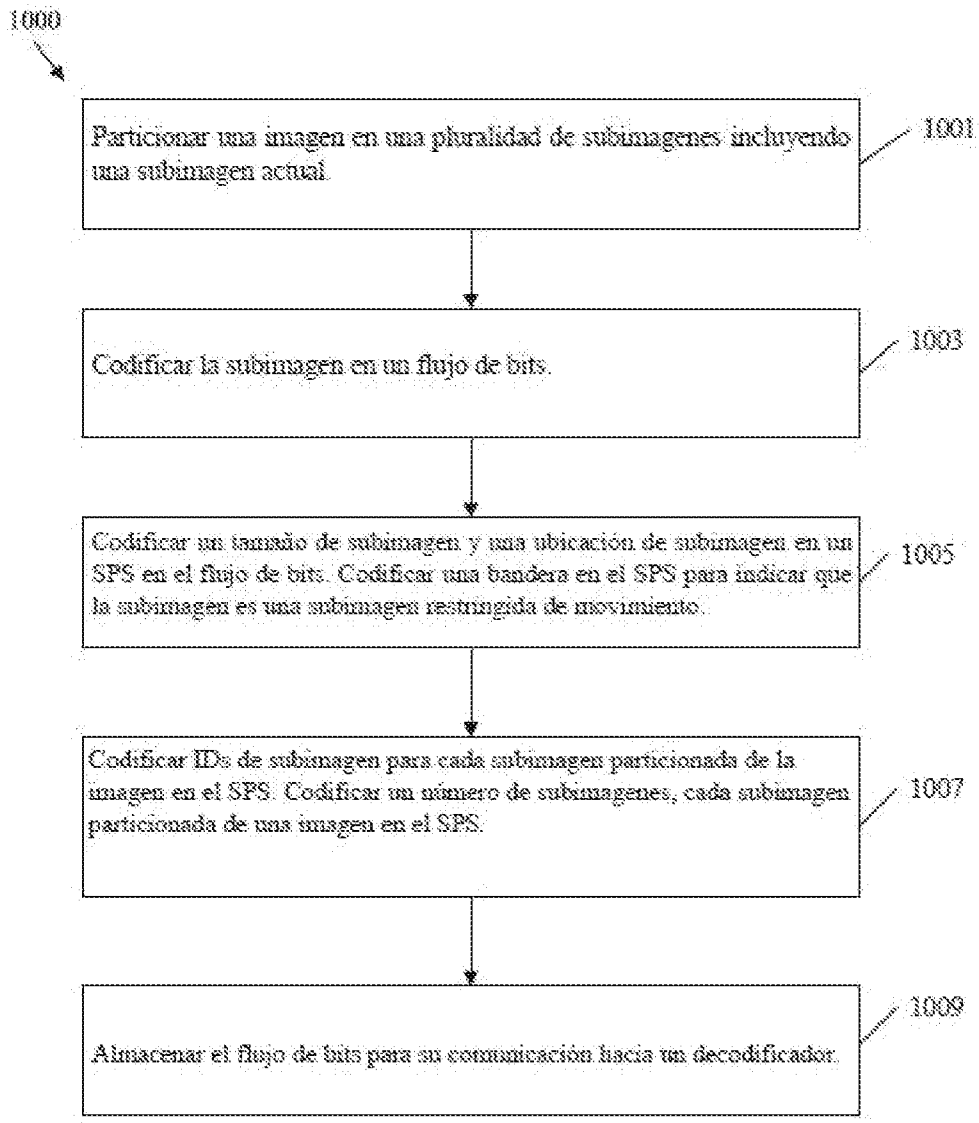
1000
Particionar una imagen en una pluralidad de subimagenes incluyendo una subimagen actual.
1001
Codificar la subimagen en un flujo de bits.
1003
Codificar un tamaño de subimagen y una ubicación de subimagen en un SPS en el flujo de bits. Codificar una bandera en el SPS para indicar que la subimagen es una subimagen restringida de movimiento.
1005
Codificar IDs de subimagen para cada subimagen particionada de la imagen en el SPS. Codificar un número de subimagenes, cada subimagen particionada de una imagen en el SPS.
1007
Almacenar el flujo de bits para su comunicación hacia un decodificador.
1009

FIG. 10

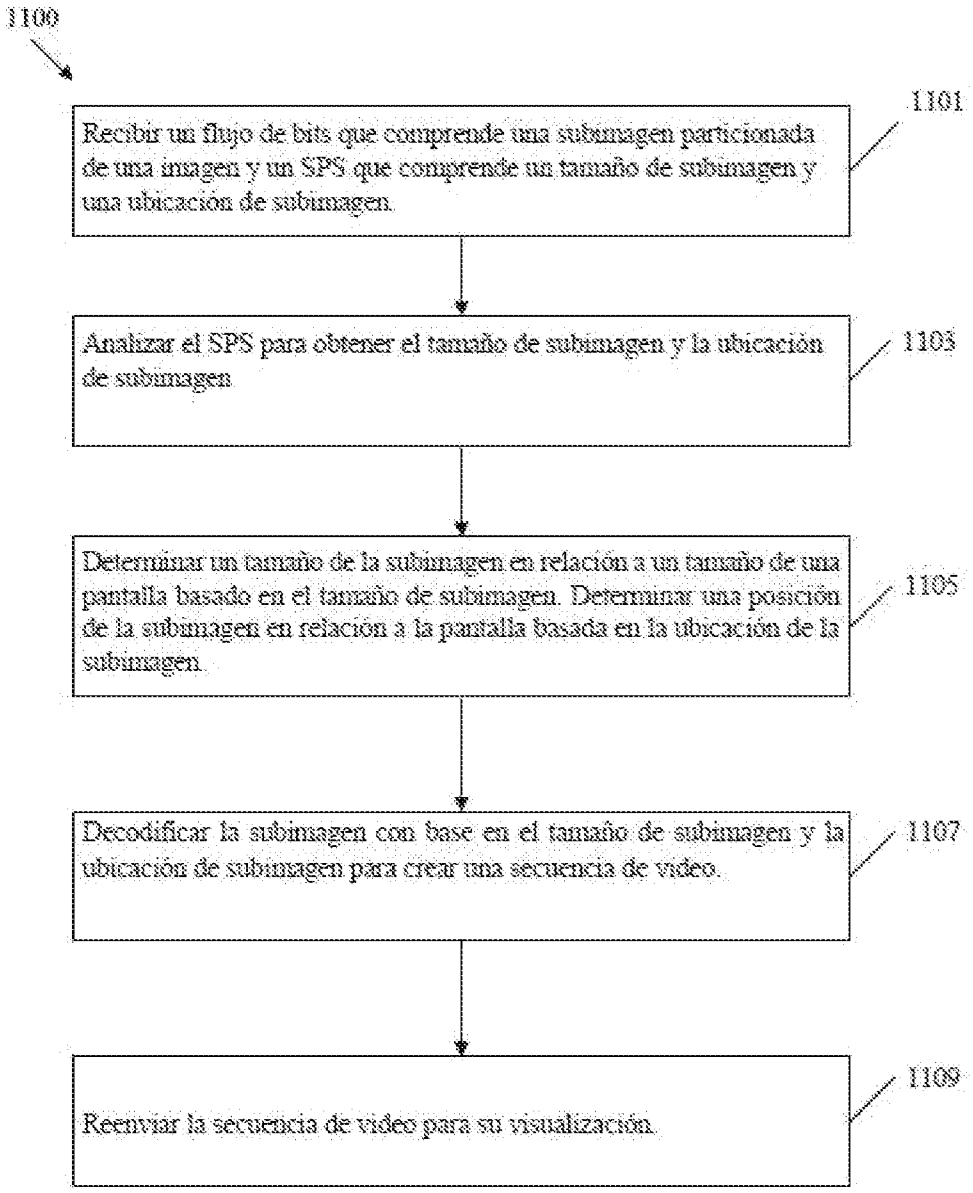
1100
Recibir un flujo de bits que comprende una subimagen particionada de una imagen y un SPS que comprende un tamaño de subimagen y una ubicación de subimagen.
1101
Analizar el SPS para obtener el tamaño de subimagen y la ubicación de subimagen
1103
Determinar un tamaño de la subimagen en relación a un tamaño de una pantalla basado en el tamaño de subimagen. Determinar una posición de la subimagen en relación a la pantalla basada en la ubicación de la subimagen.
1105
Decodificar la subimagen con base en el tamaño de subimagen y la ubicación de subimagen para crear una secuencia de video.
1107
Reenviar la secuencia de video para su visualización.
1109

FIG. 11

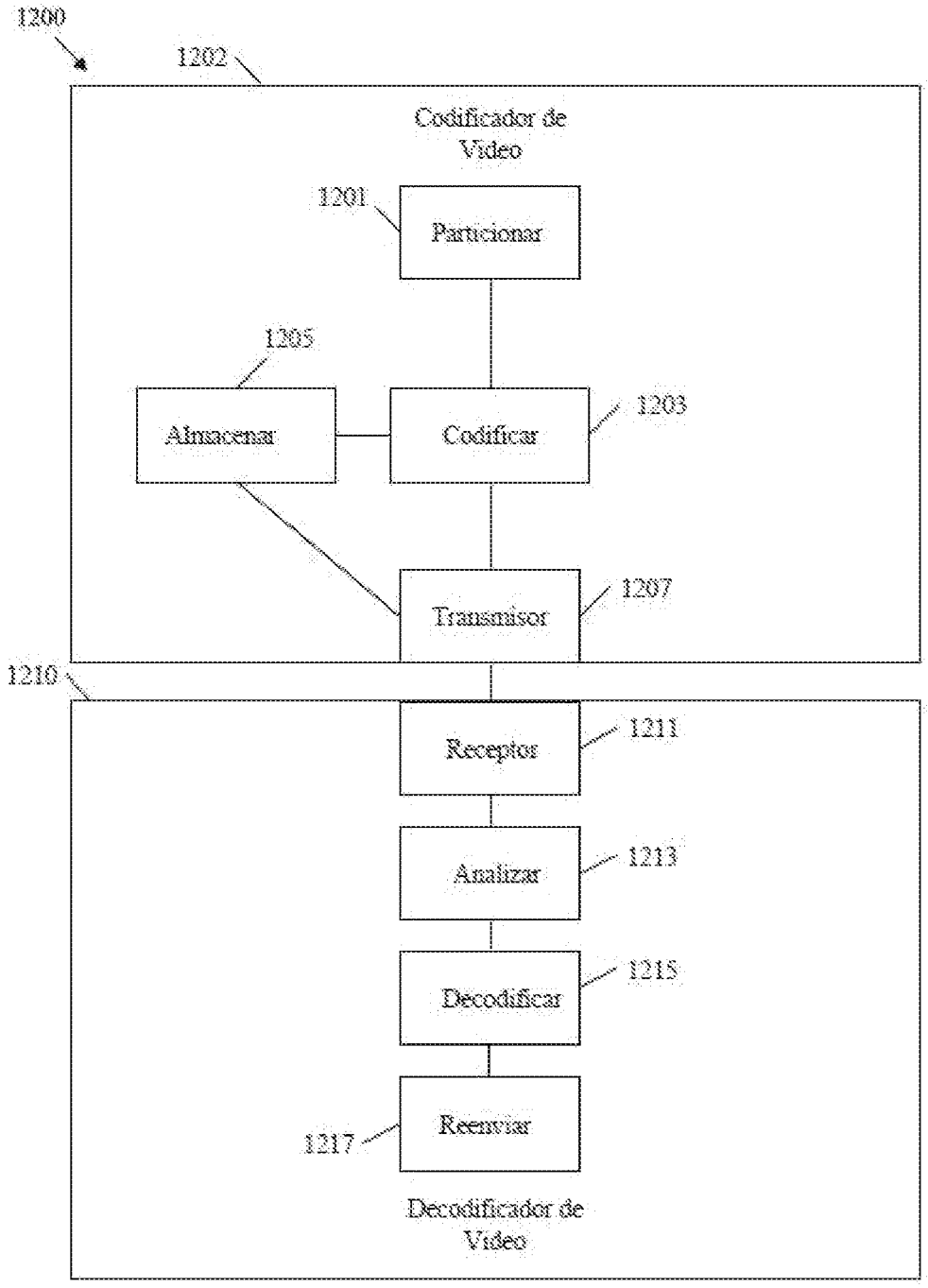
1200
1202
Codificador de Video
1201
Particionar
1205
Almacenar
Codificar
1203
Transmisor
1207
1210
Receptor
1211
Analizar
1213
Decodificar
1215
Reenviar
1217
Decodificador de Video

FIG. 12