(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 980 262**

(51) Int. Cl.:

H04N 19/174 (2014.01)
H04N 19/70 (2014.01)
H04N 19/46 (2014.01)
G06T 3/40 (2014.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **27.12.2019 PCT/US2019/068793**

(87) Fecha y número de publicación internacional: **02.07.2020 WO20140066**

(96) Fecha de presentación y número de la solicitud europea: **27.12.2019 E 19902918 (2)**

(97) Fecha y número de publicación de la concesión europea: **27.03.2024 EP 3903489**

(54) Título: **Mejoras de formación de mosaicos flexibles en codificación de vídeo**

(30) Prioridad:

27.12.2018 US 201862785511 P

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
30.09.2024

(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;
HENDRY, FNU y
SYCHEV, MAXIM

(74) Agente/Representante:

PONS ARIÑO, Ángel

DESCRIPCIÓN

Mejoras de formación de mosaicos flexibles en codificación de vídeo

Campo técnico

La presente descripción está relacionada en general con la codificación de vídeo, y está específicamente relacionada con un esquema de mosaicos de vídeo flexible que soporta múltiples mosaicos con diferentes resoluciones en la misma imagen.

Antecedentes

La cantidad de datos de vídeo necesarios para representar incluso un vídeo 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 vídeo generalmente se comprimen antes de comunicarse a través de las redes de telecomunicaciones de hoy en día. El tamaño de un vídeo también podría ser un problema cuando el vídeo se almacena en un dispositivo de almacenamiento porque los recursos de memoria pueden ser limitados. Los dispositivos de compresión de vídeo a menudo utilizan software y/o hardware en la fuente para codificar los datos de vídeo antes de su transmisión o almacenamiento, disminuyendo así la cantidad de datos necesarios para representar imágenes de vídeo digitales. Los datos comprimidos son, a continuación, recibidos en el destino por un dispositivo de descompresión de vídeo que decodifica los datos de vídeo. Con recursos de red limitados y demandas cada vez mayores de mayor calidad de vídeo, 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 Y-K WANG (HUAWE) Y COL.: "Spec text for the agreed starting point on slicing and tiling" (Texto de especificación para el punto de partida acordado sobre segmento y mosaico), 12. REUNIÓN JVET; 20181003 – 20181012; MACAO; (EL EQUIPO CONJUNTO DE EXPLORACIÓN DE VÍDEO DE ISO/IEC JTC1/SC29/WG11 E UIT-T SG. 16) proporciona una implementación del texto de especificación para el punto de partida acordado sobre segmento y mosaico. La solicitud de patente de los EE.UU. (US2013202051A1) da a conocer subimágenes para equilibrar la tasa de píxeles en plataformas multinúcleo. ZHOU (TI) M: "AHG4: Habilitar decodificación en paralelo con mosaicos", 9. REUNIÓN JCT-VC; 20120427 – 20120507; GINEBRA; (EQUIPO CONJUNTO COLABORATIVO EN CODIFICACIÓN DE VÍDEO DE ISO/IEC JTC1/SC29/WG11 E UTU-T SG. 16) propone exigir subimágenes divididas de manera uniforme para niveles altos para garantizar el equilibrio de tasa de píxeles entre núcleos cuando las subimágenes se procesan en paralelo.

El documento WO 2019/243539 da a conocer métodos generales de división de imágenes en particiones.

El documento EP 3 902 260 A1 da a conocer métodos generales de división de imágenes en particiones.

Compendio

Las realizaciones de la presente invención están definidas por las reivindicaciones independientes. En las reivindicaciones dependientes se presentan características adicionales de las realizaciones de la invención. A continuación, partes de la descripción y dibujos se refieren a realizaciones anteriores que no comprenden necesariamente todas las características para implementar realizaciones de la invención reivindicada que no se representan como realizaciones de la invención, si no como ejemplos útiles para la comprensión de las realizaciones de la invención.

En una realización, la descripción incluye un método según la reivindicación 1. Los sistemas de codificación de vídeo pueden emplear segmentos y mosaicos para dividir imágenes en particiones. Ciertas aplicaciones de transmisión (por ejemplo, realidad virtual (VR) y teleconferencias) pueden mejorarse si se puede enviar una sola imagen que contenga múltiples regiones codificadas con diferentes resoluciones. Es posible que algunos mecanismos de formación de segmentos y de mosaicos no soporten dicha funcionalidad porque los mosaicos con diferentes resoluciones pueden tratarse de manera diferente. Por ejemplo, un mosaico a una primera resolución puede contener un solo segmento de datos, mientras que un mosaico a una segunda resolución puede contener múltiples segmentos de datos debido a las diferencias en la densidad de píxeles. Los presentes aspectos emplean un esquema de formación de mosaicos flexible que incluye mosaicos de primer nivel y mosaicos de segundo nivel. Los mosaicos de segundo nivel se crean dividiendo en particiones los mosaicos de primer nivel. Este esquema de mosaicos permite que un mosaico de primer nivel contenga un segmento de datos a una primera resolución y que un mosaico de primer nivel contenga mosaicos de segundo nivel para contener una pluralidad de segmentos a una segunda resolución. Los mosaicos se pueden asignar a grupos de mosaicos. Los aspectos presentes limitan los grupos de mosaicos que contienen mosaicos de segundo nivel a ser rectangulares en contraste con la exploración de trama. Este enfoque crea límites que soportan la extracción y el tratamiento separados de diferentes contenidos. Por ejemplo, los grupos de mosaicos que contienen contenido a una primera resolución y los grupos de mosaicos que contienen contenido a una segunda resolución tienen una forma natural para soportar la visualización simultánea en

pantallas y o la extracción separada para su uso en dispositivos de visualización montados en la cabeza. Por lo tanto, el esquema de formación de mosaicos flexible dado a conocer permite que un codificador/decodificador (códec) soporte una imagen que contenga múltiples resoluciones y, por lo tanto, aumenta la funcionalidad tanto del codificador como del decodificador.

Opcionalmente, en cualquiera de los aspectos anteriores, se proporciona otra implementación del aspecto, en donde los mosaicos de primer nivel fuera del subconjunto contienen datos de imagen a una primera resolución y los mosaicos de segundo nivel contienen datos de imagen a una segunda resolución diferente de la primera resolución.

Opcionalmente, en cualquiera de los aspectos anteriores, se proporciona otra implementación del aspecto, en donde cada uno de los uno o más grupos de mosaicos está limitados a cubrir una parte rectangular de la imagen cuando cualquier mosaico de primer nivel se divide en una pluralidad de mosaicos de segundo nivel.

Opcionalmente, en cualquiera de los aspectos anteriores, se proporciona otra implementación del aspecto, en donde los mosaicos de primer nivel y los mosaicos de segundo nivel no se asignan al uno o más grupos de mosaicos mediante un orden de exploración de trama que atraviesa horizontalmente la imagen desde un límite izquierdo a un límite derecho.

Opcionalmente, en cualquiera de los aspectos anteriores, se proporciona otra implementación del aspecto, en donde cubrir la parte rectangular de la imagen incluye cubrir menos de una parte horizontal completa de la imagen.

Opcionalmente, en cualquiera de los aspectos anteriores, se proporciona otra implementación del aspecto, en donde cada mosaico de segundo nivel contiene un solo segmento de datos de imagen de la imagen.

Opcionalmente, en cualquiera de los aspectos anteriores, se proporciona otra implementación del aspecto, que comprende además la codificación, mediante el procesador, de filas de mosaicos de segundo nivel y columnas de mosaicos de segundo nivel para mosaicos de primer nivel divididos en particiones, en donde las filas de mosaicos de segundo nivel y las columnas de mosaicos de segundo nivel están codificadas en un conjunto de parámetros de imagen asociados con la imagen.

Opcionalmente, en cualquiera de los aspectos anteriores, se proporciona otra implementación del aspecto, en donde los datos que indican explícitamente si un mosaico de primer nivel está dividido en particiones de mosaicos de segundo nivel se omiten de la corriente de bits para mosaicos de primer nivel con un ancho que es menor que un umbral de ancho mínimo y una altura que es menor que un umbral de altura mínima, y en donde las filas de mosaicos de segundo nivel y las columnas de mosaicos de segundo nivel se omiten de la corriente de bits para mosaicos de primer nivel divididos en particiones con un ancho que es menor que el doble del umbral de ancho mínimo y una altura que es menor que el doble del umbral de altura mínima.

En una realización, la descripción incluye un método según la reivindicación 7. Los sistemas de codificación de vídeo pueden emplear segmentos y mosaicos para dividir imágenes en particiones. Ciertas aplicaciones de transmisión en directo (por ejemplo, realidad virtual y teleconferencias) pueden mejorarse si se puede enviar una sola imagen que contenga múltiples regiones codificadas con diferentes resoluciones. Es posible que algunos mecanismos de formación de segmentos y mosaicos no soporten dicha funcionalidad porque los mosaicos con diferentes resoluciones pueden tratarse de manera diferente. Por ejemplo, un mosaico a una primera resolución puede contener un solo segmento de datos, mientras que un mosaico a una segunda resolución puede contener múltiples segmentos de datos debido a las diferencias en la densidad de píxeles. Los presentes aspectos emplean un esquema de formación de mosaicos flexible que incluye mosaicos de primer nivel y mosaicos de segundo nivel. Los mosaicos de segundo nivel se crean dividiendo mosaicos de primer nivel en particiones. Este esquema de mosaicos permite que un mosaico de primer nivel contenga un segmento de datos a una primera resolución y un mosaico de primer nivel que contenga mosaicos de segundo nivel para contener una pluralidad de segmentos a una segunda resolución. Los mosaicos se pueden asignar a grupos de mosaicos. Los aspectos presentes limitan los grupos de mosaicos que contienen mosaicos de segundo nivel a ser rectangulares en contraste con la exploración de trama. Este enfoque crea límites que soportan la extracción y el tratamiento separados de diferentes contenidos. Por ejemplo, los grupos de mosaicos que contienen contenido a una primera resolución y los grupos de mosaicos que contienen contenido a una segunda resolución tienen una forma natural para soportar la visualización simultánea en pantallas y/o la extracción separada para su uso en dispositivos de visualización montados en la cabeza. Por lo tanto, el esquema de formación de mosaicos flexible dado a conocer permite que un códec soporte una imagen que contenga múltiples resoluciones y, por lo tanto, aumenta la funcionalidad tanto del codificador como del decodificador.

Opcionalmente, en cualquiera de los aspectos anteriores, se proporciona otra implementación del aspecto, en donde los mosaicos de primer nivel fuera del subconjunto contienen datos de imagen a una primera resolución y los mosaicos de segundo nivel contienen datos de imagen a una segunda resolución diferente de la primera

resolución.

Opcionalmente, en cualquiera de los aspectos anteriores, se proporciona otra implementación del aspecto, en donde cada uno de los uno o más grupos de mosaicos están limitados a cubrir una parte rectangular de la imagen cuando cualquier mosaico de primer nivel se divide en particiones de una pluralidad de mosaicos de segundo nivel.

Opcionalmente, en cualquiera de los aspectos anteriores, se proporciona otra implementación del aspecto, en donde los mosaicos de primer nivel y los mosaicos de segundo nivel no se asignan al uno o más grupos de mosaicos mediante una orden de exploración de trama que atraviesa horizontalmente la imagen desde un límite izquierdo a un límite derecho.

Opcionalmente, en cualquiera de los aspectos anteriores, se proporciona otra implementación del aspecto, en donde cubrir la parte rectangular de la imagen incluye cubrir menos de una parte horizontal completa de la imagen.

Opcionalmente, en cualquiera de los aspectos anteriores, se proporciona otra implementación del aspecto, en donde cada mosaico de segundo nivel contiene un solo segmento de datos de imagen de la imagen.

Opcionalmente, en cualquiera de los aspectos anteriores, se proporciona otra implementación del aspecto, que comprende además la obtención, mediante el procesador, de filas de mosaicos de segundo nivel y columnas de mosaicos de segundo nivel para mosaicos de primer nivel divididos en particiones a partir de un conjunto de parámetros de imagen asociados con la imagen.

Opcionalmente, en cualquiera de los aspectos anteriores, se proporciona otra implementación del aspecto, en donde los datos que indican explícitamente si un mosaico de primer nivel está dividido en mosaicos de segundo nivel se omiten de la corriente de bits para mosaicos de primer nivel con un ancho que es menor que un umbral de ancho mínimo y una altura que es menor que un umbral de altura mínima, y en donde las filas de mosaicos de segundo nivel y las columnas de mosaicos de segundo nivel se omiten de la corriente de bits para mosaicos de primer nivel divididos en particiones con un ancho que es menor que el doble del umbral de ancho mínimo y una altura que es menor que el doble del umbral de altura mínima.

En una realización, la descripción incluye un dispositivo de codificación de vídeo según la reivindicación 12, y dispositivo de decodificación de vídeo según la reivindicación 13.

En una realización, la descripción incluye un medio legible por ordenador no transitorio según la reivindicación 17.

En un ejemplo, la descripción incluye un codificador que comprende: un medio de división en particiones para: dividir una imagen en una pluralidad de mosaicos de primer nivel; y dividir un subconjunto de los mosaicos de primer nivel en una pluralidad de mosaicos de segundo nivel; un medio de asignación para asignar los mosaicos de primer nivel y los mosaicos de segundo nivel en uno o más grupos de mosaicos de tal manera que todos los mosaicos en un grupo de mosaicos asignado que contiene los mosaicos de segundo nivel estén limitados a cubrir una parte rectangular de la imagen; un medio de codificación para codificar, mediante el procesador, los mosaicos de primer nivel y los mosaicos de segundo nivel en una corriente de bits; y un medio de almacenamiento para almacenar la corriente de bits para la comunicación hacia un decodificador.

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

En un ejemplo, la descripción incluye un decodificador que comprende: un medio de recepción para recibir una corriente de bits que incluye una imagen dividida en una pluralidad de mosaicos de primer nivel, en donde un subconjunto de mosaicos de primer nivel se divide además en una pluralidad de mosaicos de segundo nivel, y en donde los mosaicos de primer nivel y los mosaicos de segundo nivel se asignan a uno o más grupos de mosaicos de tal manera que todos los mosaicos de un grupo de mosaicos asignado que contiene los mosaicos de segundo nivel están limitados a cubrir una parte rectangular de la imagen; un medio de determinación para determinar una configuración de los mosaicos de primer nivel y una configuración de los mosaicos de segundo nivel basándose en el uno o más grupos de mosaicos; un medio de decodificación para decodificar los mosaicos de primer nivel y los mosaicos de segundo nivel basándose en la configuración de los mosaicos de primer nivel y en la configuración de los mosaicos de segundo nivel; y un medio de generación para generar una secuencia de vídeo reconstruida para la visualización basándose en los mosaicos de primer nivel y en los mosaicos de segundo nivel decodificados.

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

anteriores.

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

Estas y otras características se comprenderán 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 descripción, 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 ejemplar de codificación de una señal de vídeo.

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

La fig. 3 es un diagrama esquemático que ilustra un codificador de vídeo ejemplar.

La fig. 4 es un diagrama esquemático que ilustra un decodificador de vídeo ejemplar.

La fig. 5 es un diagrama esquemático que ilustra una corriente de bits ejemplar que contiene una secuencia de vídeo codificada.

Las figs. 6A a 6E ilustran un mecanismo ejemplar para crear una pista de extracción para combinar subimágenes de múltiples resoluciones de diferentes corrientes de bits en una sola imagen para su uso en aplicaciones de realidad virtual (VR).

La fig.7 ilustra una aplicación de videoconferencia ejemplar que empalma imágenes de múltiples resoluciones de diferentes corrientes de bits en una sola imagen para su visualización.

Las figs. 8A a 8B son diagramas esquemáticos que ilustran un esquema de formación de mosaicos de vídeo flexible ejemplar capaz de soportar múltiples mosaicos con diferentes resoluciones en la misma imagen.

La fig. 9 es un diagrama esquemático de un dispositivo de codificación de vídeo ejemplar.

La fig. 10 es un diagrama de flujo de un método ejemplar de codificación de una imagen empleando un esquema de formación de mosaicos flexible.

La fig. 11 es un diagrama de flujo de un método ejemplar de descodificación de una imagen empleando un esquema de formación de mosaicos flexible.

La fig. 12 es un diagrama esquemático de un sistema ejemplar para codificación de una secuencia de vídeo empleando un esquema de formación de mosaicos flexible.

Descripción detallada

Debería comprenderse desde el principio que aunque a continuación se proporciona una implementación ilustrativa de una o más realizaciones, los sistemas y/ o métodos dados a conocer pueden implementarse utilizando cualquier número de técnicas, ya sean conocidas actualmente o existentes. La descripción no debería limitarse de ninguna manera a las implementaciones ilustrativas, dibujos y técnicas ilustradas a continuación, incluyendo los diseños e implementaciones ejemplares ilustrados y descritos en la presente memoria, pero puede modificarse dentro del alcance de las reivindicaciones adjuntas junto con su alcance completo de equivalentes.

En la presente memoria se emplean varios acrónimos, tales como bloque de codificación en árbol (CTB), unidad de codificación en árbol (CTU), unidad de codificación (CU), secuencia de vídeo codificado (CVS), equipo conjunto de expertos en vídeo (JVET), conjunto de mosaicos con limitación de movimiento (MCTS), unidad de transferencia máxima (MTU), capa de abstracción de red (NAL), recuento de orden de imágenes (POC), carga útil de secuencia de bytes sin procesar (RBSP), conjunto de parámetros de secuencia (SPS), codificación de vídeo versátil (VVC) y borrador de trabajo (WD) .

Se pueden emplear muchas técnicas de compresión de vídeo para reducir el tamaño de los archivos de vídeo 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 parte de una imagen de vídeo) puede dividirse en bloques de vídeo, que también pueden denominarse bloques de árbol, bloques de codificación en árbol (CTB), unidades de codificación en árbol (CTU), unidades de codificación (CU) y/o nodos de codificación. Los bloques de vídeo de un segmento intra-codificado (I) de una imagen se codifican utilizando la predicción espacial con respecto a las muestras de referencia de los bloques contiguos de la misma imagen. Los bloques de vídeo en un segmento de predicción unidireccional (P) o predicción bidireccional (B) inter-codificado de una imagen pueden codificarse empleando la predicción espacial con respecto a las muestras de referencia de los bloques contiguos 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. Por consiguiente, un bloque inter-codificado 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 intra-codificado se codifica según un modo de intra-codificación y los datos residuales. Para una compresión adicional, los datos residuales se pueden transformar del dominio de píxeles a un dominio de transformada. Estos dan como resultado coeficientes de transformada residuales, que pueden cuantificarse. Los coeficientes de transformada cuantificados pueden disponerse inicialmente en una formación bidimensional. Los coeficientes de transformada cuantificados se pueden escanear con el fin de producir un vector unidimensional de coeficientes de transformada. Se puede aplicar codificación por entropía para lograr una compresión aún mayor. Dichas técnicas de compresión de vídeo se dan a conocer con mayor detalle a continuación.

Para asegurar que un vídeo codificado se pueda decodificar con precisión, el vídeo se codifica y decodifica según los estándares de codificación de vídeo correspondientes. Los estándares de codificación de vídeo incluyen el sector de normalización de la unión internacional de telecomunicaciones (UIT) (UIT-T) H.261, grupo de expertos de imágenes en movimiento (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, UIT-T H.263, ISO/IEC MPEG-4 parte 2, codificación de vídeo avanzada (AVC), también conocida como UIT-T H.264 o ISO/IEC MPEG-4 parte 10 y codificación de vídeo de alta eficiencia (HEVC), también conocida como UIT-T H.265 o MPEG-H parte 2. AVC incluye extensiones, tales como codificación de vídeo escalable (SVC), codificación de vídeo de vista múltiple (MVC) y codificación de vídeo 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 vídeo (JVET) de UIT-T e ISO/IEC ha comenzado a desarrollar un estándar de codificación de vídeo denominado codificación de vídeo versátil (VVC). VVC se incluye en un borrador de trabajo (WD), que incluye JVET-L1001-v5.

Con el fin de codificar una imagen de vídeo, primero se divide la imagen en particiones y las particiones se codifican en una corriente de bits. Se encuentran disponibles varios esquemas de división de imágenes en particiones. Por ejemplo, una imagen se puede dividir en segmentos regulares, segmentos dependientes, mosaicos y/o según el procesamiento paralelo del frente de onda (WPP). En aras de la simplicidad, HEVC limita los codificadores de manera que solo se puedan utilizar segmentos regulares, segmentos dependientes, mosaicos, WPP y combinaciones de los mismos al dividir un segmento en grupos de CTB para la codificación de vídeo. Dicha división en particiones se puede aplicar para soportar la coincidencia de tamaño de la unidad de transferencia máxima (MTU), el procesamiento en paralelo y el retardo reducido de una extremidad a otra. MTU indica 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 parte dividida en particiones de una imagen que se puede reconstruir independientemente de otros segmentos regulares dentro de la misma imagen, a pesar de algunas inter-dependencias debidas a las operaciones de filtrado en 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 intra-muestra, 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 segmento se asocia con un encabezado de segmento separado. El uso de segmentos regulares puede incurrir en una sobrecarga de codificación sustancial debido al coste 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 debería 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 división en particiones de los límites del bloque de árbol de la imagen sin romper la predicción en imagen. por consiguiente, los segmentos dependientes permiten que un segmento regular se fragmente en múltiples unidades NAL, lo que proporciona un retardo de extremidad a extremidad 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 dividida en particiones 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 CTB es local dentro de un mosaico. Por consiguiente, los CTB en un primer mosaico se codifican en orden de exploración de trama, antes de pasar a los CTB 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 utilizar 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 contiguos puede limitarse a transmitir un encabezado de segmento compartido (cuando los mosaicos adyacentes están en el mismo segmento), y realizar el intercambio relacionado con el filtrado en 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 señalizar en el encabezado del segmento. Para cada segmento y mosaico, se deberían 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 divide en filas individuales de CTB. Los mecanismos de decodificación y predicción por entropía pueden utilizar datos de CTB en otras filas. El procesamiento en paralelo es hecho posible mediante la decodificación en paralelo de filas de CTB. 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 CTB. Este retraso asegura que los datos relacionados con el CTB anterior y el CTB anterior 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 de bloques de árbol contiguas dentro de una imagen, la comunicación entre procesadores/entre núcleos para habilitar la predicción en imagen puede ser sustancial. La división en particiones WPP considera los tamaños de unidad NAL. Por lo tanto, WPP no soporta la coincidencia de tamaño de MTU. Sin embargo, los segmentos regulares se pueden utilizar 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 con limitación de movimiento. Un conjunto de mosaicos con limitación de movimiento (MCTS) es un conjunto de mosaicos diseñado de tal manera que los vectores de movimiento asociados están limitados 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 la corriente de bits y señalizar los MCTS. El mensaje SEI de MCTS proporciona información complementaria que se puede utilizar en la extracción de la sub-corriente de bits de MCTS (especificada como parte de la semántica del mensaje SEI) para generar una corriente de bits conforme para un MCTS. La información incluye varios conjuntos de información de extracción, cada uno de los cuales define un número 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 sub-corriente de bits de MCTS. Cuando se extrae un sub-corriente de bits según el proceso de extracción de la sub-corriente 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 la sub-corriente de bits extraída.

Los diversos esquemas de formación de mosaicos se pueden emplear al dividir una imagen en particiones para su codificación adicional. Como ejemplo particular, los mosaicos se pueden asignar a grupos de mosaicos, que pueden tomar el lugar de los segmentos en algunos ejemplos. En algunos ejemplos, cada grupo de mosaicos se puede extraer independientemente de otros grupos de mosaicos. Por consiguiente, la agrupación de mosaicos puede soportar la paralelización al permitir que cada grupo de mosaicos se asigne a un procesador diferente. La agrupación de mosaicos también se puede emplear en los casos donde un decodificador no desee decodificar una imagen completa. Como ejemplo particular, se pueden emplear esquemas de codificación de vídeo para soportar vídeo de realidad virtual (VR), que se puede codificar según el formato de aplicación de medios omnidireccional (OMAF).

En el vídeo de VR, una o más cámaras pueden grabar el entorno alrededor de la cámara o cámaras. A continuación, un usuario puede ver el vídeo de VR como si el usuario estuviera presente en la misma ubicación que la cámara. En el vídeo de VR, una imagen abarca todo un entorno alrededor del usuario. A continuación, el usuario ve una subparte de la imagen. Por ejemplo, un usuario puede emplear un dispositivo de visualización montado en la cabeza que cambia la subparte de la imagen mostrada basándose en los movimientos de la cabeza del usuario. La parte del vídeo que se muestra puede denominarse ventana gráfica ("viewport").

Por consiguiente, una característica distintiva del vídeo omnidireccional es que solo se muestra una ventana gráfica en un momento determinado. Esto contrasta con otras aplicaciones de vídeo que pueden mostrar un vídeo completo. Esta característica se puede utilizar para mejorar el rendimiento de los sistemas de vídeo omnidireccionales, por ejemplo, mediante la entrega selectiva dependiendo de la ventana gráfica del usuario (o cualquier otro criterio, tales como los metadatos cronometrados recomendados de la ventana gráfica). La entrega dependiente de la ventana gráfica puede habilitarse, por ejemplo, empleando empaquetamiento regional y/o codificación de vídeo dependiente de la ventana gráfica. La mejora del rendimiento puede resultar en un ancho de banda de transmisión más bajo, una menor complejidad de decodificación o ambas en comparación con otros sistemas de vídeo omnidireccionales cuando se emplea la misma resolución/calidad de vídeo.

Un ejemplo de operación dependiente de la ventana gráfica es un enfoque basado en MCTS para lograr una resolución de cinco mil muestras (por ejemplo, 5120 × 2560 muestras de luma) resolución (5K) de proyección equirrectangular efectiva (ERP) con perfil de vídeo OMAF dependiente de la ventana gráfica basada en HEVC. Este enfoque se describe con mayor detalle a continuación. Pero, en general, este enfoque divide el vídeo de VR en grupos de mosaicos y codifica el vídeo a una pluralidad de resoluciones. El decodificador puede indicar la ventana gráfica utilizada actualmente por el usuario durante la transmisión en directo. El servidor de vídeo que proporciona los datos de vídeo de VR puede, a continuación, reenviar el grupo o grupos de mosaicos asociados con la ventana gráfica en alta resolución y reenviar los grupos de mosaicos no vistos a una resolución más baja. Esto permite al usuario ver el vídeo de VR en alta resolución sin necesidad de enviar la imagen completa en alta resolución. Las subpartes no visualizadas se descartan y, por lo tanto, el usuario puede desconocer las resoluciones más bajas. Sin embargo, los grupos de mosaicos de resolución más baja pueden mostrarse al usuario si el usuario cambia las ventanas gráficas. La resolución de la nueva ventana gráfica se puede, a continuación, aumentar a medida que avanza el vídeo. Con el fin de implementar un sistema de este tipo, deberían crearse imágenes que contengan tanto los grupos de mosaicos de mayor resolución como los grupos de mosaicos de resolución más baja.

En otro ejemplo, las aplicaciones de videoconferencia pueden diseñarse para reenviar imágenes que incluyen múltiples resoluciones. Por ejemplo, una videoconferencia puede contener varios participantes. El participante que está hablando actualmente puede mostrarse a una resolución más alta y otros participantes pueden mostrarse a resoluciones más bajas. Con el fin de implementar un sistema de este tipo, deberían crearse imágenes que contengan tanto los grupos de mosaicos de mayor resolución como los grupos de mosaicos de resolución más baja.

En la presente memoria se dan a conocer varios mecanismos de formación de mosaicos flexibles para soportar la creación de una imagen con subimágenes codificadas a múltiples resoluciones. Por ejemplo, un vídeo se puede codificar a una pluralidad de resoluciones. El vídeo también se puede codificar empleando segmentos con cada resolución. Los segmentos de resolución más baja son más pequeños que los segmentos de resolución más alta. Con el fin de crear una imagen con múltiples resoluciones, la imagen se puede dividir en mosaicos de primer nivel. Los segmentos de la resolución más alta se pueden incluir directamente en los mosaicos de primer nivel. Además, los mosaicos de primer nivel se pueden dividir en mosaicos de segundo nivel que son más pequeños que los mosaicos de primer nivel. Por consiguiente, los mosaicos de segundo nivel más pequeños pueden aceptar directamente los segmentos de resolución más baja. De esta manera, los segmentos de cada resolución se pueden comprimir en una sola imagen a través de una relación de índice de mosaico sin requerir que mosaicos de diferente resolución se redirijan dinámicamente para utilizar un esquema de direccionamiento consistente. Los mosaicos de primer nivel y los mosaicos de segundo nivel pueden implementarse como MCTS y, por lo tanto, pueden aceptar datos de

imagen con limitación de movimiento a diferentes resoluciones. La presente descripción incluye muchos aspectos. Como ejemplo particular, los mosaicos de primer nivel se dividen en mosaicos de segundo nivel. Los mosaicos de segundo nivel están limitados, a continuación a que cada uno contenga un solo segmento rectangular de datos de imagen (por ejemplo, con la resolución más pequeña). Como se ha utilizado en la presente memoria, un mosaico es una parte dividida en particiones de una imagen creada por límites horizontales y verticales (por ejemplo, según columnas y filas). Un segmento rectangular es un segmento limitado a mantener una forma rectangular y, por lo tanto, se codifica basándose en los límites de la imagen horizontal y vertical. Por consiguiente, un segmento rectangular no se codifica basándose en un grupo de exploración de trama (que contiene CTU en una línea de izquierda a derecha y de arriba a abajo y puede que no mantenga una forma rectangular). Un segmento es una región espacialmente distinta de una imagen/fotograma que se codifica por separado de cualquier otra región en el mismo fotograma/imagen. En un aspecto adicional, los mosaicos de primer nivel y los mosaicos de segundo nivel se asignan a grupos de mosaicos. Los grupos de mosaicos empleados junto con los esquemas de formación de mosaicos flexibles están limitados a ser rectangulares en contraste con la exploración de trama. Por ejemplo, los mosaicos de primer nivel se incluyen en un grupo de mosaicos rectangulares y los mosaicos de segundo nivel correspondientes están limitados a formar parte del mismo grupo de mosaicos que los mosaicos de primer nivel de los que se dividen dichos mosaicos de segundo nivel. Este enfoque crea límites rectangulares en lugar de límites de exploración de trama, que proceden de izquierda a derecha y de arriba a abajo y generalmente no son rectangulares. Al limitar los grupos de mosaicos para que tengan una forma rectangular, los grupos de mosaicos dan como resultado formas que soportan la extracción y visualización de sub-imágenes. Por consiguiente, los grupos de mosaicos que contienen subimágenes con diferentes resoluciones tienen una forma natural para soportar la visualización simultánea en pantallas y/o la extracción separada para su uso en dispositivos de visualización montados en la cabeza.

La fig. 1 es un diagrama de flujo de un método 100 operativo ejemplar de codificación de una señal de vídeo. Específicamente, una señal de vídeo se codifica en un codificador. El proceso de codificación comprime la señal de vídeo empleando varios mecanismos para reducir el tamaño del archivo de vídeo. Un tamaño de archivo más pequeño permite que el archivo de vídeo comprimido se transmita hacia un usuario, al tiempo que reduce la sobrecarga de ancho de banda asociada. El decodificador, a continuación, decodifica el archivo de vídeo comprimido para reconstruir la señal de vídeo 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 vídeo.

En la etapa 101, la señal de vídeo se introduce en el codificador. Por ejemplo, la señal de vídeo puede ser un archivo de vídeo sin comprimir almacenado en la memoria. Como otro ejemplo, el archivo de vídeo puede ser capturado por un dispositivo de captura de vídeo, tal como una cámara de vídeo, y codificado para soportar la transmisión en vivo del vídeo. El archivo de vídeo puede incluir tanto un componente de audio como un componente de vídeo. El componente de vídeo 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 la presente memoria 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 soportar la visualización tridimensional.

En la etapa 103, el vídeo se divide en bloques. La división en particiones incluye subdividir los píxeles de cada fotograma en bloques cuadrados y/o rectangulares para la compresión. Por ejemplo, en codificación de vídeo de alta eficiencia (HEVC) (también conocida como H.265 y MPEG-H parte 2) el fotograma se puede dividir primero en unidades de codificación en árbol (CTU), que son bloques de un tamaño predefinido (por ejemplo, sesenta y cuatro píxeles por sesenta y cuatro píxeles). Las CTU contienen muestras de luma y croma. Pueden emplearse árboles de codificación para dividir las CTU en bloques y, a continuación, subdividir recursivamente los bloques hasta que se logren configuraciones que soportan 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. Por consiguiente, los mecanismos de división en particiones varían según el contenido de los fotogramas de vídeo.

En la etapa 105, se emplean varios mecanismos de compresión para comprimir los bloques de imagen divididos en particiones en la etapa 103. Por ejemplo, se puede emplear la inter-predicción y/o intra-predicción. La inter-predicción 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 fotogramas adyacentes. Específicamente, un objeto, tal como una mesa, puede permanecer en una posición constante en múltiples fotogramas. Por lo tanto, la mesa se describe una vez y los fotogramas adyacentes pueden hacer referencia de nuevo al fotograma de referencia. Pueden emplearse mecanismos de coincidencia de patrones para hacer coincidir objetos en múltiples fotogramas. Además, los objetos en movimiento se pueden representar a través de múltiples fotogramas, por ejemplo, debido al movimiento del objeto o al movimiento de la cámara. Como

ejemplo particular, un vídeo puede mostrar un automóvil que se mueve por la pantalla en múltiples fotogramas. 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 fotograma a las coordenadas del objeto en un fotograma de referencia. Como tal, la inter-predicción 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 intra-predicción codifica bloques en un fotograma común. La intra-predicción aprovecha el hecho de que los componentes de luma y croma tienden a agruparse en un fotograma. Por ejemplo, una zona de verde en una parte de un árbol tiende a colocarse adyacente a zonas de verde similares. La intra-predicción emplea múltiples modos de predicción direccional (por ejemplo, treinta y tres en HEVC), un modo plano y un modo de corriente continua (CC). Los modos direccionales indican que un bloque actual es similar/igual que las muestras de un bloque contiguo en una 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 basándose en los bloques contiguos 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 CC 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 contiguos asociados con las direcciones angulares de los modos de predicción direccional. Por consiguiente, los bloques de intra-predicción pueden representar bloques de imagen como varios valores de modo de predicción relacional en lugar de los valores reales. Además, los bloques de inter-predicción 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 transformadas a los bloques residuales para comprimir aún más el archivo.

En la etapa 107, se pueden aplicar varias técnicas de filtrado. En HEVC, los filtros se aplican según un esquema de filtrado en bucle. La predicción basada en bloques dada a conocer 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, a continuación, 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 en bucle adaptativo y filtros de desplazamiento adaptativo 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 manera que es menos probable que los artefactos creen artefactos adicionales en los bloques posteriores que se codifican basándose en los bloques de referencia reconstruidos.

Una vez que la señal de vídeo ha sido dividida en particiones, comprimida y filtrada, los datos resultantes se codifican en una corriente de bits en la etapa 109. La corriente de bits incluye los datos dados a conocer anteriormente, así como cualquier dato de señalización deseado para soportar la reconstrucción adecuada de la señal de vídeo en el decodificador. Por ejemplo, tales datos pueden incluir datos de división en particiones, datos de predicción, bloques residuales y varios indicadores que proporcionan instrucciones de codificación al decodificador. La corriente de bits puede almacenarse en la memoria para su transmisión hacia un decodificador cuando se solicite. La corriente de bits también se puede emitir y/o difundir de forma simultánea hacia una pluralidad de decodificadores. La creación de la corriente de bits es un proceso iterativo. Por consiguiente, las etapas 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 descripción, y no pretende limitar el proceso de codificación de vídeo a un orden en particular.

El decodificador recibe la corriente de bits y comienza el proceso de decodificación en la etapa 111. Específicamente, el decodificador emplea un esquema de decodificación por entropía para convertir la corriente de bits en la sintaxis y los datos de vídeo correspondientes. El decodificador emplea los datos de sintaxis de la corriente de bits para determinar las particiones para los fotogramas en la etapa 111. La división en particiones debería coincidir con los resultados de la división en bloques en la etapa 103. Ahora se describe la codificación/decodificación por entropía empleada en la etapa 111. El codificador hace muchas elecciones durante el proceso de compresión, tales como seleccionar esquemas de división en bloques entre varias opciones posibles basadas en el posicionamiento espacial de los valores en la imagen o imágenes de entrada. La señalización de las opciones exactas puede emplear una gran cantidad de contenedores. Como se utiliza en la presente memoria, 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, a continuación, 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, a continuación, 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 la etapa 113, el decodificador realiza la decodificación de bloques. Específicamente, el decodificador emplea transformadas inversas para generar bloques residuales. A continuación, el decodificador emplea los bloques residuales y los bloques de predicción correspondientes para reconstruir los bloques de imagen según la división en particiones. Los bloques de predicción pueden incluir tanto bloques de intra-predicción como bloques de inter-predicción como se generan en el codificador en la etapa 105. A continuación, los bloques de imágenes reconstruidos se colocan en fotogramas de una señal de vídeo reconstruida según los datos de división en particiones determinados en la etapa 111. La sintaxis para la etapa 113 también se puede señalizar en la corriente de bits mediante la codificación por entropía como se ha dado a conocer anteriormente.

En la etapa 115, se realiza el filtrado en los fotogramas de la señal de vídeo reconstruida de una manera similar a la etapa 107 en el codificador. Por ejemplo, se pueden aplicar filtros de supresión de ruido, filtros de desbloqueo, filtros en 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 emitir en un dispositivo de visualización en la etapa 117 para su visualización por un usuario final.

La fig. 2 es un diagrama esquemático de un de sistema 200 de codificación y decodificación (códec) ejemplar para codificación de vídeo. Específicamente, el sistema 200 de códec proporciona funcionalidad para soportar la implementación del método 100 operativo. El sistema 200 de códec está generalizado para representar los componentes empleados tanto en un codificador como en un decodificador. El sistema 200 de códec recibe y divide en particiones una señal de vídeo como se ha dado a conocer con respecto a las etapas 101 y 103 en el método 100 operativo, lo que da como resultado una señal 201 de vídeo dividida en particiones. A continuación, el sistema 200 de códec comprime la señal 201 de vídeo dividida en particiones en una corriente de bits codificada cuando actúa como un codificador como se da a conocer con respecto a las etapas 105, 107 y 109 en el método 100. Cuando actúa como un decodificador, el sistema 200 de códec genera una señal de vídeo de salida a partir de la corriente de bits como se da a conocer con respecto a las etapas 111, 113, 115 y 117 en el método 100 operativo. El sistema 200 de códec incluye un componente 211 de control de codificador general, un componente 213 de modificación de escala de transformada y cuantificación, un componente 215 de estimación intra-imagen, un componente 217 de predicción intra-imagen, un componente 219 de compensación de movimiento, un componente 221 de estimación de movimiento, un componente 229 de transformada inversa y de modificación de escala, un componente 227 de análisis de control de filtro, un componente 225 de filtros en bucle, un componente 223 de memoria intermedia de imágenes decodificadas y un componente 231 de codificación aritmética binaria adaptativa de contexto (CABAC) y de formato de encabezado. 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 200 de códec pueden estar presentes en el codificador. El decodificador puede incluir un subconjunto de los componentes del sistema 200 de códec. Por ejemplo, el decodificador puede incluir el componente 217 de predicción intra-imagen, el componente 219 de compensación de movimiento, el componente 229 de transformada inversa y de modificación de escala, el componente 225 de filtros en bucle y el componente 223 de memoria intermedia de imágenes decodificadas. Estos componentes se describen ahora.

La señal 201 de vídeo dividida en particiones es una secuencia de vídeo capturada que ha sido dividida 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, a continuación, 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 (CU) en algunos casos. Por ejemplo, una CU puede ser una sub-parte de una CTU que contiene un bloque de luma, un bloque o bloques de croma (Cr) de diferencia de rojo y un bloque o bloques 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 en particiones un nodo en dos, tres o cuatro nodos secundarios, respectivamente, de diferentes formas dependiendo de los modos de división empleados. La señal 201 de vídeo dividida en particiones se reenvía al componente 211 de control de codificador general, al componente 213 de modificación de escala de transformada y cuantificación, al componente 215 de estimación intra-imagen, al componente 227 de análisis de control de filtro y al componente 221 de estimación de movimiento para su compresión.

El componente 211 de control de codificador general está configurado para tomar decisiones relacionadas con la codificación de las imágenes de la secuencia de vídeo en la corriente de bits según las limitaciones de la

aplicación. Por ejemplo, el componente 211 de control de codificador general gestiona la optimización del tamaño de tasa de bits/corriente de bits frente a la calidad de reconstrucción. Dichas decisiones se pueden tomar basándose en la disponibilidad de espacio de almacenamiento/ancho de banda y las solicitudes de resolución de imagen. El componente 211 de control de codificador general también gestiona la utilización de la memoria intermedia a la luz de la velocidad de transmisión para mitigar los problemas de saturación y falta de ejecución de la memoria intermedia. Para gestionar estos problemas, el componente 211 de control de codificador general gestiona la división en particiones, la predicción y el filtrado de los otros componentes. Por ejemplo, el componente 211 de control de codificador general 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 211 de control de codificador general controla los otros componentes del sistema 200 de códec para equilibrar la calidad de reconstrucción de la señal de vídeo con las preocupaciones sobre la tasa de bits. El componente 211 de control de codificador general crea datos de control, que controlan el funcionamiento de los otros componentes. Los datos de control también se reenvían al componente de CABAC y de formato de encabezado 231 para ser codificados en la corriente de bits para señalizar parámetros para decodificar en el decodificador.

La señal 201 de vídeo dividida en particiones también se envía al componente 221 de estimación de movimiento y al componente 219 de compensación de movimiento para la inter-predicción. Un fotograma o segmento de la señal 201 de vídeo dividida en particiones puede dividirse en múltiples bloques de vídeo. El componente 221 de estimación de movimiento y el componente 219 de compensación de movimiento realizan una codificación inter-predictiva 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 200 de códec puede realizar múltiples pases de codificación, por ejemplo, para seleccionar un modo de codificación apropiado para cada bloque de datos de vídeo.

El componente 221 de estimación de movimiento y el componente 219 de compensación de movimiento pueden estar muy integrados, pero se ilustran por separado con fines conceptuales. La estimación de movimiento, realizada por el componente 221 de estimación de movimiento, es el proceso de generación de vectores de movimiento, que estiman el movimiento para bloques de vídeo. Un vector de movimiento, por ejemplo, puede indicar el desplazamiento de un 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 incluyendo una CTU, bloques de codificación en árbol (CTB) y CU. Por ejemplo, una CTU se puede dividir en CTB, que, a continuación, se pueden dividir en CB para su inclusión en las CU. Una CU puede codificarse como una unidad de predicción (PU) que contiene datos de predicción y/o una unidad de transformada (TU) que contiene datos residuales transformados para la CU. El componente 221 de estimación de movimiento genera vectores de movimiento, PU y TU utilizando un análisis de distorsión de tasa como parte de un proceso de optimización de distorsión de tasa. Por ejemplo, el componente 221 de estimación de movimiento 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 vídeo (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 200 de códec puede calcular valores para posiciones de píxeles sub-enteros de imágenes de referencia almacenadas en el componente 223 de memoria intermedia de imágenes decodificadas. Por ejemplo, el sistema 200 de códec de vídeo 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 221 de estimación de movimiento 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 221 de estimación de movimiento calcula un vector de movimiento para una PU de un bloque de vídeo en un segmento inter-codificado comparando la posición de la PU con la posición de un bloque predictivo de una imagen de referencia. El componente 221 de estimación de movimiento envía el vector de movimiento calculado como datos de movimiento al componente 231 de CABAC y de formato de encabezado para codificación y movimiento al componente 219 de compensación de movimiento.

La compensación de movimiento, realizada por el componente 219 de compensación de movimiento, puede implicar buscar o generar el bloque predictivo basándose en el vector de movimiento determinado por el componente 221 de estimación de movimiento. De nuevo, el componente 221 de estimación de movimiento y el componente 219 de compensación de movimiento pueden integrarse funcionalmente, en algunos ejemplos. Al recibir el vector de movimiento para la PU del bloque de vídeo actual, el componente 219 de compensación de movimiento 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 221 de estimación de movimiento realiza una estimación de movimiento con relación a los componentes de luma, y el componente 219 de compensación de movimiento utiliza vectores de movimiento calculados basándose en los componentes de luma tanto para los componentes de croma como para los componentes de luma. El bloque predictivo y el bloque residual se reenvían al componente 213 de modificación de escala de transformada y cuantificación.

La señal 201 de vídeo dividida en particiones también se envía al componente 215 de estimación intra-imagen y al componente 217 de predicción intra-imagen. Al igual que con el componente 221 de estimación de movimiento y el componente 219 de compensación de movimiento, el componente 215 de estimación intra-imagen y el componente 217 de predicción intra-imagen pueden estar muy integrados, pero se ilustran por separado con fines conceptuales. El componente 215 de estimación intra-imagen y el componente 217 de predicción intra-imagen predicen mediante intra-predicción un bloque actual en relación con los bloques en un fotograma actual, como alternativa a la inter-predicción realizada por el componente 221 de estimación de movimiento y el componente 219 de compensación de movimiento entre fotogramas, como se ha descrito anteriormente. En particular, el componente 215 de estimación intra-imagen determina un modo de intra-predicción a utilizar para codificar un bloque actual. En algunos ejemplos, el componente 215 de estimación intra-imagen selecciona un modo de intra-predicción apropiado para codificar un bloque actual a partir de múltiples modos de intra-predicción probados. Los modos de intra-predicción seleccionados se reenvían, a continuación, al componente 231 de CABAC y de formato de encabezado para su codificación.

Por ejemplo, el componente 215 de estimación intra-imagen calcula los valores de distorsión de tasa utilizando un análisis de distorsión de tasa para los diversos modos de intra-predicción probados y selecciona el modo de intra-predicción 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 ha sido 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 215 de estimación intra-imagen calcula las relaciones a partir de las distorsiones y tasas para los diversos bloques codificados para determinar qué modo de intra-predicción presenta el mejor valor de distorsión de tasa para el bloque. Además, el componente 215 de estimación intra-imagen 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 217 de predicción intra-imagen puede generar un bloque residual a partir del bloque predictivo basado en los modos de intra-predicción seleccionados determinados por el componente 215 de estimación intra-imagen cuando se implementa en un codificador o leer el bloque residual de la corriente 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 reenvía al componente 213 de modificación de escala de transformada y cuantificación. El componente 215 de estimación intra-imagen y el componente 217 de predicción intra-imagen pueden operar tanto en componentes de luma como de croma.

El componente 213 de modificación de escala de transformada y cuantificación está configurado para comprimir más el bloque residual. El componente 213 de modificación de escala de transformada y cuantificación 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 vídeo que comprende valores de coeficiente de transformadas residuales. También se podrían utilizar transformadas wavelet, transformadas de enteros, transformadas de subbandas u otros tipos de transformadas. La transformada puede convertir la información residual de un dominio de valor de píxel en un dominio de transformada, tal como un dominio de frecuencia. El componente 213 de modificación de escala de transformada y cuantificación 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 de manera que la información de frecuencia diferente se cuantifique en diferentes granularidades, lo que puede afectar la calidad visual final del vídeo reconstruido. El componente 213 de modificación de escala de transformada y cuantificación también está configurado para cuantificar los coeficientes de transformada 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 213 de modificación de escala de transformada y cuantificación puede, a continuación, realizar una exploración de la matriz que incluye los coeficientes de transformada cuantificados. Los coeficientes de transformada cuantificados se envían al componente 231 de CABAC y de formato de encabezado para ser codificados en la corriente de bits.

El componente 229 de transformada inversa y de modificación de escala aplica una operación inversa del componente 213 de modificación de escala de transformada y cuantificación para soportar la estimación del movimiento. El componente 229 de transformada inversa y de modificación de escala aplica escala inversa,

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 221 de estimación de movimiento y/o el componente 219 de compensación de movimiento 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/fotograma posterior. Los filtros se aplican a los bloques de referencia reconstruidos para mitigar los artefactos creados durante el escalado, la cuantificación y la transformada. De lo contrario, tales artefactos podrían causar una predicción inexacta (y crear artefactos adicionales) cuando se predicen bloques posteriores.

El componente 227 de análisis de control de filtro y el componente 225 de filtros en bucle aplican los filtros a los bloques residuales y/o a los bloques de imagen reconstruidos. Por ejemplo, el bloque residual transformado del componente 229 de transformada inversa y de modificación de escala puede combinarse con un bloque de predicción correspondiente del componente 217 de predicción intra-imagen y/o el componente 219 de compensación de movimiento 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 227 de análisis de control de filtro y el componente 225 de filtros en bucle 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 227 de análisis de control de filtro analiza los bloques de referencia reconstruidos para determinar dónde deberían aplicarse dichos filtros y establece los parámetros correspondientes. Dichos datos se reenvían al componente 231 de CABAC y de formato de encabezado como datos de control de filtro para la codificación. El componente 225 de filtros en bucle 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 en 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 223 de memoria intermedia de imágenes decodificadas para su uso posterior en la estimación de movimiento como se ha dado a conocer anteriormente. Cuando funciona como decodificador, el componente 223 de memoria intermedia de imágenes decodificadas almacena y reenvía los bloques reconstruidos y filtrados hacia un dispositivo de visualización como parte de una señal de vídeo de salida. El componente 223 de memoria intermedia de imágenes decodificadas puede ser cualquier dispositivo de memoria capaz de almacenar bloques de predicción, bloques residuales y/o bloques de imágenes reconstruidas.

El componente 231 de CABAC y de formato de encabezado recibe los datos de los diversos componentes del sistema 200 de códec y codifica dichos datos en una corriente de bits codificada para su transmisión hacia un decodificador. Específicamente, el componente 231 de CABAC y de formato de encabezado 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, incluyendo los datos de intra-predicción y de movimiento, así como los datos residuales en forma de datos de coeficientes de transformada cuantificados, están todos codificados en la corriente de bits. La corriente de bits final incluye toda la información deseada por el decodificador para reconstruir la señal 201 de vídeo dividida en particiones original. Dicha información también puede incluir tablas de índice de modo de intra-predicción (también denominadas tablas de asignación de palabras de código), definiciones de contextos de codificación para varios bloques, indicaciones de los modos de intra-predicción 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 particiones de intervalos de probabilidad (PIPE) u otra técnica de codificación por entropía. Después de la codificación por entropía, la corriente de bits codificada 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 300 de vídeo de ejemplar. Puede emplearse el codificador 300 de vídeo para implementar las funciones de codificación del sistema 200 de códec y/o implementar las etapas 101, 103, 105, 107 y/o 109 del método 100 operativo. El codificador 300 divide en particiones una señal de vídeo de entrada, dando como resultado una señal 301 de vídeo dividida en particiones, que es sustancialmente similar a la señal 201 de vídeo dividida en particiones. La señal 301 de vídeo dividida en particiones es, a continuación, comprimida y codificada en una corriente de bits por componentes del codificador 300.

Específicamente, la señal 301 de vídeo dividida en particiones se reenvía a un componente 317 de predicción intra-imagen para la intra-predicción. El componente 317 de predicción intra-imagen puede ser sustancialmente similar al componente 215 de estimación intra-imagen y al componente 217 de predicción

intra-imagen. La señal 301 de vídeo dividida en particiones también se reenvía a un componente 321 de compensación de movimiento para la inter-predicción basada en bloques de referencia en un componente 323 de memoria intermedia de imágenes decodificadas. El componente 321 de compensación de movimiento puede ser sustancialmente similar al componente 221 de estimación de movimiento y al componente 219 de compensación de movimiento. Los bloques de predicción y los bloques residuales del componente 317 de predicción intra-imagen y el componente 321 de compensación de movimiento se reenvían a un componente 313 de transformada y cuantificación para la transformada y cuantificación de los bloques residuales. El componente 313 de transformada y cuantificación puede ser sustancialmente similar al componente 213 de modificación de escala de transformada y cuantificación. Los bloques residuales transformados y cuantificados y los bloques de predicción correspondientes (junto con los datos de control asociados) se reenvían a un componente 331 de codificación por entropía para codificar en una corriente de bits. El componente 331 de codificación por entropía puede ser sustancialmente similar al componente 231 de CABAC y de formato de encabezado.

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

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

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

Los bloques residuales reconstruidos y/o los bloques de predicción se reenvían al componente 417 de predicción intra-imagen para su reconstrucción en bloques de imagen basados en operaciones de intra-predicción. El componente 417 de predicción intra-imagen puede ser similar al componente 215 de estimación intra-imagen y al componente 217 de predicción intra-imagen. Específicamente, el componente 417 de predicción intra-imagen 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 de intra-predicción. Los bloques de imagen intra-predichos reconstruidos y/o los bloques residuales y los datos de inter-predicción correspondientes se reenvían a un componente 423 de memoria intermedia de imágenes decodificadas a través de un componente 425 de filtros en bucle, que puede ser sustancialmente similar al componente 223 de memoria intermedia de imágenes decodificadas y al componente 225 de filtros en bucle, respectivamente. El componente 425 de filtros en bucle filtra los bloques de imágenes reconstruidas, los bloques residuales y/o los bloques de predicción, y dicha información se almacena en el componente 423 de memoria intermedia de imágenes. Los bloques de imágenes reconstruidas del componente 423 de memoria intermedia de imágenes decodificadas se reenvían a un componente 421 de compensación de movimiento para la inter-predicción. El componente 421 de compensación de movimiento puede ser sustancialmente similar al componente 221 de estimación de movimiento y/o al componente 219 de compensación de movimiento. Específicamente, el componente 421 de compensación de movimiento 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 reenviarse a través del componente 425 de

filtros en bucle al componente 423 de memoria intermedia de imágenes decodificadas. El componente 423 de memoria intermedia de imágenes decodificadas 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 un dispositivo de visualización como una señal de vídeo de salida reconstruida.

La fig. 5 es un diagrama esquemático que ilustra una corriente de bits 500 ejemplar que contiene una secuencia de vídeo codificada. Por ejemplo, la corriente de bits 500 se puede generar mediante un sistema 200 de códec y/o un codificador 300 para decodificar mediante un sistema 200 de códec y/o un decodificador 400. Como otro ejemplo, la corriente de bits 500 puede ser generada por un codificador en la etapa 109 del método 100 para su uso por un decodificador en la etapa 111.

La corriente de bits 500 incluye un conjunto de parámetros de secuencia (SPS) 510, una pluralidad de conjuntos de parámetros de imagen (PPS) 512, encabezados 514 de grupo de mosaicos y datos 520 de imagen. Un SPS 510 contiene datos de secuencia comunes a todas las imágenes de la secuencia de vídeo contenida en la corriente de bits 500. Dichos datos pueden incluir el tamaño de la imagen, la profundidad de bits, los parámetros de la herramienta de codificación, las limitaciones 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, desplazamientos, parámetros de herramientas de codificación específicas de imagen (por ejemplo, controles de filtro), etc. El encabezado 514 de grupo de mosaicos contiene parámetros que son específicos para cada grupo de mosaicos en una imagen. Por lo tanto, puede haber un encabezado 514 de grupo de mosaicos por grupo de mosaicos en la secuencia de vídeo. El encabezado 514 de grupo de mosaicos puede contener información del grupo de mosaicos, recuentos de orden de imágenes (POC), listas de imágenes de referencia, ponderaciones de predicción, puntos de entrada de mosaicos, parámetros de desbloqueo, etc. Debería observarse que algunos sistemas hacen referencia al encabezado 514 de grupo de mosaicos como un encabezado de segmento, y utilizan dicha información para soportar segmentos en lugar de grupos de mosaicos.

Los datos 520 de imagen contienen datos de vídeo codificados según la inter-predicción y/o intra-predicción, así como los correspondientes datos residuales transformados y cuantificados. Dichos datos 520 de imagen se clasifican según la división en particiones utilizada para dividir en particiones la imagen antes de la codificación. Por ejemplo, la imagen en los datos 520 de imagen se divide en mosaicos 523. Los mosaicos 523 se dividen además en unidades de codificación en árbol (CTU). Las CTU se dividen además en bloques de codificación basados en árboles de codificación. A continuación, los bloques de codificación se pueden codificar/decodificar según los mecanismos de predicción. Una imagen puede contener uno o más mosaicos 523.

Un mosaico 523 es una parte dividida en particiones de una imagen creada por límites horizontales y verticales. Los mosaicos 523 pueden ser rectangulares y/o cuadrados. Específicamente, un mosaico 523 incluye cuatro lados que están conectados en ángulos rectos. Los cuatro lados incluyen dos pares de lados paralelos. Además, los lados de un par de lados paralelos tienen la misma longitud. Como tal, un mosaico 523 puede tener cualquier forma rectangular, donde un cuadrado es un caso especial de un rectángulo donde los cuatro lados tienen la misma longitud. Se puede solicitar una imagen en filas y columnas de mosaicos 523. Una fila de mosaicos es un conjunto de mosaicos 523 colocados de manera horizontalmente adyacente para crear una línea continua desde el límite izquierdo al límite derecho de una imagen (o viceversa). Una columna de mosaicos es un conjunto de mosaicos 523 colocados de manera verticalmente adyacente para crear una línea continua desde el límite superior hasta el límite inferior de la imagen (o viceversa). Los mosaicos 523 pueden permitir o no la predicción basada en otros mosaicos 523, según el ejemplo. Cada mosaico 523 puede tener un índice de mosaico único en la imagen. Un índice de mosaico es un identificador numérico seleccionado de forma procedimental que se puede utilizar para distinguir un mosaico 523 de otro. Por ejemplo, los índices de mosaicos pueden aumentar numéricamente en el orden de exploración de trama. El orden de exploración de trama es de izquierda a derecha y de arriba a abajo. Debería observarse que, en algunos ejemplos, a los mosaicos 523 también se les pueden asignar identificadores de mosaico (ID). Un ID de mosaico es un identificador asignado que puede utilizarse para distinguir un mosaico 523 de otro. Los cálculos pueden emplear ID de mosaico en lugar de índices de mosaico en algunos ejemplos. Además, los ID de mosaico se pueden asignar para que tengan los mismos valores que los índices de mosaico en algunos ejemplos. Los índices e/o ID de mosaicos se pueden señalizar para indicar los grupos de mosaicos que contienen los mosaicos 523. Por ejemplo, los índices e/o ID de mosaico pueden emplearse para asignar datos de imagen asociados con un mosaico 523 a una posición adecuada para su visualización. Un grupo de mosaicos es un conjunto relacionado de mosaicos 523 que se puede extraer y codificar por separado, por ejemplo, para soportar la visualización de una región de interés y/o para soportar el procesamiento en paralelo. Los mosaicos 523 en un grupo de mosaicos se pueden codificar sin referencia a los mosaicos 523 fuera del grupo de mosaicos. Cada mosaico 523 puede asignarse a un grupo de mosaicos correspondiente y, por lo tanto, una imagen puede contener una pluralidad de grupos de mosaicos.

Las figs. 6A a 6E ilustran un mecanismo 600 ejemplar para crear una pista 610 de extracción para combinar subimágenes de múltiples resoluciones de diferentes corrientes de bits en una sola imagen para su uso en aplicaciones de realidad virtual (VR). Puede emplearse el mecanismo 600 para soportar un caso de uso ejemplar del método 100. Por ejemplo, el mecanismo 600 se puede emplear para generar una corriente de bits 500 para la transmisión desde un sistema 200 de códec y/o un codificador 300 hacia un sistema 200 de códec y/o un decodificador 400. Como ejemplo específico, el mecanismo 600 se puede emplear para utilizar junto con VR, OMAF, vídeo de trescientos sesenta grados, etc.

En VR, sólo se muestra una parte del vídeo al usuario. Por ejemplo, el vídeo de VR se puede filmar para incluir una esfera que rodea a un usuario. El usuario puede emplear un dispositivo de visualización montado en la cabeza (HMD) para ver el vídeo de VR. El usuario puede apuntar el HMD hacia una región de interés. La región de interés se muestra al usuario y se descartan otros datos de vídeo. De esta manera, un usuario ve solo una parte del vídeo de VR seleccionada por el usuario en cualquier instante. Este enfoque imita las percepciones del usuario y, por lo tanto, hace que el usuario experimente un entorno virtual de una manera que imita un entorno real. Uno de los problemas con este enfoque es que todo el vídeo de VR puede transmitirse al usuario, pero sólo se utiliza realmente una ventana gráfica actual del vídeo y el resto se descarta. Con el fin de aumentar la eficiencia de la señalización para las aplicaciones de transmisión en directo, la ventana gráfica actual del usuario se puede transmitir a una primera resolución más alta y otras ventanas gráficas se pueden transmitir a una segunda resolución más baja. De esta manera, las ventanas gráficas que probablemente se descartarán ocupan menos ancho de banda que la ventana o ventanas gráficas que probablemente verá el usuario. En el caso de que el usuario seleccione una nueva ventana gráfica, se puede mostrar el contenido de resolución más baja hasta que el decodificador pueda solicitar que se transmita una ventana gráfica actual diferente a la primera resolución más alta. Puede emplearse el mecanismo 600 para crear una pista 610 de extracción, como se muestra en la fig. 6E, para soportar esta funcionalidad. Una pista 610 de extracción es una pista de datos de imagen que encapsula una imagen con múltiples resoluciones para su uso como se describe anteriormente.

El mecanismo 600 codifica el mismo contenido de vídeo a una primera resolución 611 y a una segunda resolución 612, como se muestra en las figs. 6A y 6B, respectivamente. Como ejemplo específico, la primera resolución 611 puede ser de 5120 x 2560 muestras de luma y la segunda resolución 612 puede ser de 2560 x 1280 muestras de luma. Las imágenes del vídeo pueden dividirse en mosaicos 601 a la primera resolución 611 y mosaicos 603 a la segunda resolución 612, respectivamente. En el ejemplo que se muestra, los mosaicos 601 y 603 están divididos cada uno en una cuadrícula de 4x2. Además, se puede codificar un MCTS para cada posición de mosaico 601 y 603. Las imágenes a la primera resolución 611 y a la segunda resolución 612 dan como resultado cada una, una secuencia de MCTS que describe el vídeo a lo largo del tiempo a una resolución correspondiente. Cada secuencia de MCTS codificada se almacena como una pista de subimagen o una pista de mosaico. El mecanismo 600 puede, a continuación, utilizar las imágenes para crear segmentos para soportar la selección de MCTS adaptativa de la ventana gráfica. Por ejemplo, se considera cada rango de orientaciones de visualización que provocan una selección diferente de MCTS de alta y baja resolución. En el ejemplo ilustrado, se obtienen cuatro mosaicos 601 que contienen MCTS a la primera resolución 611 y cuatro mosaicos 603 que contienen MCTS a la segunda resolución 613.

El mecanismo 600 puede, a continuación, crear una pista 610 de extracción para cada posible selección de MCTS adaptable de ventana gráfica. Las figs. 6C y 6D ilustran un ejemplo de selección de MCTS adaptable de ventana gráfica. Específicamente, un conjunto de mosaicos 605 y 607 seleccionados se seleccionan a la primera resolución 611 y a la segunda resolución 612, respectivamente. Los mosaicos 605 y 607 seleccionados se ilustran en sombreado gris. En el ejemplo que se muestra, los mosaicos 605 seleccionados son los mosaicos 601 a la primera resolución 611 que se han de mostrar al usuario y los mosaicos 607 seleccionados son los mosaicos 603 a la segunda resolución 612 que probablemente se han de descartar pero se han de mantener para soportar la visualización en caso de que el usuario seleccione una nueva ventana gráfica. Los mosaicos 605 y 607 seleccionados se combinan, a continuación, en una sola imagen que contiene datos de imagen tanto a la primera resolución 611 como a la segunda resolución 612. Estas imágenes se combinan para crear una pista 610 de extracción. La fig. 6E ilustra una única imagen de una pista 610 de extracción correspondiente con fines ilustrativos. Como se muestra, la imagen en la pista 610 de extracción contiene los mosaicos 605 y 607 seleccionados de la primera resolución 611 y de la segunda resolución 612. Como se ha observado anteriormente, las figs. 6C-6E ilustran una selección de MCTS adaptativa de ventana gráfica única. Con el fin de permitir la selección del usuario de cualquier ventana gráfica, se debería crear una pista 610 de extracción para cada combinación posible de mosaicos 605 y 607 seleccionados.

En el ejemplo mostrado, cada selección de mosaicos 603 que encapsulan contenido de la corriente de bits la segunda resolución 612 contiene dos segmentos. Puede incluirse un RegionWisePackingBox en la pista 610 de extracción para crear una asignación entre la imagen empaquetada y una imagen proyectada del formato ERP. En el ejemplo presentado, las corrientes de bits resueltas de las pistas de extracción tienen una resolución de 3200X2560. Por consiguiente, un decodificador con capacidad para cuatro mil muestras (4K)

puede decodificar contenido donde la ventana gráfica se extrae de una corriente de bits codificada con una resolución de cinco mil muestras 5K (5120X2560).

Como se muestra, la pista 610 de extracción contiene dos filas de mosaicos 601 de alta resolución y cuatro filas de mosaicos 603 de baja resolución. Por consiguiente, la pista 610 de extracción contiene dos segmentos de contenido de alta resolución y cuatro segmentos de contenido de baja resolución. Es posible que la formación de mosaicos uniforme no soporte este caso de uso. La formación de mosaicos uniforme se define por un conjunto de columnas de mosaicos y un conjunto de filas de mosaicos. Las columnas de mosaicos se extienden desde la parte superior de una imagen hasta la parte inferior de la imagen. Del mismo modo, las filas de mosaicos se extienden desde la izquierda de la imagen hasta la derecha de la imagen. Si bien dicha estructura se puede definir simplemente, esta estructura no puede soportar de manera efectiva casos de uso avanzados, tales como el caso de uso descrito por el mecanismo 600. En el ejemplo mostrado, se emplean diferentes números de filas en diferentes secciones de la pista 610 de extracción. Si se emplea formación de mosaicos uniforme, los mosaicos en el lado derecho de la pista 610 de extracción deberían reescribirse para aceptar dos segmentos cada uno. Este enfoque es ineficiente y computacionalmente complejo.

La presente descripción incluye un esquema de formación de mosaicos flexible, como se describe a continuación, que no requiere que los mosaicos se reescriban para incluir diferentes números de segmentos. El esquema de formación de mosaicos flexible permite que un mosaico 601 contenga contenido a una primera resolución 611. El esquema de formación de mosaicos flexible también permite que un mosaico 601 se divida en mosaicos más pequeños que se pueden asignar cada uno directamente a los mosaicos 603 a una segunda resolución 612. Esta asignación directa es más eficiente ya que tal enfoque no requiere que los mosaicos se reescriban/cambien de dirección cuando se combinan diferentes resoluciones como se ha descrito anteriormente.

La fig. 7 ilustra una aplicación 700 de videoconferencia ejemplar que empalma imágenes de múltiples resoluciones de diferentes corrientes de bits en una sola imagen para su visualización. La aplicación 700 puede emplearse para soportar un caso de uso ejemplar del método 100. Por ejemplo, la aplicación 700 puede emplearse en un sistema 200 de códec y/o un decodificador 400 para mostrar contenido de vídeo desde la corriente de bits 500 desde un sistema 200 de códec y/o un codificador 300. La aplicación 700 de videoconferencia muestra una secuencia de vídeo a un usuario. La secuencia de vídeo contiene imágenes que muestran a un participante 701 que habla y otros participantes 703. El participante 701 que habla se muestra a una primera resolución más alta y los otros participantes 703 se muestran a una segunda resolución más baja. Con el fin de codificar una imagen de este tipo, la imagen debería contener una parte con una sola fila y una parte con tres filas. Para soportar un escenario de este tipo con una formación de mosaicos uniforme, la imagen se divide en un mosaico izquierdo y otro derecho. A continuación, el mosaico de la derecha se reescribe/cambia de dirección para incluir tres filas. Este cambio de dirección da como resultado una compresión y una penalización del rendimiento. El esquema de formación de mosaicos flexible descrito a continuación permite que un solo mosaico se divida en mosaicos más pequeños y asignarlo a mosaicos en corrientes de bits de subimagen asociadas con los otros participantes 703. De esta manera, el participante 701 que habla se puede asignar directamente en un mosaico de primer nivel y los otros participantes 703 se pueden asignar a mosaicos de segundo nivel divididos desde el primer mosaico sin tal reescritura/cambio de dirección.

Las figs. 8A a 8B son diagramas esquemáticos que ilustran un esquema 800 de formación de mosaicos de vídeo flexible ejemplar capaz de soportar múltiples mosaicos con diferentes resoluciones en la misma imagen. El esquema 800 de formación de mosaicos de vídeo flexible se puede emplear para soportar un mecanismo 600 de codificación y una aplicación 700 más eficientes. Por consiguiente, el esquema 800 de formación de mosaicos de vídeo flexible se puede emplear como parte del método 100. Además, el esquema 800 de formación de mosaicos de vídeo flexible puede ser empleado por un sistema 200 de códec, un codificador 300 y/o un decodificador 400. Los resultados del esquema 800 de formación de mosaicos de vídeo flexible pueden almacenarse en una corriente de bits 500 para su transmisión entre el codificador y el decodificador.

Como se muestra en la fig. 8A, la imagen (por ejemplo, fotograma, imagen, etc.) se puede dividir en mosaicos 801 de primer nivel, también conocidos como mosaicos de nivel uno. Como se muestra en la fig. 8B, los mosaicos 801 de primer nivel se pueden dividir en particiones selectivamente para crear mosaicos 803 de segundo nivel, también conocidos como mosaicos de nivel dos. Los mosaicos 801 de primer nivel y los mosaicos 803 de segundo nivel, a continuación, se pueden emplear para crear una imagen con subimágenes codificadas a varias resoluciones. Un mosaico 801 de primer nivel es un mosaico generado al dividir en particiones completamente una imagen en un conjunto de columnas y un conjunto de filas. Un mosaico 803 de segundo nivel es un mosaico generado al dividir en particiones un mosaico 801 de primer nivel.

Como se ha descrito anteriormente, en varios escenarios un vídeo se puede codificar a una pluralidad de resoluciones, por ejemplo, en VR y/o teleconferencia. El vídeo también se puede codificar empleando segmentos a cada resolución. Los segmentos de resolución más baja son más pequeños que los segmentos de mayor resolución. Con el fin de crear una imagen con múltiples resoluciones, la imagen se puede dividir en

mosaicos 801 de primer nivel. Los segmentos de la resolución más alta se pueden incluir directamente en los mosaicos 801 de primer nivel. Además, los mosaicos 801 de primer nivel se pueden dividir en mosaicos 803 de segundo nivel que son más pequeños que los mosaicos 801 de primer nivel. Por consiguiente, los mosaicos 803 de segundo nivel más pequeños pueden aceptar directamente los segmentos de resolución más baja. De esta manera, los segmentos de cada resolución se pueden comprimir en una sola imagen, por ejemplo, a través de una relación de índice de mosaicos, sin requerir que mosaicos de diferente resolución se cambien de dirección dinámicamente para utilizar un esquema de cambio de dirección consistente. Los mosaicos 801 de primer nivel y los mosaicos 803 de segundo nivel pueden implementarse como MCTS y, por lo tanto, pueden aceptar datos de imagen con limitación de movimiento a diferentes resoluciones.

Los mosaicos 801 de primer nivel y los mosaicos 803 de segundo nivel también se pueden asignar a los grupos de mosaicos 811, 812 y/o 813. Un grupo de mosaicos 811, 812 y/o 813 es una selección de mosaicos relacionados que están sujetos a un tratamiento similar durante la codificación y decodificación. Como ejemplo, los grupos de mosaicos 811, 812 y/o 813 se pueden emplear para almacenar diferentes subimágenes a diferentes resoluciones. Los diferentes grupos de mosaicos 811, 812 y/o 813 pueden tener diferentes parámetros y, por lo tanto, los grupos de mosaicos 811, 812 y/o 813 permiten que diferentes subimágenes sean tratadas de manera diferente por un codificador y/o un decodificador. Los grupos de mosaicos 811, 812 y/o 813 pueden limitarse a ser rectangulares, en lugar del orden de exploración de trama, para soportar la funcionalidad descrita en la presente memoria. Debería observarse que el cuadrado es un caso especial de un rectángulo y, por lo tanto, debería comprenderse que las formas rectangulares incluyen formas cuadradas. Como ejemplo particular, los mosaicos 801 de primer nivel pueden asignarse al grupo de mosaicos 811, 812 y/o 813 rectangulares. Los mosaicos 803 de segundo nivel generados al dividir los mosaicos 801 de primer nivel pueden, a continuación, asignarse cada uno al grupo de mosaicos 811, 812 y/o 813 rectangulares asociado con el correspondiente mosaico 801 de primer nivel a partir del cual se ha dividido el mosaico 803 de segundo nivel. Como se ha observado anteriormente, los grupos de mosaicos 811, 812 y/o 813 son rectangulares y no son de exploración de trama. El orden de exploración de trama avanza en orden de CTU de izquierda a derecha a través de una imagen hasta que se alcanza el lado derecho de la imagen. La exploración de trama, a continuación, se mueve hacia abajo a la siguiente línea de CTU y se mueve desde el lado izquierdo hacia el lado derecho de la imagen. Por ejemplo, un grupo de mosaicos de orden de exploración de trama no podría incluir sólo mosaicos 801 de primer nivel o sólo mosaicos 803 de segundo nivel como se muestra en la fig. 8B, ya que un orden de exploración de trama avanzaría a través de la imagen antes de moverse hacia abajo. Como tal, los grupos de mosaicos 811, 812 y/o 813 rectangulares permiten un tratamiento separado de los mosaicos 801 de primer nivel y sólo los mosaicos 803 de segundo nivel, que pueden contener subimágenes de diferente resolución.

La presente descripción incluye muchos aspectos. Como ejemplo particular, los mosaicos 801 de primer nivel se dividen en mosaicos 803 de segundo nivel. Los mosaicos 803 de segundo nivel pueden, a continuación, limitarse para que cada uno contenga un sólo segmento rectangular de datos de imagen (por ejemplo, a la resolución más baja). Un segmento rectangular es un segmento limitado para mantener una forma rectangular y, por lo tanto, se codifica basándose en los límites de la imagen horizontal y vertical. Por consiguiente, un segmento rectangular no se codifica basándose en un grupo de exploración de trama (que contiene CTU en una línea de izquierda a derecha y de arriba a abajo y puede que no mantenga una forma rectangular). Un segmento es una región espacialmente distinta de una imagen/fotograma que se codifica por separado de cualquier otra región en el mismo fotograma/imagen. En otro ejemplo, el mosaico 801 de primer nivel se puede dividir en dos o más mosaicos 803 de segundo nivel completos. En tal caso, un mosaico 801 de primer nivel no puede contener un mosaico 803 de segundo nivel parcial. En otro ejemplo, una configuración de los mosaicos 801 de primer nivel y los mosaicos 803 de segundo nivel se puede señalizar en un conjunto de parámetros en una corriente de bits, tal como un PPS asociado con una imagen dividida en particiones para crear los mosaicos. En un ejemplo, una indicación dividida, tal como una indicación, puede codificarse en un conjunto de parámetros para cada mosaico 801 de primer nivel. La indicación denota qué mosaicos 801 de primer nivel se dividen adicionalmente en mosaicos 803 de segundo nivel. En otro ejemplo, la configuración de los mosaicos 803 de segundo nivel se puede señalizar como varias columnas de mosaicos de segundo nivel y un número de filas de mosaicos de segundo nivel.

En otro ejemplo, los mosaicos 801 de primer nivel y los mosaicos 803 de segundo nivel pueden asignarse a grupos de mosaicos. Dichos grupos de mosaicos se pueden limitar de manera que todos los mosaicos en un grupo de mosaicos correspondiente estén limitados a cubrir una región rectangular de la imagen (por ejemplo, en contraste con la exploración de trama). Por ejemplo, algunos sistemas pueden añadir mosaicos a un grupo de mosaicos en orden de exploración de trama. Esto incluye añadir un mosaico inicial en una fila actual, proceder a añadir cada mosaico en la fila hasta que se alcance el límite de la imagen izquierda de la fila actual, continuar con el límite derecho de la siguiente fila y añadir cada mosaico en la siguiente fila, etc. hasta que se alcance un mosaico final. Este enfoque puede dar como resultado formas no rectangulares que se extienden a lo largo de la imagen. Tales formas pueden no ser útiles para crear imágenes con múltiples resoluciones como se describe en la presente memoria. En cambio, el presente ejemplo puede limitar los grupos de mosaicos de modo que cualquier mosaico 801 de primer nivel y/o mosaico 803 de segundo nivel se pueda añadir al grupo de mosaicos (por ejemplo, en cualquier orden), pero el grupo de mosaicos resultante

debe ser un rectángulo o cuadrado (por ejemplo, incluir cuatro lados conectados en ángulo recto). Esta limitación puede asegurar que los mosaicos 803 de segundo nivel divididos en particiones de un solo mosaico 801 de primer nivel no se coloquen en diferentes grupos de mosaicos.

En otro ejemplo, los datos que indican explícitamente un número de columnas de mosaicos de segundo nivel y un número de filas de mosaicos de segundo nivel se pueden omitir de una corriente de bits cuando el ancho de un mosaico de primer nivel es menos que el doble del umbral de ancho mínimo y la altura de un mosaico de primer nivel es menor que el doble del umbral de altura mínima. Esto se debe a que un mosaico 801 de primer nivel que cumpla tales condiciones no puede dividirse en más de una columna o una fila, respectivamente, y por lo tanto el decodificador puede inferir dicha información. En otro ejemplo, las indicaciones divididas que indican qué mosaicos 801 de primer nivel están divididos en mosaicos 803 de segundo nivel pueden omitirse de la corriente de bits para ciertos mosaicos 801 de primer nivel. Por ejemplo, tales datos se pueden omitir cuando el mosaico 801 de primer nivel tiene un ancho de mosaico de primer nivel que es menor que un umbral de ancho mínimo y una altura de mosaico de primer nivel que es menor que un umbral de altura mínima. Esto se debe a que un mosaico 801 de primer nivel que cumple tales condiciones es demasiado pequeño para dividirse en mosaicos 803 de segundo nivel y, por tanto, el decodificador puede inferir dicha información.

Como se ha descrito anteriormente, un esquema 800 de formación de mosaicos de vídeo flexible soporta la fusión de subimágenes de diferentes corrientes de bits en una imagen que contiene múltiples resoluciones. A continuación se describen varias realizaciones que soportan dichas funcionalidades. En general, esta descripción describe métodos para la señalización y codificación de mosaicos en la codificación de vídeo que dividen en particiones las imágenes de una manera que es más flexible que el esquema de formación de mosaicos en HEVC. Más específicamente, esta descripción describe algunos esquemas de formación de mosaicos en donde las columnas de mosaicos pueden no extenderse uniformemente desde la parte superior a la inferior de una imagen codificada e igualmente las filas de mosaicos pueden no extenderse uniformemente de izquierda a derecha de una imagen codificada.

Por ejemplo, basándose en un enfoque de formación de mosaicos HEVC, algunos mosaicos deberían dividirse adicionalmente en múltiples filas de mosaicos para soportar la funcionalidad descrita en las figs. 6A a 6E y 7. Además, dependiendo de cómo se coloquen los mosaicos, un mosaico debería dividirse aún más en columnas de mosaicos. Por ejemplo, en la fig. 7, los participantes del dos al cuatro pueden colocarse debajo del participante uno en algunos casos, lo que podría soportarse dividiendo un mosaico en columnas. Para satisfacer estos escenarios, un mosaico de primer nivel se puede dividir en filas de mosaicos y columnas de mosaicos de mosaicos de segundo nivel como se describe a continuación.

Por ejemplo, la estructura de mosaicos se puede relajar de la siguiente manera. No se requiere que los mosaicos en la misma imagen tengan un número particular de filas de mosaicos. Además, no se requiere que los mosaicos en la misma imagen sean un número particular de columnas de mosaicos. Para la señalización de mosaicos flexibles, se pueden seguir las siguientes etapas. Una estructura de mosaicos de primer nivel puede definirse mediante columnas de mosaicos y filas de mosaicos como se define en HEVC. Las columnas de mosaicos y las filas de mosaicos pueden ser de tamaño uniforme o no uniforme. Cada uno de estos mosaicos puede denominarse mosaico de primer nivel. Se puede señalizar un indicador para especificar si cada mosaico de primer nivel está dividido adicionalmente en una o más columnas de mosaicos y una o más filas de mosaicos o no. Si un mosaico de primer nivel se divide aún más, las columnas y filas de mosaicos pueden ser de tamaño uniforme o no uniforme. Los nuevos mosaicos resultantes de la división de mosaicos de primer nivel se denominan mosaicos de segundo nivel. La estructura de mosaicos flexible puede limitarse sólo a los mosaicos de segundo nivel y, por lo tanto, en algunos ejemplos no se permite una división adicional de ningún mosaico de segundo nivel. En otros ejemplos, se puede aplicar una división adicional de mosaicos de segundo nivel para crear mosaicos de nivel posterior de una manera similar a la creación de mosaicos de segundo nivel a partir de mosaicos de primer nivel.

Para simplificar, cuando un mosaico de primer nivel se divide en dos o más mosaicos de segundo nivel, la división siempre puede utilizar columnas de mosaicos de tamaño uniforme y filas de mosaicos uniformes. A continuación se describe la derivación de las ubicaciones, los tamaños, los índices y el orden de exploración de los mosaicos flexibles definidos por este enfoque. Por simplicidad, cuando se utiliza tal estructura de mosaicos flexible, un grupo de mosaicos puede estar limitado para incluir uno o más mosaicos completos de primer nivel. En este ejemplo, cuando un grupo de mosaicos contiene un mosaico de segundo nivel, todos los mosaicos de segundo nivel que se han originado a partir de la división del mismo mosaico de primer nivel deberían estar incluidos en el grupo de mosaicos. Además, se puede limitar que cuando se utiliza tal estructura de mosaicos flexible, un grupo de mosaicos contiene uno o más mosaicos y juntos todos los mosaicos pertenecen a un grupo de mosaicos que cubre una región rectangular de una imagen. En otro aspecto, cuando se utiliza dicha estructura de mosaicos flexible, un grupo de mosaicos contiene uno o más mosaicos de primer nivel y juntos todos los mosaicos pertenecen a un grupo de mosaicos que cubre una región rectangular de una imagen.

En un ejemplo, la señalización de mosaicos flexibles puede ser la siguiente. Un ancho mínimo de mosaico y una altura mínima de mosaico son valores definidos. Una estructura de mosaicos de primer nivel se puede definir mediante columnas de mosaicos y filas de mosaicos. Las columnas de mosaicos y las filas de mosaicos pueden ser de tamaño uniforme o no uniforme. Cada una de estos mosaicos puede denominarse mosaico de primer nivel. Se puede señalizar un indicador para especificar si cualquiera de los mosaicos de primer nivel puede dividirse aún más. Este indicador puede no estar presente cuando el ancho de cada mosaico de primer nivel no sea mayor al doble del ancho mínimo del mosaico, y la altura de cada mosaico de primer nivel no sea mayor al doble de la altura mínima del mosaico. Cuando no está presente, se infiere que el valor del indicador es igual a cero.

En un ejemplo, se aplica lo siguiente para cada mosaico de primer nivel. Se puede señalizar un indicador para especificar si un mosaico de primer nivel se divide en una o más columnas de mosaicos y una o más filas de mosaicos o no. La presencia del indicador se puede limitar de la siguiente manera. Si el ancho del mosaico de primer nivel es mayor que el ancho mínimo del mosaico o si la altura del mosaico de primer nivel es mayor que la altura mínima del mosaico, el indicador está presente/señalizado. De lo contrario, el indicador no está presente y se infiere que el valor del indicador es igual a cero, lo que indica que el mosaico de primer nivel no se divide más.

Si un mosaico de primer nivel se divide aún más, el número de columnas de mosaicos y el número de filas de mosaicos para esta división pueden señalizarse adicionalmente. Las columnas de mosaicos y las filas de mosaicos pueden ser de tamaño uniforme o no uniforme. Los mosaicos resultantes de la división de mosaicos de primer nivel se denominan mosaicos de segundo nivel. La presencia del número de columnas de mosaicos y el número de filas de mosaicos se puede limitar de la siguiente manera. Cuando el ancho del mosaico de primer nivel es menor que el doble del ancho mínimo del mosaico, el número de columnas de mosaicos puede no ser señalizado y el valor del número de columnas de mosaicos puede inferirse que es igual a uno. La señalización puede emplear un elemento de sintaxis a_minus1 de manera que el valor del elemento de sintaxis señalizado puede ser cero y el número de columnas de mosaicos es el valor del elemento de sintaxis más uno. Este enfoque puede comprimir aún más los datos de señalización. Cuando la altura del mosaico de primer nivel es menor que el doble de la altura mínima del mosaico, el número de filas de mosaicos no se puede señalizar y el valor del número de filas de mosaicos puede inferirse que es igual a cero. El valor del elemento de sintaxis señalizado puede ser cero y el número de filas de mosaicos puede ser el valor del elemento de sintaxis más uno para comprimir aún más los datos de señalización. Los mosaicos resultantes de la división de mosaicos de primer nivel se pueden denominar mosaicos de segundo nivel. La estructura de mosaicos flexible puede limitarse a mosaicos de segundo nivel únicamente, de manera que no se permite más división de mosaicos de segundo nivel. En otros ejemplos, se puede aplicar una división adicional de mosaicos de segundo nivel de manera similar a dividir un mosaico de primer nivel en mosaicos de segundo nivel.

En un ejemplo, la señalización de una estructura de mosaicos flexible puede ser como sigue. Cuando una imagen contiene más de un mosaico, se puede emplear una señal, tal como un indicador, en un conjunto de parámetros al que se hace referencia directa o indirectamente mediante un grupo de mosaicos correspondiente. El indicador puede especificar si una estructura de mosaicos correspondiente es una estructura de mosaicos uniforme o una estructura de mosaicos no uniforme (por ejemplo, una estructura de mosaicos flexible como se describe en la presente memoria). El indicador se puede llamar uniform_tile_structure_flag. Cuando uniform_tile_structure_flag es igual a uno, se emplea la señalización de una estructura de mosaicos uniforme de estilo HEVC, por ejemplo, señalizando num_tile_columns_minus1 y num_tile_rows_minus1 para indicar un solo nivel de mosaicos uniformes. Cuando uniform_tile_structure_flag es igual a cero, también se puede señalizar la siguiente información. El número de mosaicos en una imagen se puede señalizar mediante el elemento de sintaxis num_tiles_minus2, que indica que el número de mosaicos en la imagen (NumTilesInPic) es igual a num_tiles_minus2 + 2. Esto puede resultar en un ahorro de bits durante la señalización, ya que una imagen puede considerarse un mosaico por defecto. Para cada mosaico, excluyendo el último, se señalizan las direcciones del primer bloque de codificación (por ejemplo, CTU) y el último bloque de codificación del mosaico. La dirección de un bloque de codificación puede ser el índice del bloque en una imagen (por ejemplo, el índice de CTU en la imagen). Los elementos de sintaxis para tales bloques de codificación pueden ser tile_first_block_address[i] y tile_last_block_address[i]. Estos elementos de sintaxis pueden codificarse como ue(v) o u(v). Cuando los elementos de sintaxis se codifican como u(v), el número de bits utilizados para representar cada uno de los elementos de sintaxis es ceil(log2 (número máximo del bloque de codificación en una imagen)). Las direcciones del primer y último bloque de codificación del último mosaico pueden no estar señalizadas y, en cambio, pueden derivarse en base al tamaño de la imagen en muestras de luma y la agregación de todos los demás mosaicos en la imagen.

En un ejemplo, para cada mosaico, excluyendo el último, en lugar de señalizar las direcciones del primer y último bloque de codificación del mosaico, la dirección del primer bloque de codificación del mosaico, y se puede señalizar el ancho y la altura del mosaico. En otro ejemplo, para cada mosaico, excluyendo el último, en lugar de señalizar las direcciones del primer y último bloque de codificación del mosaico, el desplazamiento del punto superior izquierdo del mosaico con respecto al original de la imagen (por ejemplo, la parte superior

izquierda de la imagen) y el ancho y la altura del mosaico pueden indicarse. En otro ejemplo más, para cada mosaico, excluyendo el último, en lugar de señalizar las direcciones del primer y último bloque de codificación del mosaico, se puede señalizar la siguiente información. Se puede señalizar el ancho y la altura del mosaico. Además, es posible que no se señalice la ubicación de cada mosaico. En su lugar, se puede señalizar un indicador para especificar si colocar el mosaico inmediatamente a la derecha o inmediatamente debajo del mosaico anterior. Este indicador puede no estar presente si el mosaico solo puede estar a la derecha o solo puede estar debajo del mosaico anterior. El desplazamiento de la parte superior izquierda del primer mosaico siempre se puede establecer para que sea el origen/parte superior izquierda de la imagen (por ejemplo, x = cero e y = cero).

Para la eficiencia de la señalización, se puede señalizar un conjunto de tamaños de mosaicos únicos (por ejemplo, ancho y alto). Se puede hacer referencia a esta lista de tamaño de mosaico único mediante el índice del bucle que incluye la señalización de cada tamaño de mosaico. En algunos ejemplos, las ubicaciones y los tamaños de mosaicos como derivadas de la estructura de mosaicos señalizados deben limitar la partición para aseguran que no se produzcan espacios ni superposiciones entre los mosaicos.

Las siguientes limitaciones también pueden aplicarse. Es posible que se requiera que las formas de los mosaicos sean rectangulares (por ejemplo, no formas de exploración de trama). La unidad de mosaicos en una imagen cubrirá la imagen sin ningún espacio ni superposición entre los mosaicos. Cuando la decodificación se realiza con un solo núcleo, para la codificación de un bloque de codificación actual (por ejemplo, CTU) que no está en el borde izquierdo de una imagen, el bloque de codificación contiguo izquierdo se decodificará antes que el bloque de codificación actual. Cuando la decodificación se realiza con un solo núcleo, para la codificación de un bloque de codificación actual (por ejemplo, CTU) que no está en el borde superior de una imagen, se decodificará un bloque de codificación contiguo superior antes del bloque de codificación actual. Cuando dos mosaicos tienen índices de mosaicos que están uno al lado del otro (por ejemplo, tres idx y cuatro idx), se cumple una de las siguientes condiciones. Los dos mosaicos comparten un borde vertical y/o cuando el primer mosaico tiene una ubicación superior izquierda en (Xa, Ya) con el tamaño (Wa y Ha representan su ancho y alto) y cuando el segundo mosaico tiene una ubicación superior izquierda en (Xb, Yb), a continuación, Yb = Ya + Ha.

Las siguientes limitaciones también pueden aplicarse. Cuando un mosaico tiene más de un mosaico contiguo izquierdo, la altura del mosaico será igual a la suma de las alturas de todos sus mosaicos contiguos izquierdos. Cuando un mosaico tiene más de un mosaico contiguo derecho, la altura del mosaico será igual a la suma de las alturas de todos sus mosaicos contiguos izquierdos. Cuando un mosaico tiene más de un mosaico contiguo superior, el ancho del mosaico será igual a la suma de los anchos de todos sus mosaicos contiguos superiores. Cuando un mosaico tiene más de un mosaico contiguo inferior, el ancho del mosaico será igual a la suma de los anchos de todos sus mosaicos contiguos inferiores.

Lo siguiente es un ejemplo de realización específico de los aspectos mencionados anteriormente. El proceso de exploración de mosaicos y exploración de CTB puede ser el siguiente. La lista ColWidth[i] para i que va de 0 a num_levell_tile_columns_minus1, inclusive, especificando el ancho de la columna del i-ésimo mosaico de primer nivel en unidades de CTB, se puede derivar de la siguiente manera.

```
if( uniform_level1_tile_spacing_flag)
for( i = 0; i <= num_level1_tile_columns_minus1; i++)
ColWidth[i] = ( ( i + 1 ) * PicWidthInCtbsY ) /
(num_level1_tile_columns_minus1 + 1 ) –
( i * PicWidthInCtbsY) / (num_level1_tile_columns_minus1 + 1 )
else {
ColWidth[num_level1_tile_columns_minus1] = PicWidthInCtbsY (6-1)
for( i = 0; i < num_level1_tile_columns_minus1; i++ ) {
ColWidth[i] = tile_level1_column_width_minus1 [i] + 1
ColWidth[num_tile_level1_columns_minus1] -= ColWidth[i]
}
}
```

La lista RowHeight [j] para j que va de 0 a num_level1_tile_rows_minus1, inclusive, especificando la altura de la fila del j-ésimo mosaico en unidades de CTB, se puede derivar de la siguiente manera:

```
if( uniform_level1_tile_spacing_flag)
for(j = 0; j <= num_level1_tile_rows_minus1; j++)
RowHeight[j] = ( ( j + 1 ) * PicHeightInCtbsY ) / (
num_level1_tile_rows_minus1 + 1 ) –
(j * PicHeightInCtbsY ) / (num_level1_tile_rows_minus1 + 1 )
else {
RowHeight[num_level1_tile_rows_minus1] = PicHeightInCtbsY (6-2)
```

```
for(j = 0; j < num_level1_tile_rows_minus1; j++){
RowHeight[j] = tile_level1_row_height_minus1 [j] + 1
RowHeight[num_level1_tile_rows_minus1] -= RowHeight[j]
}
}
```

La lista colBd[i] para i que va de 0 a num_level1_tile_columns_minus1 + 1, inclusive, especificando la ubicación del límite de columna del i-ésimo mosaico en unidades de CTB, se puede derivar de la siguiente manera:

```
for( colBd[0] = 0, i = 0; i <= num_level1_tile_columns_minus1; i++)
colBd[i + 1] = colBd[i] + ColWidth[i] (6-3)
```

La lista rowBd[j] para j que va de 0 a num_level1_tile_rows_minus1 + 1, inclusive, especificando la ubicación del límite de fila del j-ésimo mosaico en unidades de CTB, se puede derivar de la siguiente manera:

```
for( rowBd[0] = 0, j = 0; j <= num_level1_tile_rows_minus1; j++)
rowBd[j + 1] = rowBd[j] + RowHeight[j] (6-4)
```

La variable NumTilesInPic, que especifica el número de mosaicos en una imagen con referencia al PPS, y las listas TileColBd[i], TileRowBd[i], TileWidth[i] y TileHeight[i] para i que va de 0 a NumTilesInPic - 1, inclusive, especificando la ubicación del límite de columna del i-ésimo mosaico en unidades de CTB, la ubicación del límite de fila del i-ésimo mosaico en unidades de CTB, el ancho de columna del i-ésimo mosaico en unidades de CTB y la altura de columna del i-ésimo mosaico en unidades de CTB, se puede derivar de la siguiente manera:

```
for ( tileIdx = 0, i = 0; i < NumLevel1Tiles; i++ ) {
tileX = i % ( num_level1_tile_columns_minus1 + 1 )
tileY = i / ( num_level1_tile_columns_minus1 + 1 )
if ( !level2_tile_split_flag[ i ]) { (6-5)
TileColBd[ tileIdx ] = colBd[ tileX ]
TileRowBd[ tileIdx ] = rowBd[ tileY ]
TileWidth[ tileIdx ] = ColWidth[ tileX ]
TileHeight[ tileIdx ] = RowHeight[ tileY ]
tileIdx++
} else {
for( k = 0; k <= num_level2_tile_columns_minus1 [ i ]; k++ )
colWidth2[ k ] = ( ( k + 1 ) * ColWidth[ tileX ] ) /
(num_level2_tile_columns_minus1[ i ] + 1 ) -
( k * ColWidth[ tileX ]) / (num_level2_tile_columns_minus1[ i ] + 1 )
for( k = 0; k <= num_level2_tile_rows_minus1[ i ]; k++ )
rowHeight2[ k] = ((k + 1 ) * RowHeight[ tileY ] ) /
(num_level2_tile_rows_minus1[ i + 1 ) -
( k * RowHeight[ tileY ] ) / (num_level2_tile_rows_minus1[ i + 1 )
for( colBd2[ 0 ] = 0, k = 0; k <= num_level2_tile_columns_minus1[ i ]; k++)
colBd2[ k + 1 ] = colBd2[ k ] + colWidth2[ k ]
for( rowBd2[ 0 ] = 0, k = 0; k <= num_level2_tile_rows_minus1[ i ]; k++)
rowBd2[ k + 1 ] = rowBd2[ k ] + rowHeight2[ k ]
numSplitTiles = (num_level2_tile_columns_minus1[ i ] + 1) *
(num_level2_tile_rows_minus1[ i ] + 1)
for( k = 0; k < numSplitTiles; k++ ) {
tileX2 = k % (num_level2_tile_columns_minus1[ i ] + 1)
tileY2 = k / (num_level2_tile_columns_minus1[ i ] + 1)
TileColBd[ tileIdx ] = colBd[ tileX ] + colBd2[ tileX2 ]
TileRowBd[ tileIdx ] = rowBd[ tileY ] + rowBd2[ tileY2 ]
TileWidth[ tileIdx ] = colWidth2[ tileX2 ]
TileHeight[ tileIdx ] = rowHeight2[ tileY2 ]
tileIdx++
}
}
}
NumTilesInPic = tileIdx
```

La lista CtbAddrRsToTs[ctbAddrRs] para ctbAddrRs que va de 0 a PicSizeInCtbsY - 1, inclusive, especificando la conversión de una dirección CTB en la exploración de trama CTB de una imagen a una dirección CTB en la exploración de mosaicos, se puede derivar de la siguiente manera:

```
for (ctbAddrRs = 0; ctbAddrRs < PicSizeInCtbsY; ctbAddrRs++) {
tbX = ctbAddrRs % PicWidthInCtbsY
tbY = ctbAddrRs / PicWidthInCtbsYtileFound = FALSE
for (tileIdx = NumTilesInPic - 1, i = 0; i < NumTilesInPic - 1 && !tileFound; i++ ) { (6-6)
tileFound = tbX < (TileColBd[i] + TileWidth[i]) && tbY < (TileRowBd[i] + TileHeight[i] )
if (tileFound)
tileIdx = i
}
CtbAddrRsToTs[ctbAddrRs] = 0
for( i = 0; i < tileIdx; i++ )
CtbAddrRsToTs[ctbAddrRs] += TileHeight[i] * TileWidth[i] CtbAddrRsToTs[ctbAddrRs] += ( tbY - TileRowBd[tileIdx] ) * TileWidth[tileIdx] + tbX - TileColBd[tileIdx]
}
```

La lista CtbAddrTsToRs[ctbAddrTs] para ctbAddrTs que van 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, se puede derivar de la siguiente manera.

```
for( ctbAddrRs = 0; ctbAddrRs < PicSizeInCtbsY; ctbAddrRs++) (6-7)
CtbAddrTsToRs[CtbAddrRsToTs[ctbAddrRs]] = ctbAddrRs
```

La lista TileId[ctbAddrTs] para ctbAddrTs que van desde cero hasta PicSizeInCtbsY - 1, inclusive, especificando la conversión de una dirección de CTB en el exploración de mosaicos a un ID de mosaico, se puede derivar de la siguiente manera.

```
for( i = 0, tileIdx = 0; i <= NumTilesInPic; i++, tileIdx++ )
for( y = TileRowBd[i]; y < TileRowBd[i + 1]; y++ ) (6-8)
for( x = TileColBd[i]; x < TileColBd[i + 1]; x++ )
TileId[CtbAddrRsToTs[y * PicWidthInCtbsY+ x]] = tileIdx
```

La lista NumCtusInTile [tileIdx] para tileIdx que va de cero a NumTilesInPic− 1, inclusive, especificando la conversión de un índice de mosaico al número de CTU en el mosaico, se puede derivar de la siguiente manera.

```
for( i = 0,tileIdx = 0; i < NumTilesInPic; i++, tileIdx++ ) (6-9)
NumCtusInTile[tileIdx] = TileColWidth[tileIdx] * TileRowHeight[tileIdx]
```

Una sintaxis de RBSP del conjunto de parámetros de imagen ejemplar es la siguiente.

pic_parameter_set_rbsp( ) {	Descriptor
pps_pic_parameter_set_id	ue(v)
pps_seq_parameter_set_id	ue(v)
transform_skip_enabled_flag	u(1)
single_tile_in_pic_flag	u(1)
if(!single_tile_in_pic_flag) {	
num_level1_tile_columns_minus1	ue(v)
num_level1_tile_rows_minus1	ue(v)
uniform_level1_tile_spacing_flag	u(1)
if(!uniform_level1_tile_spacing_flag) {	
for(i = 0; i < num_level1_tile_columns_minus1; i++)	
level1_tile_column_width_minus1[i]	ue(v)
for(i = 0; i < num_level1_tile_rows_minus1; i++)	
level1_tile_row_height_minus1[i]	ue(v)
}	
level2_tile_present_flag	u(1)
for(i = 0; level2_tile_present_flag && i < NumLevel1Tiles; i++) {	
level2_tile_split_flag[i]	u(1)
if(level2_tile_split_flag) {	
num_level2_tile_columns_minus1[i]	ue(v)
num_level2_tile_rows_minus1[i]	ue(v)
}	
}	
if(NumTilesInPic > 1)	

loop_filter_across_tiles_enabled_flag	u(1)
}	
rbsp_trailing_bits( )	
}	

La semántica de RBSP del conjunto de parámetros de imagen ejemplar es la siguiente. El num_level1_tile_columns_minus1 más 1 especifica el número de columnas de mosaico de nivel 1 que dividen en particiones la imagen. El num_level1_tile_columns_minus1 irá de cero a PicWidthInCtbsY - 1, inclusive. Cuando no está presente, se infiere que el valor de num_level1_tile_columns_minus1 es igual a cero. El Num_level1_tile_rows_minus1 más 1 especifica el número de filas de mosaicos de nivel uno que dividen en particiones la imagen. num_level1_tile_rows_minus1 irá de cero a PicHeightInCtbsY - 1, inclusive. Cuando no está presente, se infiere que el valor de num_level1_tile_rows_minus1 es igual a cero. La variable NumLevelITiles se establece igual a (num_level1_tile_columns_minus1 + 1) * (num_level1_tile_rows_minus1 + 1). Cuando single_tile_in_pic_flag es igual a cero, NumTilesInPic será mayor que uno. El Uniform_level1_tile_spacing_flag se establece igual a uno para especificar que los límites de las columnas de mosaicos de nivel 1 y los límites de las filas de mosaicos de nivel 1 se distribuyen uniformemente a lo largo de la imagen. El Uniform_level1_tile_spacing_flag es igual a cero para especificar que los límites de las columnas de mosaicos de nivel uno y los límites de las filas de mosaicos de nivel uno no se distribuyen uniformemente a lo largo de la imagen, sino que se señalizan explícitamente utilizando los elementos de sintaxis level1_tile_column_width_minus1[i] y level1_tile_row_height_minus1[i]. Cuando no está presente, se infiere que el valor de uniform_level1_tile_spacing_flag es igual a uno. El level1_tile_column_width_minus1[i] más 1 especifica el ancho de columna del i-ésimo mosaico de nivel uno en unidades de CTB. El level1_tile_row_height_minus1[i] más 1 especifica la altura de la fila del i-ésimo mosaico de nivel uno en unidades de CTB. El level2_tile_present_flag especifica que uno o más mosaicos de nivel uno se dividen en más mosaicos. El level2_tile_split_flag[i] más 1 especifica que el i-ésimo mosaico de nivel uno se divide en dos o más mosaicos. El num_level2_tile_columns_minus1[i] más 1 especifica el número de columnas de mosaico que dividen en particiones el i-ésimo mosaico. El num_level2_tile_columns_minus1[i] irá de cero a ColWidth[i], inclusive. Cuando no está presente, se infiere que el valor de num_level2_tile_columns_minus1[i] es igual a cero. El num_level2_tile_rows_minus1[i] más 1 especifica el número de filas de mosaicos que dividen el i-ésimo mosaico. El num_level2_tile_rows_minus1[i] irá de cero a RowHeight[i], inclusive. Cuando no está presente, se infiere que el valor de num_level2_tile_rows_minus1[i] es igual a cero.

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 0 a num_level1_tile_columns_minus1, inclusive, especificando el ancho de la columna del i-ésimo mosaico de nivel 1 en unidades de CTB; la lista RowHeight[j] para j que va de 0 a num_level1_tile_rows_minus1, inclusive, especificando la altura de la fila del j-ésimo mosaico de nivel 1 en unidades de CTB; la variable NumTilesInPic, que especifica el número de mosaicos en una imagen que se refiere al PPS; la lista TileWidth[i] para i que va de 0 a NumTilesInPic, inclusive, especificando el ancho del i-ésimo mosaico en unidades de CTB; la lista TileHeight[i] para i que va de 0 a NumTilesInPic, inclusive, especificando la altura del i-ésimo mosaico en unidades de CTB; la lista TileColBd[i] para i que va de 0 a NumTilesInPic, inclusive, especificando la ubicación del límite de columna del i-ésimo mosaico en unidades de CTB; la lista TileRowBd[i] para j que va de 0 a NumTilesInPic, inclusive, especificando la ubicación del límite de fila del i-ésimo mosaico en unidades de CTB; la lista CtbAddrRsToTs [ctbAddrRs] para ctbAddrRs que va desde 0 a 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 0 a 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 0 a 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 de 0 a PicSizeInCtbsY - 1, inclusive, especificando la conversión de un índice de mosaico al número de CTU en el mosaico; y la lista FirstCtbAddrTs[tileIdx] para tileIdx que va de 0 a NumTilesInPic - 1, inclusive, especificando la conversión de un ID de mosaico a la dirección CTB en la exploración de mosaico del primer CTB en el mosaico.

La semántica de encabezado del grupo de mosaicos ejemplar es la siguiente. Un tile_group_address especifica la dirección del mosaico del primer mosaico en el grupo de mosaicos, donde la dirección del mosaico es igual a TileId[firstCtbAddrTs] como se especifica en la ecuación 6-8, siendo firstCtbAddrTs la dirección CTB en la exploración de mosaicos de los CTB de la primera CTU en el grupo de mosaicos. La longitud de tile_group_address es Ceil(Log2(NumTilesInPic)) bits. El valor de tile_group_address irá de cero a NumTilesInPic - 1, inclusive, y el valor de tile_group_address no será igual al valor de tile_group_address de cualquier otra unidad NAL del grupo de mosaicos codificado de la misma imagen codificada. Cuando tile_group_address no está presente, se infiere que es igual a cero.

Lo siguiente es un segundo ejemplo de realización específica de los aspectos mencionados anteriormente. Un ejemplo de proceso de exploración de tramas y mosaicos de CTB es el siguiente. La variable NumTilesInPic, que especifica el número de mosaicos en una imagen con referencia al PPS, y las listas TileColBd[i],

TileRowBd[i], TileWidth[i] y TileHeight[i] para i que va de cero a NumTilesInPic - 1, inclusive, especificando la ubicación del límite de columna del i-ésimo mosaico en unidades de CTB, la ubicación del límite de fila del i-ésimo mosaico en unidades de CTB, el ancho de columna del i-ésimo mosaico en unidades de CTB y la altura de columna del i-ésimo mosaicos en unidades de CTB, se derivan de la siguiente manera.

```
for ( tileIdx = 0, i = 0; i < NumLevel1Tiles; i++ ) {
tileX = i % ( num_level1_tile_columns_minus1 + 1 )
tileY = i / ( num_level1_tile_columns_minus1 + 1 )
if ( !level2_tile_split_flag[i] ) {                                    (6-5)
    TileColBd[tileIdx] = colBd[tileX]
    TileRowBd[tileIdx] = rowBd[tileY]
    TileWidth[tileIdx] = ColWidth[tileX]
    TileHeight[tileIdx] = RowHeight[tileY]
    tileIdx++
} else {
    if ( uniform_level2_tile_spacing_flag[i] ) {
        for( k = 0; k <= num_level2_tile_columns_minus1[i]; k++ )
            colWidth2[k] = ( ( k + 1 ) * ColWidth[tileX] ) /
            ( num_level2_tile_columns_minus1[i] + 1 ) −
            ( k * ColWidth[tileX] ) / ( num_level2_tile_columns_minus1[i] + 1 )
        for( k = 0; k <= num_level2_tile_rows_minus1[i]; k++ )
            rowHeight2[k] = ( ( k + 1 ) * RowHeight[tileY] ) /
            ( num_level2_tile_rows_minus1[i] + 1 ) −
            ( k * RowHeight[tileY] ) / ( num_level2_tile_rows_minus1[i] + 1 )
    } else {
        colWidth2[num_level2_tile_columns_minus1[i]] = ColWidth[tileX] )
        for( k = 0; k <= num_level2_tile_columns_minus1[i]; k++ ) {
            colWidth2[k] = tile_level2_column_width_minus1[k] + 1
            colWidth2[k] −= colWidth2[k]
        }
        rowHeight2[num_level2_tile_rows_minus1[i]] = RowHeight[tileY] )
        for( k = 0; k <= num_level2_tile_rows_minus1[i]; k++ ) {
            rowHeigh2[k] = tile_level2_column_width_minus1[k] + 1
            rowHeight2[k] −= rowHeight2[k]
        }
    }
    for( colBd2[0] = 0, k = 0; k <= num_level2_tile_columns_minus1[i]; k++ )
        colBd2[k + 1] = colBd2[k] + colWidth2[k]
    for( rowBd2[0] = 0, k = 0; k <= num_level2_tile_rows_minus1[i]; k++ )
        rowBd2[k + 1] = rowBd2[k] + rowHeight2[k]
    numSplitTiles = (num_level2_tile_columns_minus1[i] + 1) * (num_level2_tile_rows_minus1[i] + 1)
    for( k = 0; k < numSplitTiles; k++ ) {
        tileX2 = k % (num_level2_tile_columns_minus1[i] + 1)
        tileY2 = k / (num_level2_tile_columns_minus1[i] + 1)
        TileColBd[tileIdx] = colBd[tileX] + colBd2[tileX2]
        TileRowBd[tileIdx] = rowBd[tileY] + rowBd2[tileY2]
        TileWidth[tileIdx] = colWidth2[tileX2]
        TileHeight[tileIdx] = rowHeight2[tileY2]
        tileIdx++
    }
}
}
NumTilesInPic = tileIdx
```

Una sintaxis de RBSP del conjunto de parámetros de imagen ejemplar es la siguiente.

pic_parameter_set_rbsp( ) {	Descriptor
pps_pic_parameter_set_id	ue(v)
pps_seq_parameter_set_id	ue(v)
transform_skip_enabled_flag	u(1)
single_tile_in_pic_flag	u(1)
if(!single_tile_in_pic_flag) {	
num_level1_tile_columns_minus1	ue(v)
num_level1_tile_rows_minus1	ue(v)
uniform_level1_tile_spacing_flag	u(1)

if(!uniform_level1_tile_spacing_flag) {	
for(i = 0; i < num_level1_tile_columns_minus1; i++)	
level1_tile_column_width_minus1[i]	ue(v)
for(i = 0; i < num_level1_tile_rows_minus1; i++)	
level1_tile_row_height_minus1[i]	ue(v)
}	
level2_tile_present_flag	u(1)
for(i = 0; level2_tile_present_flag && i < NumLevel1Tiles; i++) {	
level2_tile_split_flag[i]	u(1)
if(level2_tile_split_flag) {	
num_level2_tile_columns_minus1[i]	ue(v)
num_level2_tile_rows_minus1[i]	ue(v)
uniform_level2_tile_spacing_flag[i]	u(1)
if(!uniform_level2_tile_spacing_flag[i]) {	
for(j = 0; j < num_level2_tile_columns_minus1[i]; j++)	
level2_tile_column_width_minus 1 [j]	ue(v)
for(j = 0; j < num_level2_tile_rows_minus1[i]; j++)	
level2_tile_row_height_minus1[j]	ue(v)
}	
}	
}	
if(NumTilesInPic > 1)	
loop_filter_across_tiles_enabled_flag	u(1)
}	
rbsp_trailing_bits( )	
}	

La semántica de RBSP del conjunto de parámetros de imagen ejemplar es la siguiente. El uniform_level2_tile_spacing_flag[i] se establece igual a uno para especificar que los límites de la columna de mosaicos del nivel dos del i-ésimo mosaico de nivel uno y del mismo modo los límites de fila de mosaico del nivel dos del i-ésimo mosaico de nivel uno se distribuyen uniformemente a través de la imagen. El uniform_level2_tile_spacing_flag[i] se puede establecer igual a cero para especificar que los límites de la columna de mosaicos de nivel dos del i-ésimo mosaico de nivel uno y de la misma manera los límites de la fila de mosaicos de nivel dos de i-ésimo mosaico de nivel uno no se distribuyen uniformemente a través de la imagen, pero se señalizan explícitamente utilizando los elementos de sintaxis level2_tile_column_width_minus1[j] y level2_tile_row_height_minus1[j]. Cuando no está presente, se infiere que el valor de uniform_level2_tile_spacing_flag[i] es igual a uno. El level2_tile_column_width_minus1[j] más 1 especifica el ancho de la columna del j-ésimo mosaico de nivel dos del i-ésimo mosaico de nivel uno en unidades de CTB. El level2_tile_row_height_minus1[j] más 1 especifica la altura de la fila del j-ésimo mosaico de nivel dos del i-ésimo mosaico de nivel uno en unidades de CTB.

Lo siguiente es un tercer ejemplo de realización específica de los aspectos antes mencionados. Una sintaxis de RBSP del conjunto de parámetros de imagen ejemplar es la siguiente.

pic_parameter_set_rbsp( ) {	Descriptor
pps_pic_parameter_set_id	ue(v)
pps_seq_parameter_set_id	ue(v)
transform_skip_enabled_flag	u(1)
if ((PicWidthInCtbsY * CtbSizeY) >= (2 * MinTileWidth) || (PicHeightInCtbsY * CtbSizeY) >= (2 * MinTileHeight))	
single_tile_in_pic_flag	u(1)
if(!single_tile_in_pic_flag) {	
if (PicWidthInCtbsY * CtbSizeY >= (2 * MinTileWidth))	
num_level1_tile_columns_minus1	ue(v)
if (PicHeightInCtbsY * CtbSizeY >= (2 * MinTileHeight))	
num_level1_tile_rows_minus1	ue(v)
uniform_level1_tile_spacing_flag	u(1)
if(!uniform_level1_tile_spacing_flag) {	
for(i = 0; i < num_level1_tile_columns_minus1; i++)	
level1_tile_column_width_minus1[i]	ue(v)
for(i = 0; i < num_level1_tile_rows_minus1; i++)	
level1_tile_row_height_minus1[i]	ue(v)
}	

if(Level1TilesMayFurtherBeSplit)	
level2_tile_present_flag	u(1)
for(i = 0; level2_tile_present_flag && i < NumLevel1Tiles; i++) {	
if (ColWidth[i] * CtbSizeY > MinTileWidth || RowHeight[i] * CtbSizeY > MinTileHeight)	
level2_tile_split_flag[i]	u(1)
if(level2_tile_split_flag) {	
if (ColWidth[i] * CtbSizeY >= (2 * MinTileWidth))	
num_level2_tile_columns_minus1[i]	ue(v)
if (RowHeight[i] * CtbSizeY >= (2 * MinTileHeight))	
num_level2_tile_rows_minus1[i]	ue(v)
}	
}	
if (NumTilesInPic > 1)	
loop_filter_across_tiles_enabled_flag	u(1)
}	
rbsp_trailing_bits( )	
}	

Una semántica de RBSP del conjunto de parámetros de imagen ejemplar es la siguiente. La conformidad de la corriente de bits puede requerir que se apliquen las siguientes limitaciones. El valor MinTileWidth especifica el ancho mínimo de mosaico y será igual a doscientas cincuenta y seis muestras de luma. El valor MinTileHeight especifica la altura mínima del mosaico y será igual a sesenta y cuatro muestras de luma. El valor del ancho mínimo de mosaico y la altura mínima de mosaico puede cambiar según la definición del perfil y el nivel. La variable Level1TilesMayBeFurtherSplit puede derivarse como sigue:

```
Level1TilesMayBeFurtherSplit = 0
for ( i = 0, !Level1TilesMayBeFurtherSplit && i = 0; i < NumLevel1Tiles; i++ )
        if ( ( ColWidth[i] * CtbSizeY >= ( 2 * MinTileWidth ) ) | |
                                        ( RowHeight[i] * CtbSizeY >= ( 2 * MinTileHeight ) ))
                Level1TilesMayBeFurtherSplit = 1
```

Level2_tile_present_flag especifica que uno o más mosaicos de nivel se dividen en más mosaicos. Cuando no está presente, se infiere que el valor de level2_tile_present_flag es igual a cero. El level2_tile_split_flag[i] más 1 especifica que el i-ésimo mosaico de nivel uno se divide en dos o más mosaicos. Cuando no está presente, se infiere que el valor de level2_tile_split_flag[i] es igual a cero.

Lo que sigue es una cuarta realización ejemplar específica de los aspectos antes mencionados. Se puede señalizar la ubicación y el tamaño de cada mosaico. La sintaxis para soportar dicha señalización de estructura de mosaico se puede tabular a continuación. Tile_top_left_address[i] y tile_bottom_right_address[i] son el índice de CTU dentro de la imagen que indica el área rectangular cubierta por el mosaico. El número de bits para señalizar estos elementos de sintaxis debería ser suficiente para representar el número máximo de CTU en la imagen.

pic_parameter_set_rbsp( ) {	Descriptor
...	.
single_tile_in_pic_flag	u(1)
if(!single_tile_in_pic_flag) {	
tile_size_unit_idc	ue(v)
uniform_tile_flag	u(1)
if(uniform_tile_flag) {	
num_tile_columns_minus1	ue(v)
num_tile_rows_minus1	ue(v)
}	
else {	
num_tiles_minus2	ue(v)
for(i = 0; i < (num_tiles_minus2 + 2); i++) {	
tile_top_left_address[i]	u(v)
tile_bottom_right_address[i]	u(v)
}	
}	
loop_filter_across_tiles_enabled_flag	u(1)
}	

rbsp_trailing_bits( )	
}	

Se puede señalizar la ubicación y el tamaño de cada mosaico. La sintaxis para soportar dicha señalización de estructura de mosaico se puede tabular a continuación. Tile_top_left_address[i] es el índice de CTU de la primera CTU en el mosaico en el orden de una exploración de trama de CTU de una imagen. El ancho y la altura del mosaico especifican el tamaño del mosaico. Se pueden guardar algunos bits al señalizar estos dos elementos de sintaxis señalizando primero la unidad de tamaño de mosaico común.

pic_parameter_set_rbsp( ) {	Descriptor
...	.
single_tile_in_pic_flag	u(1)
if(!single_tile_in_pic_flag) {	
tile_size_unit_idc	ue(v)
uniform_tile_flag	u(1)
if(uniform_tile_flag) {	
num_tile_columns_minus1	ue(v)
num_tile_rows_minus1	ue(v)
}	
else {	
num_tiles_minus2	ue(v)
for(i = 0; i < (num_tiles_minus2 + 2); i++) {	
tile_top_left_address[i]	u(v)
tile_width_minus1[i]	ue(v)
tile_height_minus1[i]	ue(v)
}	
}	
loop_filter_across_tiles_enabled_flag	u(1)
}	
rbsp_trailing_bits( )	
}	

Alternativamente, la señalización puede ser la siguiente.

pic_parameter_set_rbsp( ) {	Descriptor
...	.
single_tile_in_pic_flag	u(1)
if(!single_tile_in_pic_flag) {	
tile_size_unit_idc	ue(v)
uniform_tile_flag	u(1)
if(uniform_tile_flag) {	
num_tile_columns_minus1	ue(v)
num_tile_rows_minus1	ue(v)
}	
else {	
num_tiles_minus2	ue(v)
for(i = 0; i < (num_tiles_minus2 + 2); i++) {	
tile_x_offset[i]	ue(v)
tile_y_offset[i]	ue(v)
tile_width_minus1[i]	ue(v)
tile_height_minus1[i]	ue(v)
}	
}	
loop_filter_across_tiles_enabled_flag	u(1)
}	
rbsp_trailing_bits( )	
}	

En otro ejemplo, cada tamaño de mosaico se puede señalizar de la siguiente manera. Para señalizar una estructura de mosaico flexible, no se puede señalizar la ubicación de cada mosaico. En su lugar, se puede señalizar un indicador para especificar si se debe colocar el mosaico inmediatamente a la derecha o inmediatamente debajo del mosaico anterior. Este indicador puede no estar presente si el mosaico solo puede estar a la derecha o solo puede estar debajo del mosaico actual.

Los valores de tile_x_offset[i] y tile_y_offset[i] se pueden derivar mediante las siguientes etapas ordenadas.

tile_x_offset[0] y tile_y_offset[0] se establecen en 0.

maxWidth se establece igual a tile_width[0] y maxHeight se establece igual a tile_height[0]

runningWidth se establece igual a tile_width[0] y runningHeight se establece igual a tile_height[0]

lastNewRowHeight se establece a 0

TilePositionCannotBeInferred = falso

Para i> 0, se aplica lo siguiente:

Sea el valor isRight establecido de la siguiente manera:

si runningWidth + tile_width[i] <= PictureWidth, entonces isRight = = 1

si no, isRight = = 0

Sea el valor isBelow establecido de la siguiente manera:

si runningHeight + tile_height [i] <= PictureHeight, entonces isBelow = = 1

si no, isBelow = = 0

Si isRight = = 1 && isBelow = = 1 entonces TilePositionCannotBeInferred = verdadero

Si isRight = = 1 && isBelow = = 0, se aplica lo siguiente:

right_tile_flag[i] = 1

tile_x_offset[i] = runningWidth.

tile_y_offset[i] = (runningWidth == maxWidth) ? 0 : lastNewRowHeight

lastNewRowHeight = (runningWidth == maxWidth) ? 0 : lastNewRowHeight

de lo contrario, si isRight = = 0 && isBelow = = 1, se aplica lo siguiente:

right_tile_flag[i] = 0

tile_y_offset[i] = runningHeight

tile_x_offset[i] = (runningHeight == maxHeight) ? 0 : tile_x_offset[i – 1]

lastNewRowHeight = (runningHeight == maxHeight && runningWidth == maxWidth) ? runningHeight : lastNewRowHeight

de lo contrario, si isRight = = 1 && isBelow = = 1 && right_tile_flag [i] = = 1, se aplica lo siguiente:

tile_x_offset[i] = runningWidth.

tile_y_offset[i] = (runningWidth == maxWidth) ? 0 : lastNewRowHeight

lastNewRowHeight = (runningWidth == maxWidth) ? 0 : lastNewRowHeight

de lo contrario (es decir, isRight = = 1 && isBelow = = 1 && right_tile_flag [i] = = 0), se aplica lo siguiente:

tile_y_offset[i] = runningHeight

tile_x_offset[i] = (runningHeight == maxHeight) ? 0 : tile_x_offset[i – 1]

lastNewRowHeight = (runningHeight == maxHeight && runningWidth == maxWidth) ? runningHeight : lastNewRowHeight

si right_tile_flag[i] = = 1, se aplica lo siguiente:

runningWidth = runningWidth + tile_width[i]

si runningWidth> maxWidth, entonces establece maxWidth igual a runningWidth

runningHeight es igual a tile_y_offset [i] tile_height[i]

de lo contrario (es decir, right_tile_flag [i] = = 0), se aplica lo siguiente:

runningHeight = runningHeight + tile_height[i]

si runningHeight> maxHeight, establezca maxHeight igual a runningHeight

runningWidth es igual a tile_x_offset[i] + tile_width[i]

Lo anterior se puede describir en pseudocódigo como sigue.

```
tile_x_offset[0] = 0
tile_y_offset[0] = 0
maxWidth = tile_width[0]
maxHeight = tile_height[0]
runningWidth = tile_width[0]
runningHeight = tile_height[0]
lastNewRowHeight = 0
isRight = false
isBelow = false
TilePositionCannotBeInferred = false
for( i = 1; i < num_tiles_minus2 + 2; i++ ) {
        TilePositionCannotBeInferred = false
        isRight = ( runningWidth + tile_width[i] <= PictureWidth ) ? true : false
        isbelow = ( runningHeight + tile_height[i] <= PictureHeight ) ? true : false
                if (!isRight && !isBelow)
                //Error. ¡Este caso no sucederá!
        if (isRight && isBelow)
                TilePositionCannotBeInferred = true
                if (isRight && !isBelow) {
                right_tile_flag[i] = true
                tile_x_offst[i] = runningWidth
                tile_y_offset[i] = (runningWidth == maxWidth) ? 0 : lastNewRowHeight
                lastNewRowHeight = tile_y_offset[i]
        }
        else if (!isRight && isBelow) {
                right_tile_flag[i] = false
                tile_y_offset[i] = runningHeight
                tile_x_offset[i] = (runningHeight == maxHeight) ? 0 : tile_x_offset[i – 1]
                lastNewRowHeight = (runningHeight == maxHeight && runningWidth == maxWidth) ?
                                        runningHeight : lastNewRowHeight
        }
        else if ( right_tile_flag[i] ) {
                tile_x_offst[i] = runningWidth
                tile_y_offset[i] = (runningWidth == maxWidth) ? 0 : lastNewRowHeight
                lastNewRowHeight = tile_y_offset[i]
        }
        else {
                tile_y_offset[i] = runningHeight
                tile_x_offset[i] = (runningHeight == maxHeight) ? 0 : tile_x_offset[i – 1]
                lastNewRowHeight = (runningHeight == maxHeight && runningWidth == maxWidth) ?
                                        runningHeight : lastNewRowHeight
        }
}
if ( right_tile_flag[i] ) {
        runningWidth += tile_width[i]
        if ( runningWidth > maxWidth ) maxWidth = runningWidth
        runningHeight = tile_y_offset[i] + tile_height[i]
```

```
}
else {
        runningHeight += tile_height[i]
        if ( runningHeight > maxHeight ) maxHeight = runningHeight
        runningWidth = tile_x_offset[i] + tile_width[i]
}
```

pic_parameter_set_rbsp( ) {	Descriptor
...	.
single_tile_in_pic_flag	u(1)
if(!single_tile_in_pic_flag) {	
tile_size_unit_idc	ue(v)
uniform_tile_flag	u(1)
if(uniform_tile_flag) {	
num_tile_columns_minus1	ue(v)
num_tile_rows_minus1	ue(v)
}	
else {	
num_tiles_minus2	ue(v)
for(i = 0; i < (num_tiles_minus2 + 2); i++) {	
tile_width_minus1 [i]	ue(v)
tile_height_minus1 [i]	ue(v)
if(TilePositionCannotBeInferred)	
right_tile_flag[i]	u(1)
}	
}	
loop_filter_across_tiles_enabled_flag	u(1)
}	
rbsp_trailing_bits( )	
}	

La siguiente es una implementación para derivar el tamaño del último mosaico en pseudocódigo.

```
tile_x_offset[0] = 0
tile_y_offset[0] = 0
maxWidth = tile_width[0]
maxHeight = tile_height[0]
runningWidth = tile_width[0]
runningHeight = tile_height[0]
lastNewRowHeight = 0
isRight = false
isBelow = false
TilePositionCannotBeInferred = false
for( i = 1; i < num_tiles_minus2 + 2; i++ ) {
        currentTileWidth = ( i == num_tiles_minus2 + 1 ) ? (PictureWidth – runningWidth) % PictureWidth
                                                : tile_width[i]
        currentTileHeight = ( i == num_tiles_minus2 + 1 ) ? (PictureHeight – runningHeight) % PictureHeight
                                                : tile_Height[i]

        isRight = ( runningWidth + currentTileWidth <= PictureWidth ) ? true : false
        isbelow = ( runningHeight + currentTileHeight <= PictureHeight ) ? true : false

        if (!isRight && !isBelow)
                //Error. ¡Este caso no sucederá!
        if (isRight && isBelow)
                TilePositionCannotBeInferred = true

        if (isRight && !isBelow) {
                right_tile_flag[i] = true
                tile_x_offst[i] = runningWidth
                tile_y_offset[i] = (runningWidth == maxWidth) ? 0 : lastNewRowHeight
                lastNewRowHeight = tile_y_offset[i]
        }
        else if (!isRight && isBelow) {
                right_tile_flag[i] = false
```

```
                tile_y_offset[i] = runningHeight
                tile_x_offset[i] = (runningHeight == maxHeight) ? 0 : tile_x_offset[i – 1]
                lastNewRowHeight = (runningHeight == maxHeight && runningWidth == maxWidth) ?
                                        runningHeight : lastNewRowHeight
        }
        else if ( right_tile_flag[i] ) {
                tile_x_offst[i] = runningWidth
                tile_y_offset[i] = (runningWidth == maxWidth) ? 0 : lastNewRowHeight
                lastNewRowHeight = tile_y_offset[i]
        }
        else {
                tile_y_offset[i] = runningHeight
                tile_x_offset[i] = (runningHeight == maxHeight) ? 0 : tile_x_offset[i – 1]
                lastNewRowHeight = (runningHeight == maxHeight && runningWidth == maxWidth) ?
                                        runningHeight : lastNewRowHeight
        }
}
if ( right_tile_flag[i] ) {
        runningWidth += currentTileWidth
        if ( runningWidth > maxWidth ) maxWidth = runningWidth
        runningHeight = tile_y_offset[i] + currentTileHeight
}
else {
        runningHeight += currentTileHeight
        if ( runningHeight > maxHeight ) maxHeight = runningHeight
        runningWidth = tile_x_offset[i] + currentTileWidth
}
```

pic_parameter_set_rbsp( ) {	Descriptor
...	.
single_tile_in_pic_flag	u(1)
if(!single_tile_in_pic_flag) {	
tile_size_unit_idc	ue(v)
uniform_tile_flag	u(1)
if(uniform_tile_flag) {	
num_tile_columns_minus1	ue(v)
num_tile_rows_minus1	ue(v)
}	
else {	
num_tiles_minus2	ue(v)
for(i = 0; i < (num_tiles_minus2 + 1); i++) {	
tile_width_minus1[i]	ue(v)
tile_height_minus1[i]	ue(v)
if(TilePositionCannotBeInferred)	
right_tile_flag[i]	u(1)
}	
}	
loop_filter_across_tiles_enabled_flag	u(1)
}	
rbsp_trailing_bits( )	
}	

Para un mayor ahorro de bits de señalización, se puede señalizar el número de tamaños de mosaico únicos para soportar la tabulación del tamaño de mosaico unitario. El tamaño del mosaico puede ser referenciado a continuación solamente por un índice.

pic_parameter_set_rbsp( ) {	Descriptor
...	.
single_tile_in_pic_flag	u(1)
if(!single_tile_in_pic_flag) {	
tile_size_unit_idc	ue(v)
uniform_tile_flag	u(1)
if(uniform_tile_flag) {	
num_tile_columns_minus1	ue(v)

num_tile_rows_minus1	ue(v)
}	
else {	
num_tiles_minus2	ue(v)
num_unique_tile_sizes	ue(v)
for(i = 0; i < num_unique_tile_sizes; i++) {	
preset_tile_width_minus_1[i]	ue(v)
preset_tile_height_minus_1[i]	ue(v)
}	
for(i = 0; i < (num_tiles_minus2 + 2); i++) {	
if (num_unique_tile_sizes)	
tile_size_idx[i]	u(v)
else {	
tile_width_minus1[i]	ue(v)
tile_height_minus1[i]	ue(v)
}	
if(TilePositionCannotBeInferred)	
right_tile_flag[i]	u(1)
}	
}	
loop_filter_across_tiles_enabled_flag	u(1)
}	
rbsp_trailing_bits( )	
}	

La fig. 9 es un diagrama esquemático de un dispositivo 900 de codificación de vídeo ejemplar. El dispositivo 900 de codificación de vídeo es adecuado para implementar las realizaciones/ejemplos descritos en la presente memoria. El dispositivo 900 de codificación de vídeo comprende puertos 920 aguas abajo, puertos 950 aguas arriba y/o unidades 910 transceptoras (Tx/Rx), que incluyen transmisores y/o receptores para comunicar datos aguas arriba y/o aguas abajo a través de una red. El dispositivo 900 de codificación de vídeo 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 900 de codificación de vídeo también puede comprender componentes eléctricos, ópticos a eléctricos (OE), eléctricos a ópticos (EO) y/o componentes de comunicación inalámbrica acoplados a los puertos 950 aguas arriba y/o 920 aguas abajo para la comunicación de datos a través de redes de comunicación eléctricas, ópticas o inalámbricas. El dispositivo 900 de codificación de vídeo también puede incluir dispositivos 960 de entrada y/o salida (E/S) para comunicar datos hacia y desde un usuario. Los dispositivos 960 de E/S pueden incluir dispositivos de salida tales como un dispositivo de visualización para mostrar datos de vídeo, altavoces para emitir datos de audio, etc. Los dispositivos 960 de E/S también pueden incluir dispositivos de entrada, tales como un teclado, ratón, bola de desplazamiento, 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 de aplicación específica (ASIC) y procesadores de señales digitales (DSP). El procesador 930 está en comunicación con los puertos 920 aguas abajo, Tx/Rx 910, los puertos 950 aguas arriba y la memoria 932. El procesador 930 comprende un módulo 914 de codificación. El módulo 914 de codificación implementa las modalidades dadas a conocer descritas en la presente memoria, tales como los métodos 100, 1000 y 1100, el mecanismo 600 y/o la aplicación 700 que pueden emplear una corriente de bits 500 y/o una imagen dividida en particiones según el esquema 800 de formación de mosaicos de vídeo flexible. El módulo 914 de codificación también puede implementar cualquier otro método/mecanismo descrito en la presente memoria. Además, el módulo 914 de codificación puede implementar un sistema 200 de códec, un codificador 300 y/o un decodificador 400. Por ejemplo, el módulo 914 de codificación puede dividir un mosaico de primer nivel de imagen y dividir mosaicos de primer nivel en mosaicos de segundo nivel. El módulo 914 de codificación también puede asignar tales mosaicos a grupos de mosaicos rectangulares para soportar la extracción de subimágenes por separado. El módulo 914 de codificación también puede señalizar datos de soporte para indicar la configuración de los mosaicos de primer nivel y los mosaicos de segundo nivel. El módulo 914 de codificación soporta además el empleo de tales mecanismos para combinar subimágenes a diferentes resoluciones en una sola imagen para varios casos de uso como se describe en la presente memoria. Como tal, el módulo 914 de codificación mejora la funcionalidad del dispositivo 900 de codificación de vídeo así como también resuelve problemas que son específicos de las técnicas de codificación de vídeo. Además, el módulo 914 de codificación efectúa una transformación del dispositivo 900 de codificación de vídeo a un estado diferente. Alternativamente, el módulo 914 de codificación puede implementarse como instrucciones almacenadas en la memoria 932 y ejecutadas por el procesador 930 (por ejemplo, como un producto de programa informático almacenado en un medio no

transitorio).

La memoria 932 comprende uno o más tipos de memoria, tales como discos, unidades de cinta, unidades de estado sólido, memoria de solo lectura (ROM), memoria de acceso aleatorio (RAM), memoria flash, memoria de contenido ternario direccionable (TCAM), memoria de acceso aleatorio estática (SRAM), etc. La memoria 932 puede utilizarse 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 1000 ejemplar de codificación de una imagen que emplea un esquema de formación de mosaicos flexible, tal como el esquema 800 de formación de mosaicos de vídeo flexible. El método 1000 puede ser empleado por un codificador, tal como un sistema 200 de códec, un codificador 300 y/o un dispositivo 900 de codificación de vídeo cuando se realiza el método 100, el mecanismo 600 y/o la aplicación 700 de soporte. Además, el método 1000 puede emplearse para generar una corriente de bits 500 para su transmisión a un decodificador, tal como el decodificador 400.

El método 1000 puede comenzar cuando un codificador recibe una secuencia de vídeo que incluye una pluralidad de imágenes y determina que se codifique esa secuencia de vídeo en una corriente de bits, por ejemplo basándose en la entrada del usuario. Como ejemplo, la secuencia de vídeo y, por tanto, las imágenes, se pueden codificar a una pluralidad de resoluciones. En la etapa 1001, una imagen se divide en una pluralidad de mosaicos de primer nivel. En la etapa 1003, un subconjunto de mosaicos de primer nivel se divide en una pluralidad de mosaicos de segundo nivel. Cada mosaico de segundo nivel puede contener un solo segmento rectangular de datos de imagen. En algunos ejemplos, los mosaicos de primer nivel fuera del subconjunto contienen datos de imagen a una primera resolución, y los mosaicos de segundo nivel contienen datos de imagen a una segunda resolución diferente de la primera resolución. En algunos ejemplos, cada mosaico de primer nivel en el subconjunto de mosaicos de primer nivel incluye dos o más mosaicos completos de segundo nivel.

En la etapa 1005, los mosaicos de primer nivel y los mosaicos de segundo nivel se asignan a uno o más grupos de mosaicos de tal manera que todos los mosaicos de un grupo de mosaicos asignado que contiene los mosaicos de segundo nivel estén limitados a cubrir una parte rectangular de la imagen. Como ejemplo específico, cada uno de los uno o más grupos de mosaicos se puede limitar a cubrir una parte rectangular de la imagen cuando cualquier mosaico de primer nivel se divide en una pluralidad de mosaicos de segundo nivel (por ejemplo, en cualquier momento en que se emplee formación de mosaicos flexible). Por consiguiente, los mosaicos de primer nivel y los mosaicos de segundo nivel no se asignan a uno o más grupos de mosaicos mediante un orden de exploración de trama que atraviesa horizontalmente la imagen desde un límite izquierdo a un límite derecho. Además, cubrir la parte rectangular de la imagen puede incluir cubrir menos de una parte horizontal completa de la imagen.

En la etapa 1007, los mosaicos de primer nivel y los mosaicos de segundo nivel se codifican en una corriente de bits. En algunos ejemplos, los datos que indican una configuración de mosaicos de segundo nivel pueden codificarse en la corriente de bits en un conjunto de parámetros de imagen asociados con la imagen. La configuración de los mosaicos de segundo nivel puede señalizarse como un número de columnas de mosaicos de segundo nivel y un número de filas de mosaicos de segundo nivel para mosaicos de primer nivel divididos en particiones (por ejemplo, en el PPS). En algunos ejemplos, los datos que indican explícitamente un número de columnas de mosaicos de segundo nivel y un número de filas de mosaicos de segundo nivel se omiten de la corriente de bits cuando el ancho del mosaico de primer nivel es menor que el doble del umbral de ancho mínimo y la altura del mosaico de primer nivel es menor que el doble de un umbral de altura mínima. En algunos ejemplos, las indicaciones de división que indican los mosaicos de primer nivel que están divididos en mosaicos de segundo nivel pueden codificarse en la corriente de bits (por ejemplo, en el PPS). En algunos ejemplos, las indicaciones de división se pueden omitir de la corriente de bits para los mosaicos de primer nivel correspondientes cuando el ancho de un mosaico de primer nivel es menor que un umbral de ancho mínimo y la altura de un mosaico de primer nivel es menor que un umbral de altura mínima.

En la etapa 1009, la corriente de bits puede almacenarse en la memoria para la comunicación hacia un decodificador. La corriente de bits se puede transmitir al decodificador bajo solicitud.

La fig. 11 es un diagrama de flujo de un método 1100 ejemplar de decodificación de una imagen que emplea un esquema de formación de mosaicos flexible, tal como el esquema 800 de formación de mosaicos de vídeo flexible. El método 1100 puede ser empleado por un decodificador, tal como un sistema 200 de códec, un decodificador 400 y/o un dispositivo 900 de codificación de vídeo cuando se realiza el método 100, el mecanismo 600 y/o la aplicación 700 de soporte. Además, el método 1100 puede emplearse al recibir una corriente de bits 500 de un codificador, tal como el codificador 300.

El método 1100 puede comenzar cuando un decodificador comienza a recibir una corriente de bits de datos codificados que representan una secuencia de vídeo, por ejemplo, como resultado del método 1000. La

corriente de bits puede contener datos de vídeo de secuencias de vídeo codificadas a una pluralidad de resoluciones. En la etapa 1101, se recibe una corriente de bits. La corriente de bits incluye una imagen dividida en particiones de una pluralidad de mosaicos de primer nivel. Un subconjunto de los mosaicos de primer nivel se divide además en una pluralidad de mosaicos de segundo nivel. Cada mosaico de segundo nivel puede contener un solo segmento rectangular de datos de imagen. En algunos ejemplos, los mosaicos de primer nivel fuera del subconjunto contienen datos de imagen a una primera resolución y los mosaicos de segundo nivel contienen datos de imagen a una segunda resolución diferente de la primera resolución. En algunos ejemplos, cada mosaico de primer nivel en el subconjunto de mosaicos de primer nivel incluye dos o más mosaicos completos de segundo nivel. Los mosaicos de primer nivel y los mosaicos de segundo nivel pueden asignarse a uno o más grupos de mosaicos de manera que todos los mosaicos de un grupo de mosaicos asignado que contenga los mosaicos de segundo nivel estén limitados a cubrir una parte rectangular de la imagen. Por ejemplo, cada uno de los uno o más grupos de mosaicos puede estar limitado a cubrir una parte rectangular de la imagen cuando cualquier mosaico de primer nivel se divide en una pluralidad de mosaicos de segundo nivel (por ejemplo, se emplea formación de mosaicos flexible). Además, los mosaicos de primer nivel y los mosaicos de segundo nivel no pueden asignarse a uno o más grupos de mosaicos mediante un orden de exploración de trama que atraviesa horizontalmente la imagen desde un límite izquierdo a un límite derecho. En algunos casos, cubrir la parte rectangular de la imagen incluye cubrir menos de una parte horizontal completa de la imagen.

En la etapa 1103 se determina una configuración de los mosaicos de primer nivel y una configuración de los mosaicos de segundo nivel basándose en los uno o más grupos de mosaicos. Por ejemplo, los datos que indican la configuración de los mosaicos de segundo nivel pueden obtenerse de un conjunto de parámetros de imagen asociados con la imagen. En algunos ejemplos, la configuración de mosaicos de segundo nivel se obtiene a partir de datos que indican un número de columnas de mosaicos de segundo nivel y un número de filas de mosaicos de segundo nivel para mosaicos de primer nivel divididos en particiones. En algunos ejemplos, los datos que indican explícitamente el número de columnas de mosaicos de segundo nivel y el número de filas de mosaicos de segundo nivel se omiten de la corriente de bits para un mosaico correspondiente cuando el ancho de un mosaico de primer nivel es menor que el doble del umbral de ancho mínimo y un la altura de un mosaico de primer nivel es menor que el doble del umbral de altura mínima. En algunos ejemplos, las indicaciones de división se pueden obtener de la corriente de bits como parte de la determinación de las configuraciones de los mosaicos de primer nivel y los mosaicos de segundo nivel. Las indicaciones de división pueden indicar los mosaicos de primer nivel que están divididos en mosaicos de segundo nivel. En algunos ejemplos, los datos de indicación de división que indican explícitamente si un mosaico de primer nivel está dividido en mosaicos de segundo nivel se omiten de la corriente de bits cuando el ancho de un mosaico de primer nivel es menor que un umbral de ancho mínimo y la altura de un mosaico de primer nivel es menor que un umbral de altura mínima.

En la etapa 1105, los mosaicos de primer nivel y los mosaicos de segundo nivel se decodifican basándose en la configuración de los mosaicos de primer nivel y en la configuración de los mosaicos de segundo nivel. En la etapa 1107, se genera una secuencia de vídeo reconstruida para su visualización basada en los mosaicos de primer nivel y los mosaicos de segundo nivel decodificados.

La fig. 12 es un diagrama esquemático de un sistema 1200 ejemplar para codificar una secuencia de vídeo empleando un esquema de formación de mosaicos flexible, tal como el esquema 800 de formación de mosaicos de vídeo flexible. El sistema 1200 puede implementarse mediante un codificador y un decodificador, tal como un sistema 200 de códec, un codificador 300, un decodificador 400 y/o un dispositivo 900 de codificación de vídeo. Además, el sistema 1200 puede emplearse al implementar el método 100, 1000, 1100, el mecanismo 600 y/o la aplicación 700. El sistema 1200 también puede codificar datos en una corriente de bits, tal como la corriente de bits 500, y decodificar dicha corriente de bits para mostrarlo a un usuario.

El sistema 1200 incluye un codificador 1202 de vídeo. El codificador 1202 de vídeo comprende un módulo 1201 de división para dividir una imagen en una pluralidad de mosaicos de primer nivel y dividir un subconjunto de mosaicos de primer nivel en una pluralidad de mosaicos de segundo nivel. El codificador 1202 de vídeo comprende además un módulo 1203 de asignación para asignar los mosaicos de primer nivel y los mosaicos de segundo nivel a uno o más grupos de mosaicos de tal manera que todos los mosaicos en un grupo de mosaicos asignado que contiene los mosaicos de segundo nivel están limitados a cubrir una parte rectangular de la imagen. El codificador 1202 de vídeo comprende además un módulo 1205 de codificación para codificar los mosaicos de primer nivel y los mosaicos de segundo nivel en una corriente de bits. El codificador 1202 de vídeo comprende además un módulo 1207 de almacenamiento para almacenar la corriente de bits para la comunicación hacia un decodificador. El codificador 1202 de vídeo comprende además un módulo 1209 de transmisión para transmitir la corriente de bits hacia un decodificador. El codificador 1202 de vídeo puede configurarse además para realizar cualquiera de las etapas del método 1000.

El sistema 1200 también incluye un decodificador 1210 de vídeo. El decodificador 1210 de vídeo comprende un módulo 1211 de recepción para recibir una corriente de bits que incluye una imagen dividida en una pluralidad de mosaicos de primer nivel, en donde un subconjunto de mosaicos de primer nivel se divide

además en una pluralidad de mosaicos de segundo nivel, y en donde los mosaicos de primer nivel y los mosaicos de segundo nivel se asignan a uno o más grupos de mosaicos de tal manera que todos los mosaicos de un grupo de mosaicos asignado que contiene los mosaicos de segundo nivel están limitados a cubrir una parte rectangular de la imagen. El decodificador 1210 de vídeo comprende además un módulo 1213 de determinación para determinar una configuración de los mosaicos de primer nivel y una configuración de los mosaicos de segundo nivel basándose en los uno o más grupos de mosaicos. El decodificador 1210 de vídeo comprende además un módulo 1215 de decodificación para decodificar los mosaicos de primer nivel y los mosaicos de segundo nivel basándose en la configuración de los mosaicos de primer nivel y la configuración de los mosaicos de segundo nivel. El decodificador 1210 de vídeo comprende además un módulo 1217 de generación para generar una secuencia de vídeo reconstruida para su visualización basada en los mosaicos decodificados de primer nivel y en los mosaicos de segundo nivel. El decodificador 1210 de vídeo puede configurarse además para realizar cualquiera de las etapas 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 intervalo que incluye ± 10% del número siguiente, a menos que se indique lo contrario.

También debería comprenderse que las etapas de los métodos ejemplares expuestos en la presente memoria no se requiere necesariamente que se realicen en el orden descrito, y el orden de las etapas de dichos métodos debería comprenderse como meramente ejemplar. Asimismo, se pueden incluir etapas adicionales en tales métodos, y se pueden omitir o combinar ciertas etapas, en métodos consistentes con diversas realizaciones de la presente descripción.

Si bien se han proporcionado varias realizaciones en la presente descripción, se puede comprender que los sistemas y métodos descritos se pueden realizar en muchas otras formas específicas sin desviarse del espíritu o alcance de la presente descripción. Los presentes ejemplos han de considerarse ilustrativos y no restrictivos, y la intención no se limita a los detalles que se dan en la presente memoria. 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 realizaciones como discretos o separados pueden combinarse o integrarse con otros sistemas, componentes, técnicas o métodos sin desviarse del alcance de la presente descripción. Un experto en la técnica puede comprobar otros ejemplos de cambios, sustituciones y alteraciones y pueden realizarse.

REIVINDICACIONES

1. Un método implementado en un codificador para codificar un vídeo que incluye una pluralidad de imágenes, comprendiendo el método:

la división en particiones (1001), mediante un procesador del codificador, de una imagen de la pluralidad de imágenes a una pluralidad de mosaicos de primer nivel, en donde la pluralidad de mosaicos de primer nivel se definen por un conjunto de columnas de mosaicos y un conjunto de filas de mosaicos;

la división en particiones (1003), mediante el procesador, de un subconjunto de los mosaicos de primer nivel en una o más filas de mosaicos y una o más columnas de mosaicos para obtener una pluralidad de mosaicos de segundo nivel;

la asignación (1005), mediante el procesador, de los mosaicos de primer nivel y los mosaicos de segundo nivel a uno o más grupos de mosaicos, en donde todos los mosaicos de segundo nivel creados desde un único mosaico de primer nivel se asignan a un mismo grupo de mosaicos, de tal manera que todos los mosaicos en un grupo de mosaicos asignado estén limitados para cubrir una región rectangular de la imagen;

la codificación (1007), mediante el procesador, de los mosaicos de primer nivel y de los mosaicos de segundo nivel en una corriente de bits; y

la codificación, mediante el procesador, de un indicador, para cada uno de los mosaicos de primer nivel, en la corriente de bits, señalando el indicador si el mosaico de primer nivel se divide en los mosaicos de segundo nivel.

2. El método de la reivindicación 1, comprendiendo el método además:

el almacenamiento, en una memoria del codificador, de la corriente de bits para la comunicación hacia un decodificador.

3. El método de cualquiera de las reivindicaciones 1-2, caracterizado porque cada uno de los uno o más grupos de mosaicos están limitados a cubrir una región rectangular de la imagen cuando cualquier mosaico de primer nivel se divide en una pluralidad de mosaicos de segundo nivel.

4. El método de cualquiera de las reivindicaciones 1-3, en donde los mosaicos de primer nivel y los mosaicos de segundo nivel no se asignan al uno o más grupos de mosaicos mediante un orden de exploración de trama que atraviesa horizontalmente la imagen desde un límite izquierdo a un límite derecho para evitar la obtención de formas no rectangulares.

5. El método de cualquiera de las reivindicaciones 1-4, en donde cubrir la región rectangular de la imagen incluye cubrir menos de una región horizontal completa de la imagen.

6. El método de cualquiera de las reivindicaciones 1-5, en donde el indicador se omite de la corriente de bits para cada mosaico de primer nivel con un ancho que es menor que un umbral de ancho mínimo y una altura que es menor que un umbral de altura mínima, para indicar que el mosaico de primer nivel no se divide en mosaicos de segundo nivel.

7. Un método implementado en un decodificador para decodificar una corriente de bits de vídeo que se genera desde un vídeo que incluye una pluralidad de imágenes, comprendiendo el método:

la recepción (1101), mediante un procesador del decodificador a través de un receptor, incluyendo una corriente de bits una imagen de la pluralidad de imágenes dividida en particiones de una pluralidad de mosaicos de primer nivel, en donde un subconjunto de los mosaicos de primer nivel se divide además en una pluralidad de mosaicos de segundo nivel que cada uno contiene un único segmento rectangular de datos de imagen, en donde la pluralidad de mosaicos de primer nivel se definen por un conjunto de columnas de mosaicos y un conjunto de filas de mosaicos, y cada mosaico de segundo nivel se define por una o más columnas de mosaicos y una o más filas de mosaicos; y en donde los mosaicos de segundo nivel se asignan al uno o más grupos de mosaicos, y en donde todos los mosaicos de segundo nivel creados desde un único mosaico de primer nivel se asignan a un mismo grupo de mosaicos, de tal manera que todos los mosaicos en un grupo de mosaicos asignado están limitados a cubrir una región rectangular de la imagen; la corriente de bits incluye además un indicador para cada uno de los mosaicos de primer nivel, señalando el indicador si el mosaico de primer nivel se divide en mosaicos de segundo nivel;

la determinación (1103), mediante el procesador, de una configuración de los mosaicos de primer nivel y una configuración de los mosaicos de segundo nivel basándose en dichos indicadores en la corriente de bits;

la decodificación (1105), mediante el procesador, de los mosaicos de primer nivel y de los mosaicos de segundo nivel basándose en la configuración de los mosaicos de primer nivel y la configuración de los mosaicos de segundo nivel; y

la generación (1107), mediante el procesador, de una secuencia de vídeo reconstruida para su visualización basándose en los mosaicos de primer nivel y en los mosaicos de segundo nivel decodificados.

8. El método de cualquiera de las reivindicación 7, en donde cada uno de los uno o más grupos de mosaicos están limitados a cubrir una región rectangular de la imagen cuando cualquier mosaico de primer nivel se divide en una pluralidad de mosaicos de segundo nivel.

9. El método de cualquiera de las reivindicaciones 7-8, en donde los mosaicos de primer nivel y los mosaicos de segundo nivel no se asignan al uno o más grupos de mosaicos mediante un orden de exploración de trama que atraviesa horizontalmente la imagen desde un límite izquierdo a un límite derecho para evitar la obtención de formas no rectangulares.

10. El método de cualquiera de las reivindicaciones 7-9, en donde cubrir la región rectangular de la imagen incluye cubrir menos de una región horizontal completa de la imagen.

11. El método de cualquiera de las reivindicaciones 7-10, en donde el indicador se omite de la corriente de bits para cada mosaico de primer nivel con un ancho que es menor que un umbral de ancho mínimo y una altura que es menor que un umbral de altura mínimo, para indicar que el mosaico de primer nivel no se divide en mosaicos de segundo nivel.

12. Un dispositivo (900) de codificación de vídeo, que comprende:

un procesador (930) configurado para realizar el método de cualquiera de las reivindicaciones 1-6;

un transmisor acoplado al procesador, configurado para transmitir una corriente de bits generador por el procesador.

13. Un dispositivo (900) de decodificación de vídeo, que comprende:

un procesador (930) y un receptor acoplado al procesador, el procesador y el receptor configurados para realizar el método de cualquiera de las reivindicaciones 7-11.

14. Siendo generada una corriente de bits codificada desde un vídeo que incluye una pluralidad de imágenes, la corriente de bits comprende una pluralidad de mosaicos de primer nivel, una pluralidad de mosaicos de segundo nivel y un indicador para cada uno de los mosaicos de primer nivel, señalando el indicador si el mosaico de primer nivel se divide en mosaicos de segundo nivel;

en donde los mosaicos de primer nivel se obtienen dividiendo una imagen y los mosaicos de segundo nivel se obtienen dividiendo un subconjunto de mosaicos de primer nivel, y cada uno de los mosaicos de segundo nivel contiene un único segmento rectangular, en donde la pluralidad de mosaicos de primer nivel definidos por un conjunto de columnas de mosaicos y un conjunto de filas de mosaicos, y cada mosaico de segundo nivel se define por una o más columnas de mosaicos y una o más filas de mosaicos; y en donde los mosaicos de primer nivel y los mosaicos de segundo nivel se asignan al uno o más grupos de mosaicos, y en donde todos los mosaicos de segundo nivel creados desde un único mosaico de primer nivel se asignan a un mismo grupo de mosaicos, de tal manera que todos los mosaicos en un grupo de mosaicos correspondiente están limitados a cubrir una parte regular de la imagen.

15. Un codificador, que comprende un circuito de procesamiento para llevar a cabo el método según las reivindicaciones 1-6.

16. Un decodificador, que comprende un circuito de procesamiento para llevar a cabo el método según cualquiera de las reivindicaciones 7-11.

17. Un producto de programa informático que comprende un código de programa, que cuando el programa se ejecuta mediante un ordenador o un procesador, hace que el ordenador o el procesador lleven a cabo el método según cualquiera de las reivindicaciones 1-11.

18. Un medio legible por ordenador no transitorio que lleva un código de programa que, cuando se ejecuta mediante un dispositivo informático, hace que el dispositivo informático realice el método de cualquiera de las reivindicaciones 1-11.

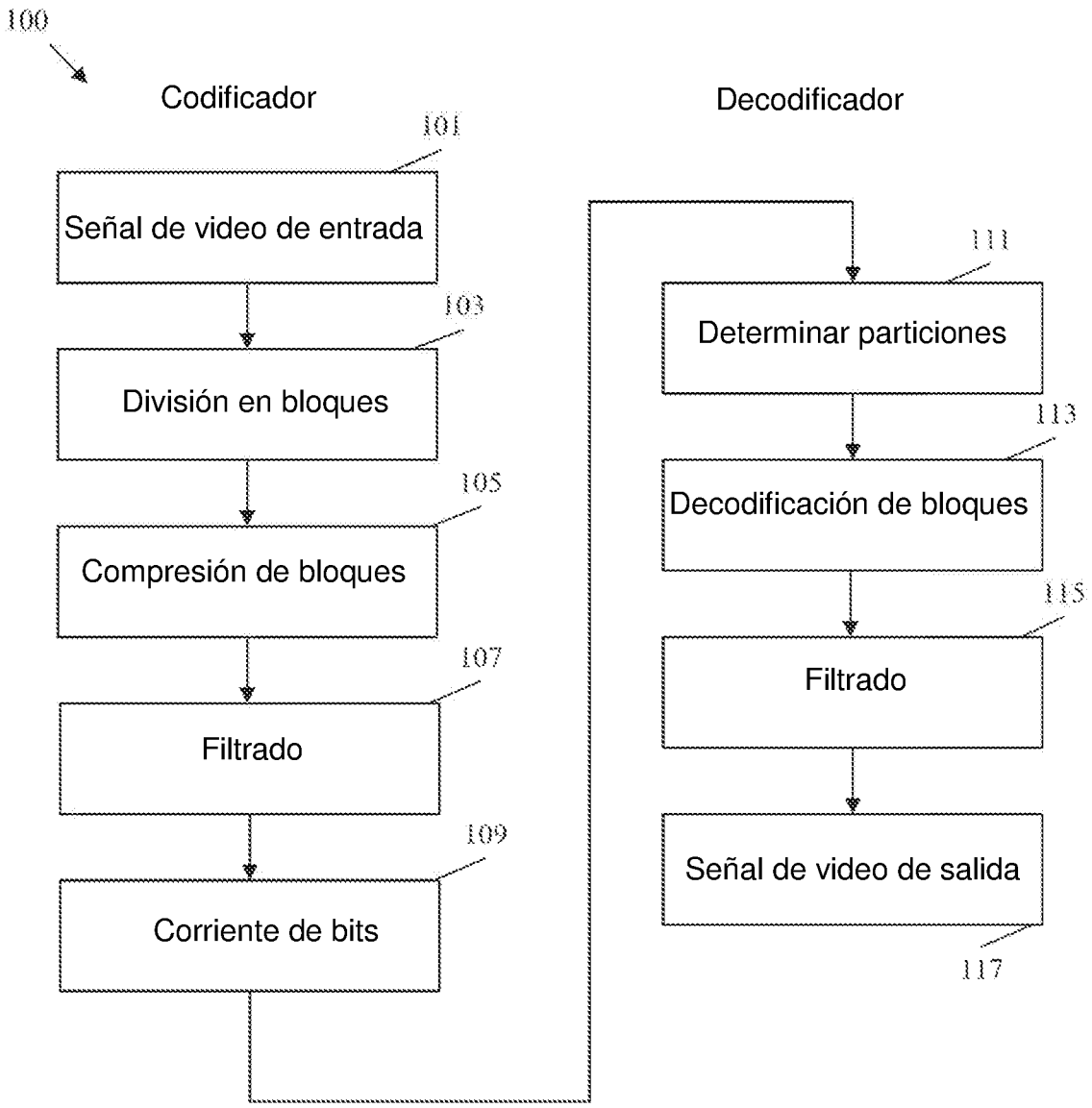
100
Codificador
Decodificador
101
Señal de video de entrada
103
División en bloques
105
Compresión de bloques
107
Filtrado
109
Corriente de bits
111
Determinar particiones
113
Decodificación de bloques
115
Filtrado
Señal de video de salida
117

FIG. 1

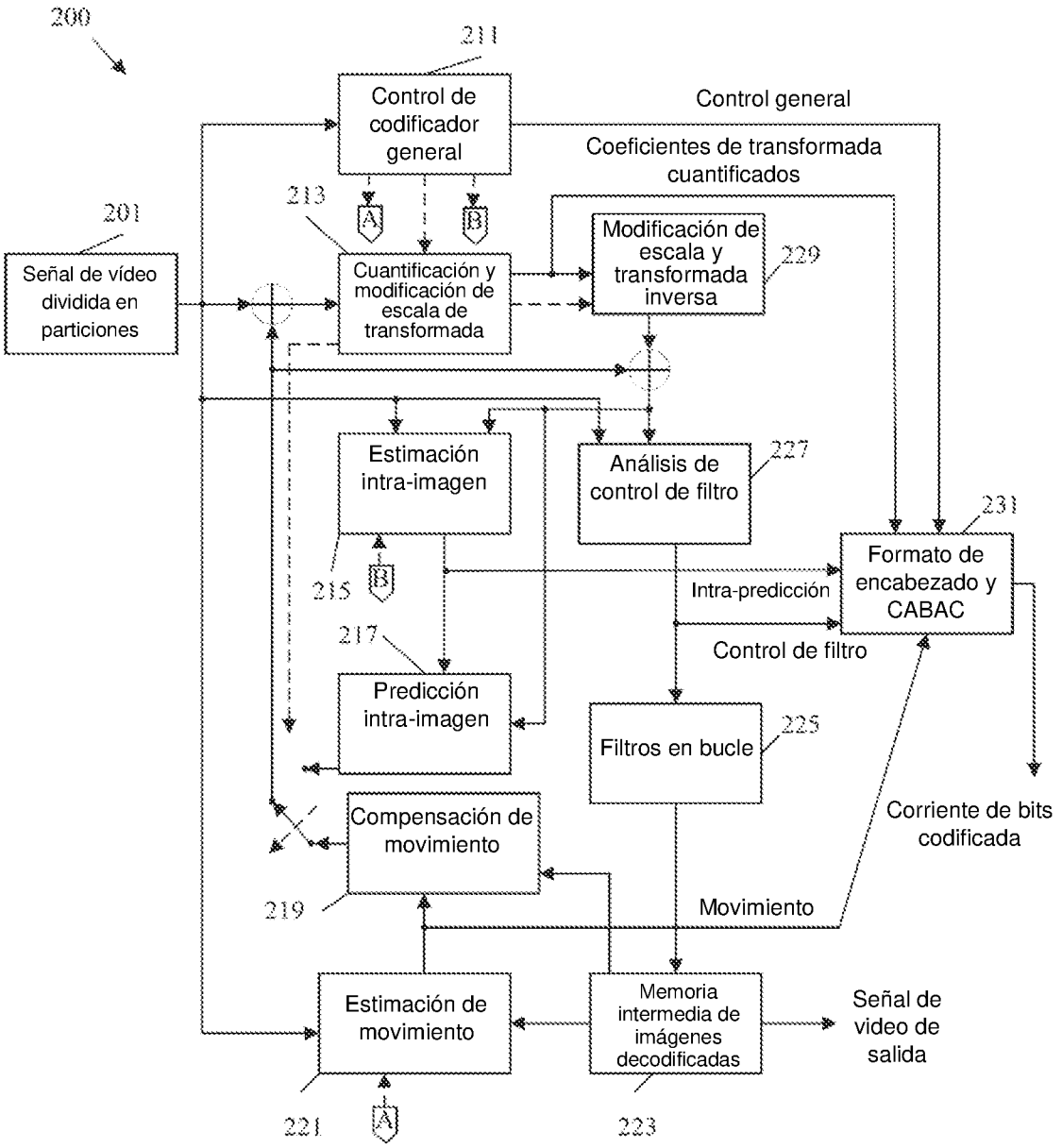
200
211
Control de codificador general
Control general
Coeficientes de transformada cuantificados
213
201
Señal de vídeo dividida en particiones
Cuantificación y modificación de escala de transformada
Modificación de escala y transformada inversa
229
Estimación intra-imagen
Análisis de control de filtro
227
231
Formato de encabezado y CABAC
215
Intra-predicción
217
Control de filtro
Predicción intra-imagen
Filtros en bucle
225
Compensación de movimiento
Corriente de bits codificada
219
Movimiento
Estimación de movimiento
Memoria intermedia de imágenes decodificadas
Señal de video de salida
221
223

FIG. 2

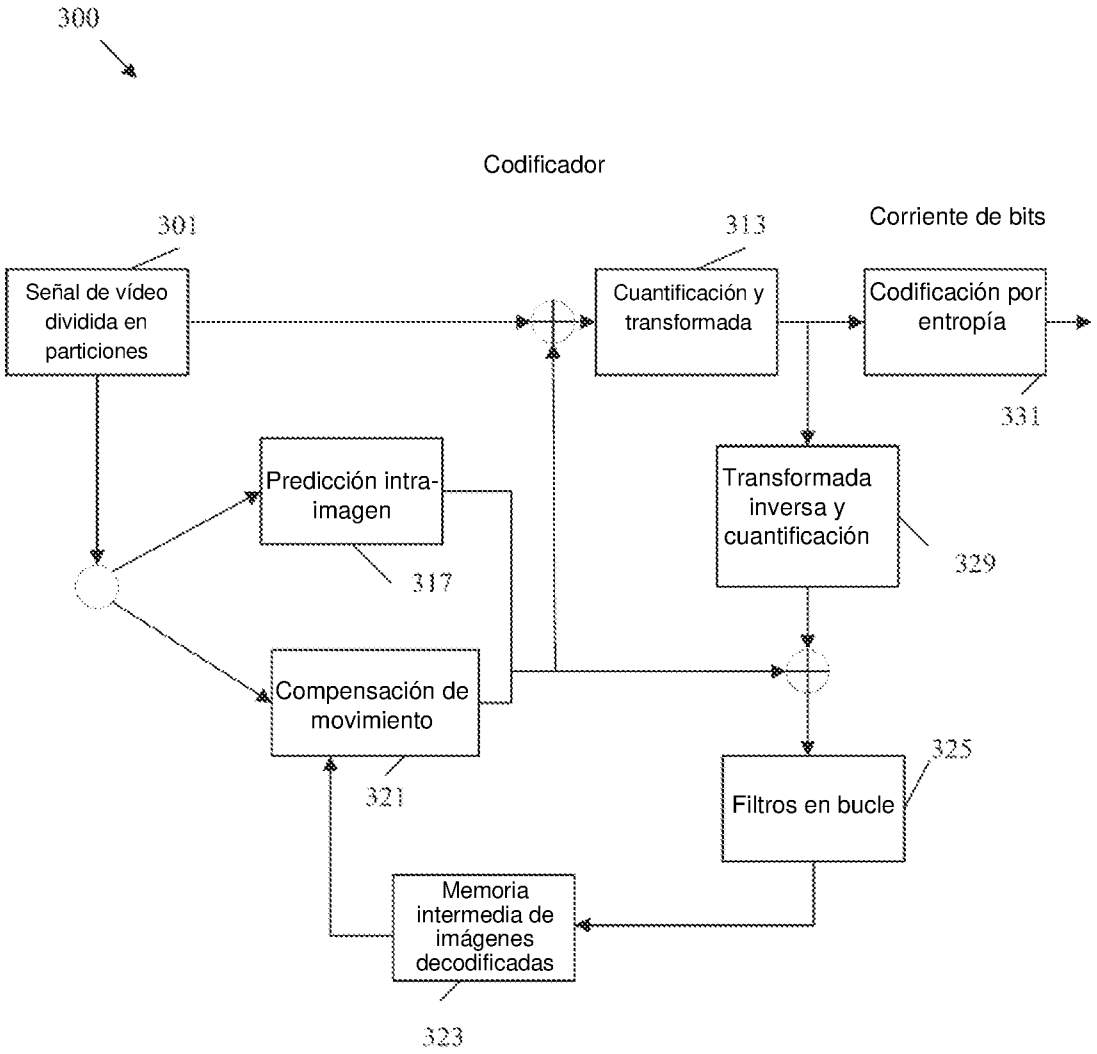
300
Codificador
301
Señal de vídeo dividida en particiones
313
Cuantificación y transformada
Corriente de bits
Codificación por entropía
331
Predicción intra-imagen
317
Transformada inversa y cuantificación
329
Compensación de movimiento
321
325
Filtros en bucle
Memoria intermedia de imágenes decodificadas
323

FIG. 3

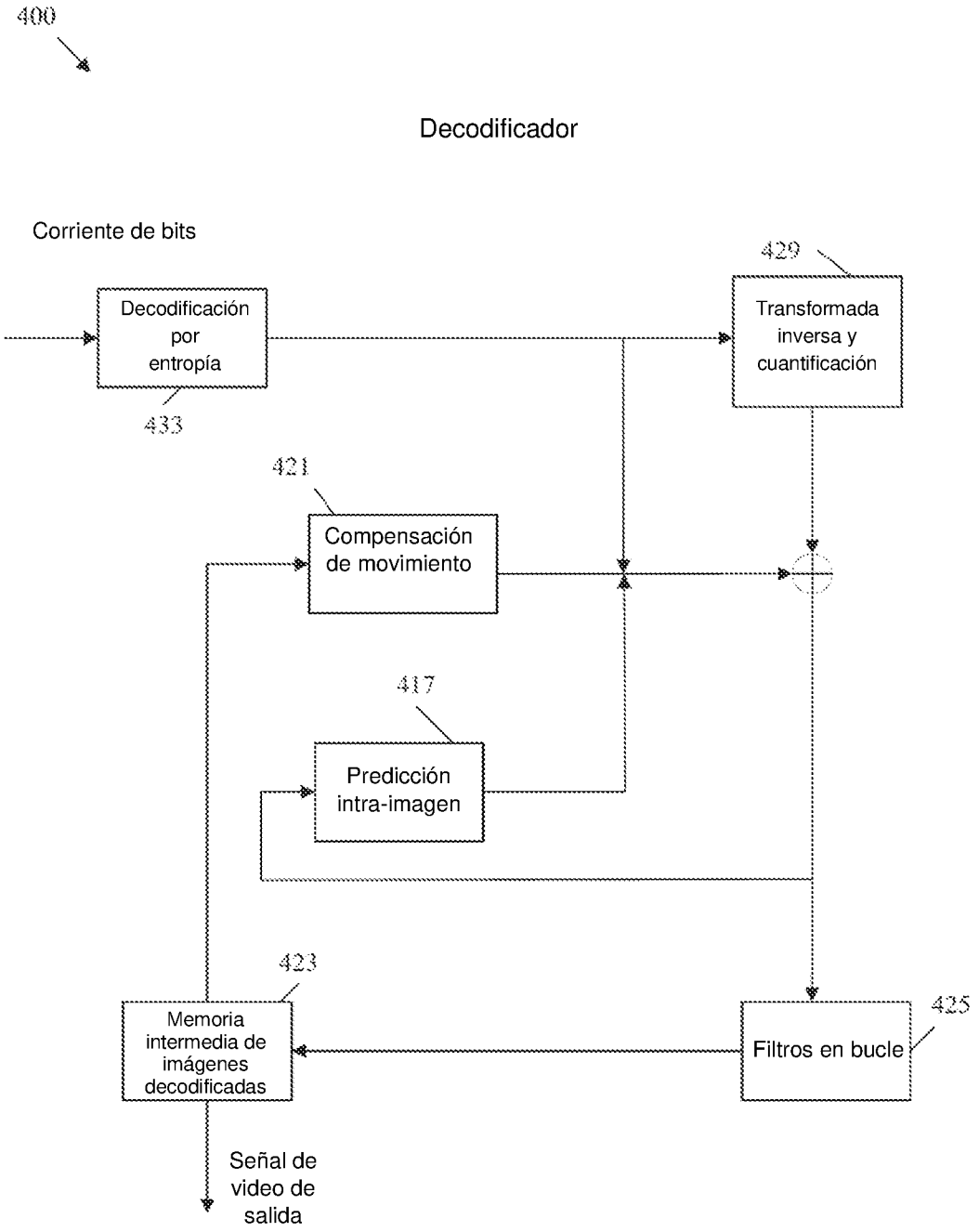
400
Decodificador
Corriente de bits
Decodificación por entropía
433
429
Transformada inversa y cuantificación
421
Compensación de movimiento
417
Predicción intra-imagen
423
Memoria intermedia de imágenes decodificadas
425
Filtros en bucle
Señal de video de salida

FIG. 4

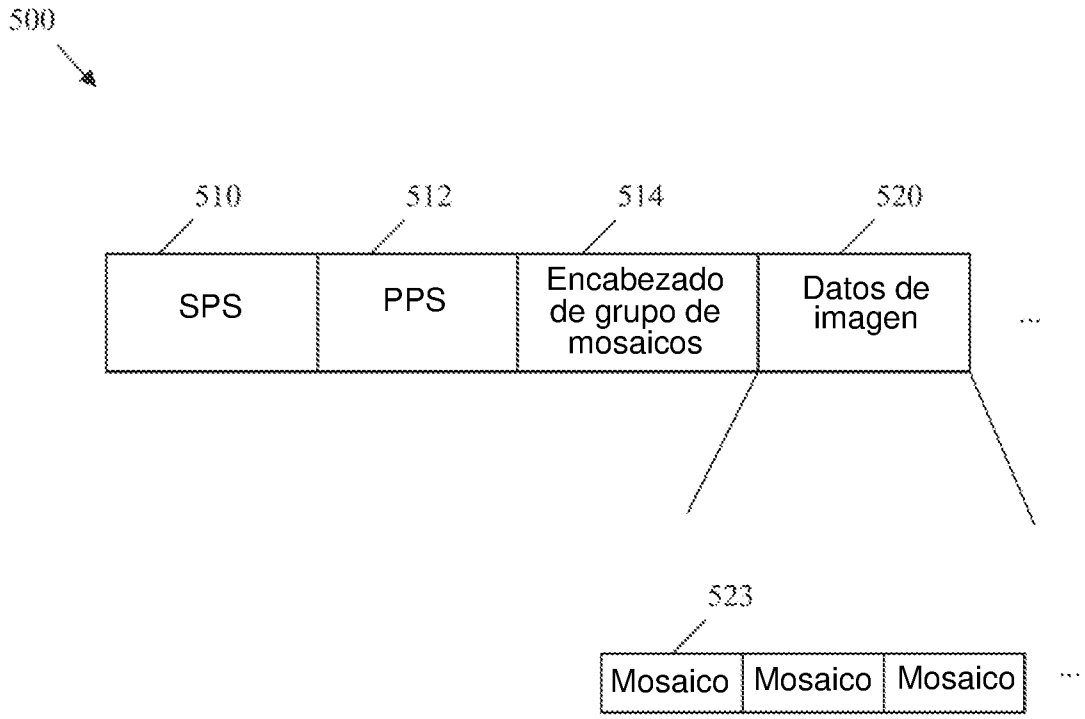
500
510
512
514
520
SPS
PPS
Encabezado de grupo de mosaicos
Datos de imagen
...
523
Mosaico
Mosaico
Mosaico
...

FIG. 5

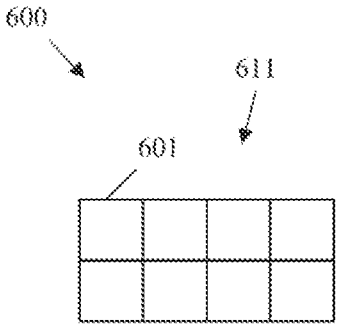

FIG. 6A

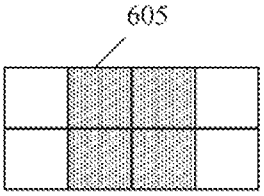

FIG. 6C

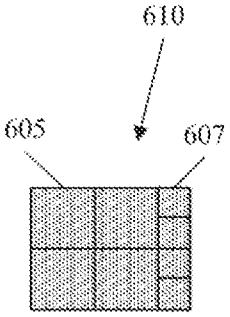

FIG. 6E

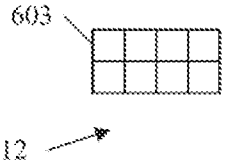

FIG. 6B

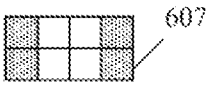

FIG. 6D

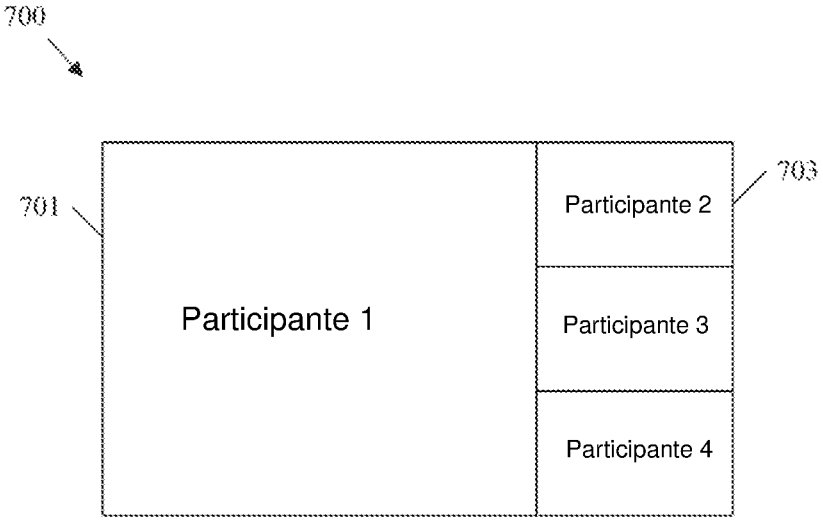

FIG. 7

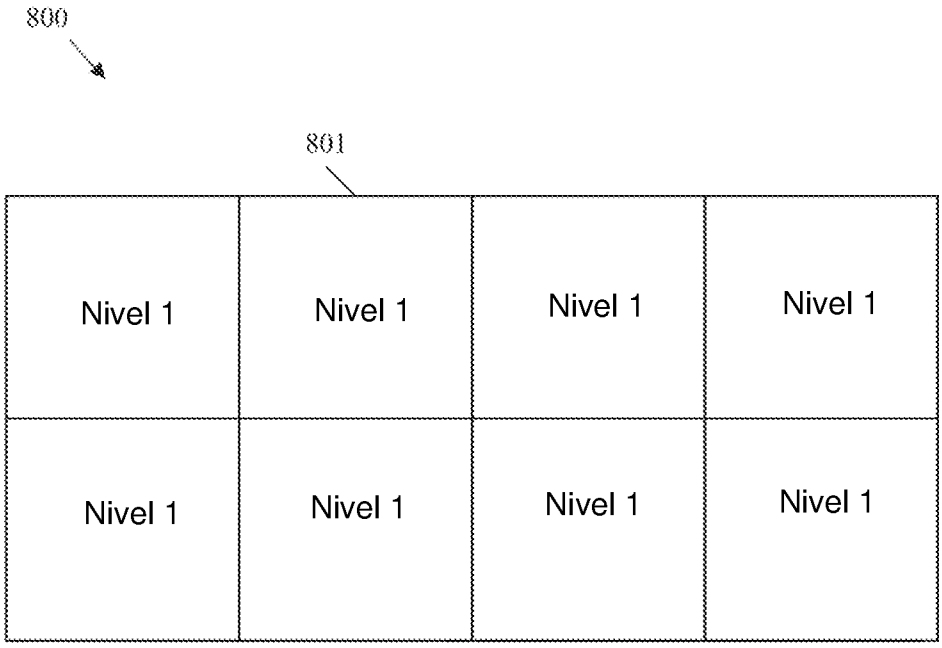
800
801
Nivel 1
Nivel 1
Nivel 1
Nivel 1
Nivel 1
Nivel 1
Nivel 1
Nivel 1

FIG. 8A

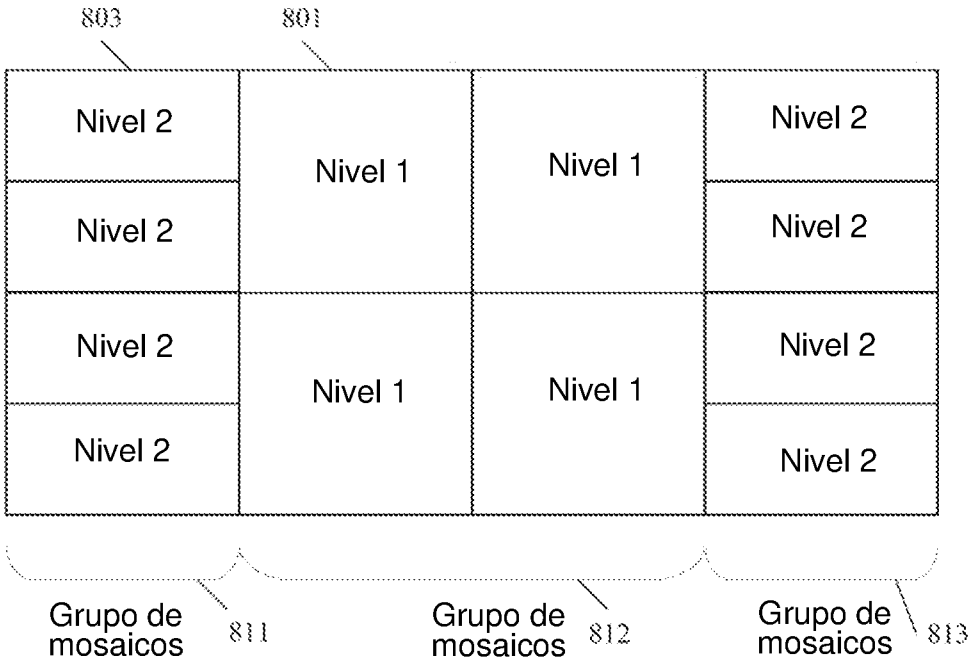
803
801
Nivel 2
Nivel 2
Nivel 2
Nivel 2
Nivel 1
Nivel 1
Nivel 1
Nivel 1
Nivel 2
Nivel 2
Nivel 2
Nivel 2
Grupo de mosaicos
811
Grupo de mosaicos
812
Grupo de mosaicos
813

FIG. 8B

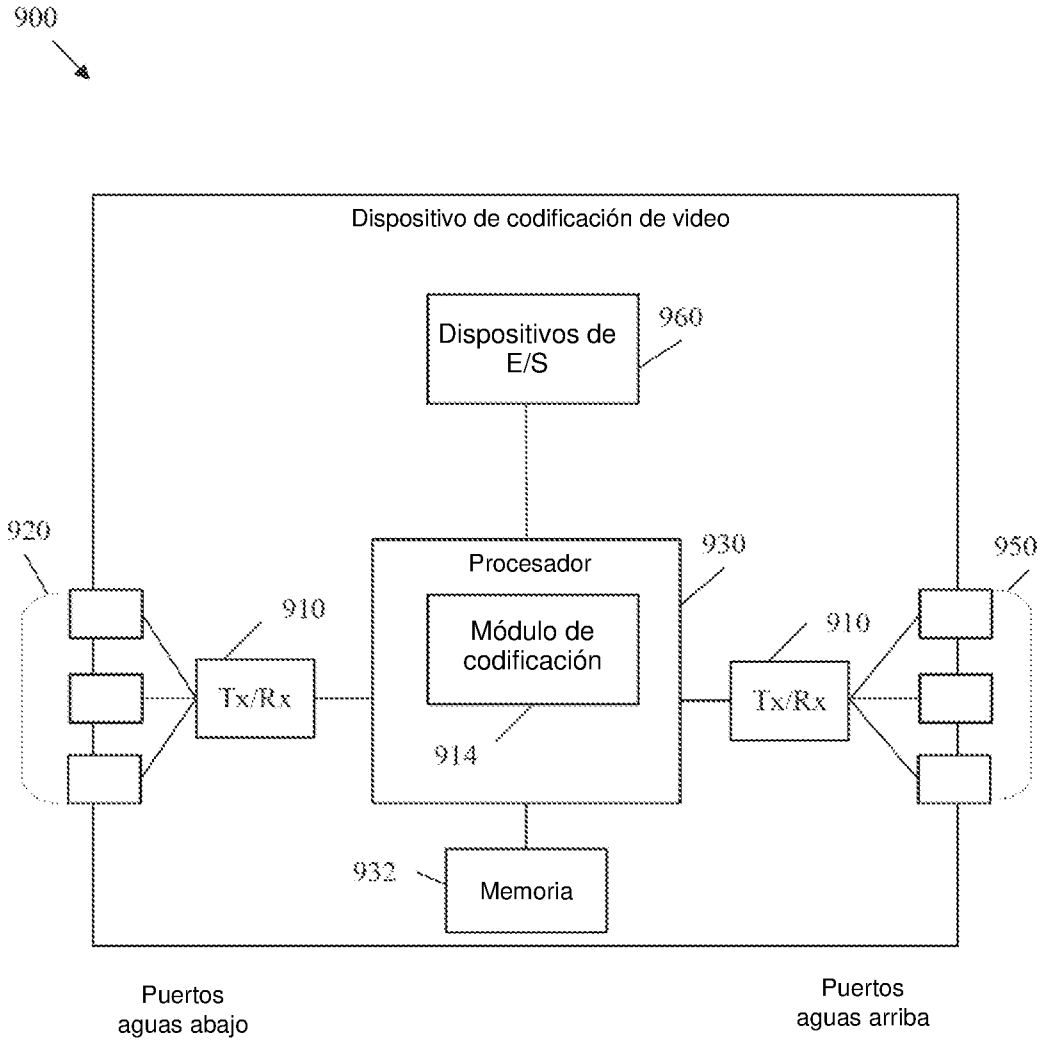
900
Dispositivo de codificación de video
Dispositivos de E/S
960
920
Procesador
930
950
910
Módulo de codificación
910
Tx/Rx
Tx/Rx
914
932
Memoria
Puertos aguas abajo
Puertos aguas arriba

FIG. 9

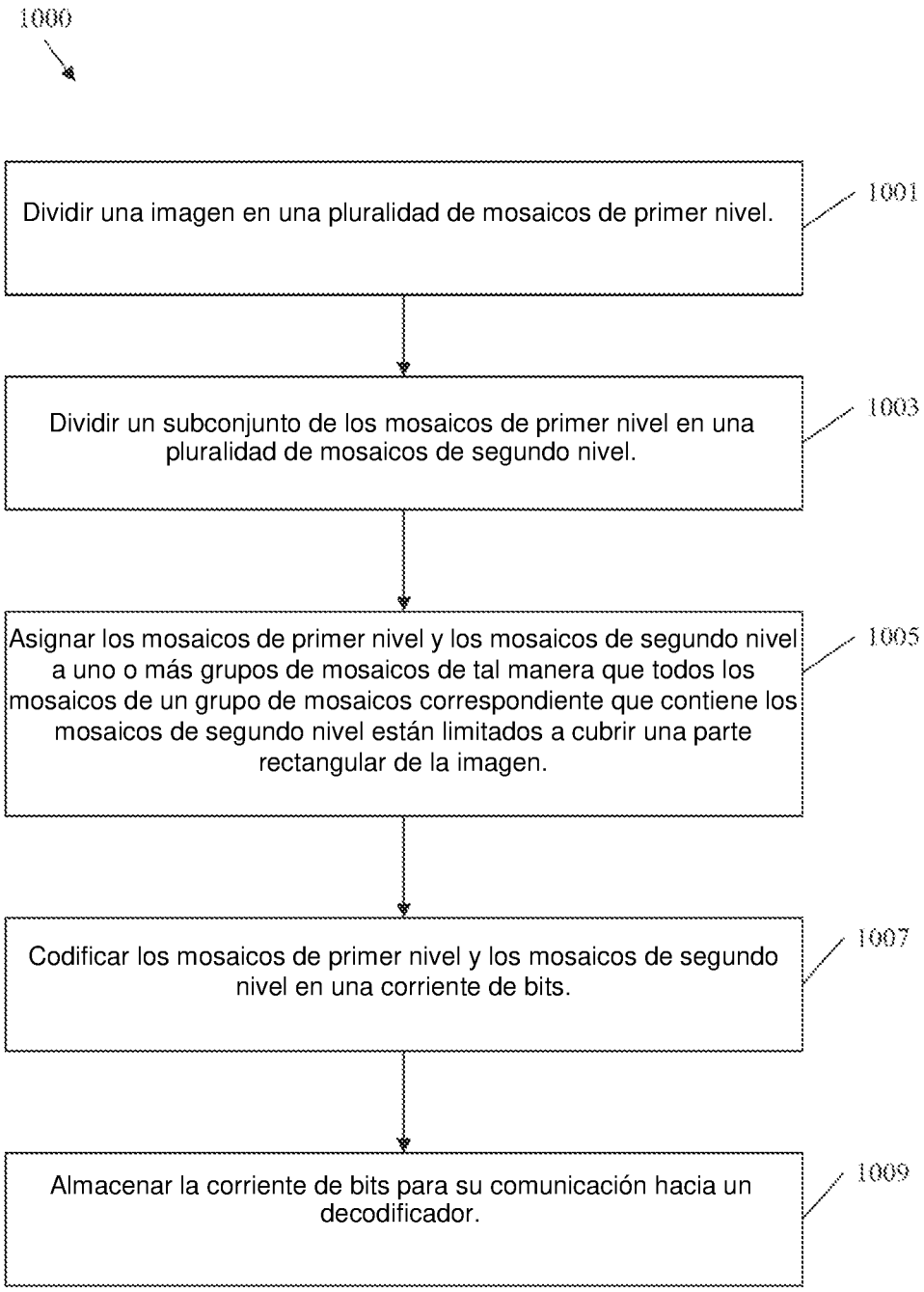
1000
Dividir una imagen en una pluralidad de mosaicos de primer nivel.
1001
Dividir un subconjunto de los mosaicos de primer nivel en una pluralidad de mosaicos de segundo nivel.
1003
Asignar los mosaicos de primer nivel y los mosaicos de segundo nivel a uno o más grupos de mosaicos de tal manera que todos los mosaicos de un grupo de mosaicos correspondiente que contiene los mosaicos de segundo nivel están limitados a cubrir una parte rectangular de la imagen.
1005
Codificar los mosaicos de primer nivel y los mosaicos de segundo nivel en una corriente de bits.
1007
Almacenar la corriente de bits para su comunicación hacia un decodificador.
1009

FIG. 10

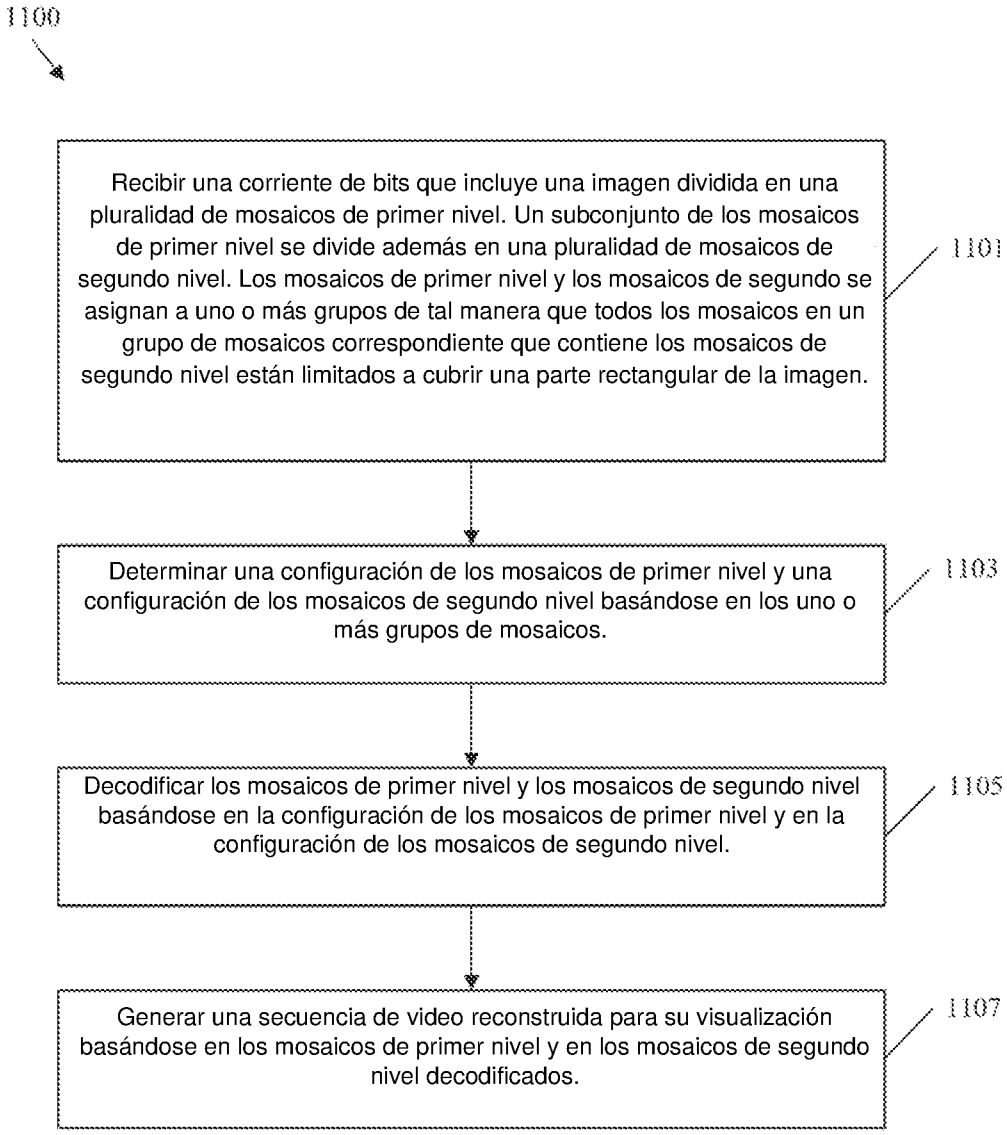
1100
Recibir una corriente de bits que incluye una imagen dividida en una pluralidad de mosaicos de primer nivel. Un subconjunto de los mosaicos de primer nivel se divide además en una pluralidad de mosaicos de segundo nivel. Los mosaicos de primer nivel y los mosaicos de segundo se asignan a uno o más grupos de tal manera que todos los mosaicos en un grupo de mosaicos correspondiente que contiene los mosaicos de segundo nivel están limitados a cubrir una parte rectangular de la imagen.
1101
Determinar una configuración de los mosaicos de primer nivel y una configuración de los mosaicos de segundo nivel basándose en los uno o más grupos de mosaicos.
1103
Decodificar los mosaicos de primer nivel y los mosaicos de segundo nivel basándose en la configuración de los mosaicos de primer nivel y en la configuración de los mosaicos de segundo nivel.
1105
Generar una secuencia de video reconstruida para su visualización basándose en los mosaicos de primer nivel y en los mosaicos de segundo nivel decodificados.
1107

FIG. 11

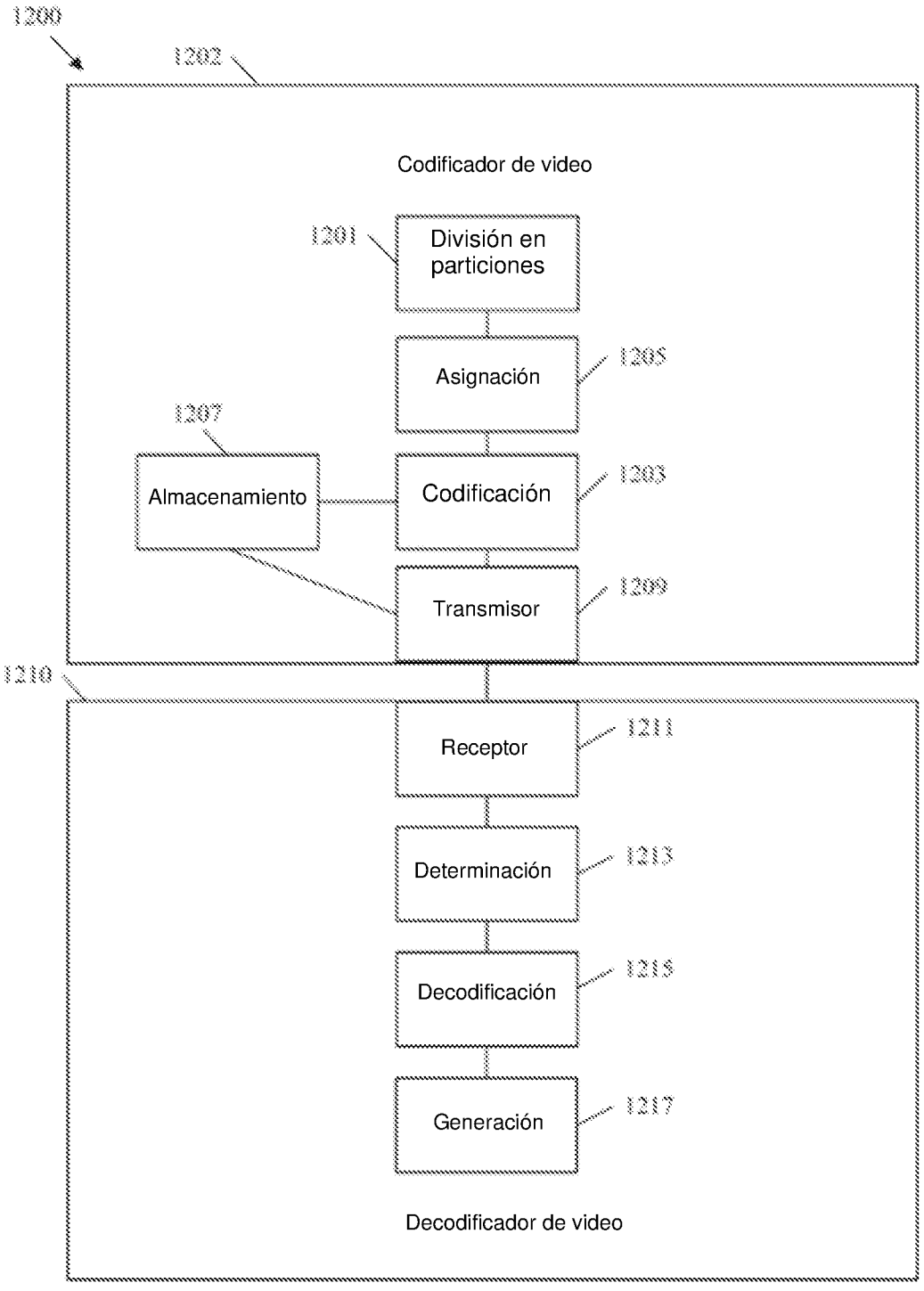
1200
1202
Codificador de video
1201
División en particiones
Asignación
1205
1207
Almacenamiento
Codificación
1203
Transmisor
1209
1210
Receptor
1211
Determinación
1213
Decodificación
1215
Generación
1217
Decodificador de video

FIG. 12