(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 981 568**

(51) Int. Cl.:

H04N 19/46 (2014.01)
H04N 19/176 (2014.01)
H04N 19/17 (2014.01)
H04N 19/174 (2014.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **09.01.2020 PCT/US2020/012974**

(87) Fecha y número de publicación internacional: **16.07.2020 WO20146665**

(96) Fecha de presentación y número de la solicitud europea: **09.01.2020 E 20739138 (4)**

(97) Fecha y número de publicación de la concesión europea: **24.04.2024 EP 3906687**

(54) Título: **Señalización indicadora de nivel de subimagen en codificación de vídeo**

(30) Prioridad:

09.01.2019 US 201962790207 P

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
09.10.2024

(73) Titular/es:

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

(72) Inventor/es:

WANG, YE-KUI y
HENDRY, FNU

(74) Agente/Representante:

SÁNCHEZ SILVA, Jesús Eladio

DESCRIPCIÓN

Señalización indicadora de nivel de subimagen en codificación de vídeo

Campo técnico

La presente memoria se refiere generalmente a la codificación de vídeo, y se refiere específicamente a la gestión de subimágenes en la codificación de vídeo.

Antecedentes

La cantidad de datos de vídeo necesarios para representar incluso un vídeo relativamente corto puede ser sustancial, lo que puede dar como resultado dificultades cuando los datos se van a enviar por flujo continuo o comunicar de cualquier otra manera a través de una red de comunicaciones con capacidad de ancho de banda limitada. Por lo tanto, los datos de vídeo por lo general se comprimen antes de comunicarse a través de redes de telecomunicaciones modernas. 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 usan software y/o hardware en la fuente para codificar los datos de vídeo antes de su transmisión o almacenamiento, disminuyendo de esta manera la cantidad de datos necesarios para representar imágenes de vídeo digitales. A continuación, los datos comprimidos se reciben en el destino mediante 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 US 2014/0.362.919 A1 describe unidades de codificación que comparten una característica o atributo de codificación que pueden agruparse como un grupo de coherencia. Se explican un método de empaquetado y una sintaxis que permiten el transporte de este novedoso formato de empaquetado de vídeo. Además, el método propuesto puede utilizar la redundancia de datos reducida reutilizando la característica compartida almacenándola en caché para un acceso rápido.

ZHOU (TI) M, "AHG4: Enable parallel decoding with tiles", 9. JCT-VC MEETING; 100. REUNIÓN MPEG; 27-4-2012 - 7-5-2012; GENEVA; (JOINT COLLABORATIVE TEAM ON VIDEO CODING OF ISO/IEC JTC1/SC29/WG11 AND ITU-T SG.16); URL: HTTP://WFTP3.ITU.INT/AV-ARCH/JCTVC-SITE/, (20120416), no. JCTVC-I0118, se describe que una imagen se divide en varias subimágenes de igual tamaño; se puede construir un decodificador 4Kx2K @60 simplemente replicando cuatro veces el decodificador de núcleo único 1080p @60, cuatro núcleos de vídeo codificadores pueden codificar 4 subimágenes de igual tamaño en cuatro núcleos de vídeo codificadores en paralelo y pueden decodificarse en paralelo mediante cuatro núcleos de decodificador de vídeo.

El documento US 2014/0.003.504 A1 presenta que una unidad SEI NAL puede contener uno o más mensajes SEI, que no son necesarios para la decodificación de las imágenes de salida, pero que pueden ayudar en los procesos relacionados, como la temporización de salida de imágenes, la representación, la detección de errores, la ocultación de errores y la reserva de recursos.

Breve descripción de la invención

La invención se define en las reivindicaciones independientes. Las características adicionales de la invención se proporcionan en las reivindicaciones dependientes. A continuación, las partes de la descripción y los dibujos que se refieren a las realizaciones que no cubren las reivindicaciones no se presentan como realizaciones de la invención, sino como ejemplos útiles para comprender la invención.

En un ejemplo, la descripción incluye un método implementado en un decodificador, cuyo método comprende: recibir, por un receptor del decodificador, un flujo de bits que comprende una o más subimágenes divididas a partir de una imagen y un indicador del nivel de subimagen que indica los requisitos de recursos para decodificar una subimagen actual; analizar, mediante un procesador del decodificador, el flujo de bits para obtener el indicador de nivel de subimagen y la subimagen actual; asignar, por parte del procesador, recursos para decodificar la subimagen actual basándose en el indicador de nivel de subimagen; decodificar, mediante el procesador, la subimagen actual para crear una secuencia de vídeo empleando los recursos asignados; y reenviar, mediante el procesador, la secuencia de vídeo para su visualización. En algunos sistemas de codificación de vídeo, se señaliza un nivel para una imagen. Un nivel indica los recursos de hardware necesarios para decodificar la imagen. En algunos casos, diferentes subimágenes pueden tener una funcionalidad diferente en algunos casos y, por lo tanto, pueden tratarse de manera diferente durante el proceso de codificación. Como tal, un nivel basado en imágenes puede no ser útil para decodificar algunas subimágenes. Los presentes ejemplos son niveles de señal para cada subimagen. De esta manera, cada subimagen puede codificarse independientemente de otras subimágenes sin sobrecargar innecesariamente el decodificador al establecer requisitos de decodificación demasiado altos para las subimágenes codificadas de acuerdo con mecanismos menos complejos. La información a nivel de subimagen señalizada admite una mayor funcionalidad y/o

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

Opcionalmente, en cualquiera de los aspectos anteriores, se proporciona otra implementación del aspecto, en la que el indicador de nivel de subimagen se incluye en un mensaje SEI en el flujo de bits.

Opcionalmente, en cualquiera de los aspectos anteriores, se proporciona otra implementación del aspecto, en la que el indicador de nivel de subimagen se incluye en un SPS del flujo de bits.

Opcionalmente, en cualquiera de los aspectos anteriores, se proporciona otra implementación del aspecto, en donde el indicador de nivel de subimagen indica el tamaño de la subimagen, el recuento de píxeles de la subimagen, la velocidad de bits de la subimagen o combinaciones de los mismos.

Opcionalmente, en cualquiera de los aspectos anteriores, se proporciona otra implementación del aspecto, en donde el flujo de bits incluye un SPS que comprende los ID de subimagen para cada una de las subimágenes.

Opcionalmente, en cualquiera de los aspectos anteriores, se proporciona otra implementación del aspecto, en donde el SPS comprende además una ubicación de subimagen para cada una de las subimágenes.

Opcionalmente, en cualquiera de los aspectos anteriores, se proporciona otra implementación del aspecto, en donde el SPS comprende además un tamaño de subimagen para cada una de las subimágenes.

En un ejemplo, la descripción incluye un método implementado en un codificador, comprendiendo el método: dividir, mediante un procesador del codificador, una imagen en una pluralidad de subimágenes; codificar, mediante el procesador, una o más de la pluralidad de subimágenes en un flujo de bits; determinar, mediante el procesador, los requisitos de recursos para decodificar cada una de las una o más subimágenes; codificar en un flujo de bits, mediante el procesador, indicadores de nivel de subimágenes que indican los requisitos de recursos para decodificar una o más subimágenes; y almacenar, en una memoria del codificador, el flujo de bits para la comunicación hacia un decodificador. En algunos sistemas de codificación de vídeo, se señaliza un nivel para una imagen. Un nivel indica los recursos de hardware necesarios para decodificar la imagen. En algunos casos, diferentes subimágenes pueden tener una funcionalidad diferente en algunos casos y, por lo tanto, pueden tratarse de manera diferente durante el proceso de codificación. Como tal, un nivel basado en imágenes puede no ser útil para decodificar algunas subimágenes. Los presentes ejemplos son niveles de señal para cada subimagen. De esta manera, cada subimagen puede codificarse independientemente de otras subimágenes sin sobrecargar innecesariamente el decodificador al establecer requisitos de decodificación demasiado altos para las subimágenes codificadas de acuerdo con mecanismos menos complejos. La información a nivel de subimagen señalizada admite una mayor funcionalidad y/o una mayor eficiencia de codificación, lo que reduce el uso de recursos de red, recursos de memoria y/o recursos de procesamiento en el codificador y el decodificador.

Opcionalmente, en cualquiera de los aspectos anteriores, se proporciona otra implementación del aspecto, en donde los indicadores de nivel de subimagen se codifican en uno o más mensajes SEI en el flujo de bits.

Opcionalmente, en cualquiera de los aspectos anteriores, se proporciona otra implementación del aspecto, en donde los indicadores de nivel de subimagen se codifican en un SPS en el flujo de bits.

Opcionalmente, en cualquiera de los aspectos anteriores, se proporciona otra implementación del aspecto, en donde los indicadores de nivel de subimagen indican el tamaño de la subimagen, el recuento de píxeles de la subimagen, la velocidad de bits de la subimagen o combinaciones de los mismos.

Opcionalmente, en cualquiera de los aspectos anteriores, otra implementación del aspecto proporciona, que comprende además la codificación en el flujo de bits, por parte del procesador, de un SPS que comprende identificadores (ID) de subimagen para cada una de las subimágenes.

Opcionalmente, en cualquiera de los aspectos anteriores, otra implementación del aspecto proporciona, que comprende además la codificación en el flujo de bits, por parte del procesador, de un SPS que comprende una ubicación de subimagen para cada una de las subimágenes.

Opcionalmente, en cualquiera de los aspectos anteriores, otra implementación del aspecto proporciona, que comprende además la codificación en el flujo de bits, por parte del procesador, de un SPS que comprende un tamaño de subimagen para cada una de las subimágenes.

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

En una modalidad, la descripción incluye un medio no transitorio legible por ordenador que comprende un producto de programa informático para ser usado por un dispositivo de codificación de vídeo, comprendiendo el producto de programa informático instrucciones ejecutables por ordenador que están almacenadas en el medio no transitorio legible por ordenador, de manera que, cuando son ejecutadas por un procesador, hacen que el dispositivo de codificación de vídeo realice el método de cualquiera de los aspectos anteriores.

En un ejemplo, la descripción incluye un decodificador que comprende: un medio de recepción para recibir un flujo de bits que comprende una o más subimágenes divididas a partir de una imagen y un indicador del nivel de subimagen que indica los requisitos de recursos para decodificar la subimagen actual; un medio de análisis para analizar el flujo de bits para obtener el indicador de nivel de subimagen y la subimagen actual; un medio de asignación para asignar recursos para decodificar la subimagen actual basándose en el indicador de nivel de subimagen; un medio de decodificación para decodificar la subimagen actual para crear una secuencia de vídeo empleando los recursos asignados; y un medio de reenvío para reenviar la secuencia de vídeo para su visualización.

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.

En una modalidad, la descripción incluye un codificador que comprende: un medio de partición para dividir una imagen en una pluralidad de subimágenes; un medio de determinación para determinar los requisitos de recursos para decodificar cada una de las una o más subimágenes; un medio de codificación para: codificar en un flujo de bits indicadores a nivel de subimágenes que indican los requisitos de recursos para decodificar una o más subimágenes; y codificar una o más de la pluralidad de subimágenes en el flujo de bits; y un medio de almacenamiento para almacenar el flujo 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.

Para no dar lugar a dudas, cualquiera de las modalidades anteriores puede combinarse con una o más de las otras modalidades anteriores para crear una nueva modalidad dentro del alcance de la presente descripción.

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

Breve descripción de los dibujos

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

La figura 1 es un diagrama de flujo de un ejemplo de método para codificar una señal de vídeo.

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

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

La figura 4 es un diagrama esquemático que ilustra un ejemplo de codificador de vídeo.

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

La figura 6 es un diagrama esquemático que ilustra una imagen ejemplar dividida en subimágenes.

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

La figura 8 es un diagrama esquemático que ilustra otra imagen ejemplar dividida en subimágenes.

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

La figura 10 es un diagrama de flujo de un método ejemplar para codificar indicadores de nivel de subimágenes en un flujo de bits para admitir la decodificación de subimágenes.

La figura 11 es un diagrama de flujo de un procedimiento ejemplar para decodificar un flujo de bits de subimágenes basándose en indicadores de nivel de subimágenes.

La figura 12 es un diagrama esquemático de un sistema ejemplar para señalizar indicadores de nivel de subimagen a través de un flujo de bits.

Descripción detallada

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

En este documento se emplean varios acrónimos, como bloque de árbol de codificación (CTB), unidad de árbol de codificación (CTU), unidad de codificación (CU), secuencia de vídeo codificada (CVS), equipo conjunto de expertos en vídeo (JVET), conjunto de mosaicos con restricción de movimiento (MCTS), unidad de transferencia máxima (MTU), capa de abstracción de red (NAL), recuento del orden de las imágenes (POC), carga útil de secuencia de bytes sin procesar (RBSP), conjunto de parámetros (SPS), codificación de vídeo versátil (WC) 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 modalidad de una predicción espacial (por ejemplo, dentro de una imagen) y/o una predicción temporal (por ejemplo, entre imágenes) 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 árbol de codificación (CTB), unidades de árbol de codificación (CTU), unidades de codificación (CU) y/o nodos de codificación. Los bloques de vídeo en un segmento intracodificado (I) de una imagen se codifican usando predicción espacial con respecto a las muestras de referencia en bloques vecinos de la misma imagen. Los bloques de vídeo en un segmento de predicción unidireccional (P) o predicción bidireccional (B) intercodificado de una imagen pueden codificarse empleando la predicción espacial con respecto a las muestras de referencia en bloques vecinos en la misma imagen o la predicción temporal con respecto a las muestras de referencia en otras imágenes de referencia. Las imágenes pueden denominarse marcos y/o imágenes, y las imágenes de referencia pueden denominarse marcos de referencia y/o imágenes de referencia. La predicción espacial o temporal da como resultado un bloque predictivo que representa un bloque de imágenes. Los datos residuales representan las diferencias de píxeles entre el bloque de imagen original y el bloque predictivo. Por consiguiente, un bloque intercodificado se codifica de acuerdo con un vector de movimiento que apunta a un bloque de muestras de referencia que forman el bloque predictivo y los datos residuales que indican la diferencia entre el bloque codificado y el bloque predictivo. Un bloque intracodificado se codifica de acuerdo con un modo de intracodificación y los datos residuales. Para una mayor compresión, los datos residuales pueden transformarse del dominio de píxeles a un dominio de transformación. Esto da como resultado coeficientes de transformación residuales, que pueden cuantificarse. Los coeficientes de transformación cuantificados pueden disponerse inicialmente en una matriz bidimensional. Los coeficientes de transformación cuantificados pueden escanearse para producir un vector unidimensional de coeficientes de transformación. La codificación por entropía se puede aplicar para lograr incluso más compresión. Dichas técnicas de compresión de vídeo se describen con mayor detalle a continuación.

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

Para codificar una imagen de vídeo, la imagen se divide primero y las particiones se codifican en un flujo de bits. Están disponibles varios esquemas de partición de imágenes. Por ejemplo, una imagen se puede dividir en segmentos regulares, segmentos dependientes, mosaicos y/o de acuerdo con el procesamiento paralelo de frente de onda (WPP). Para simplificar, el HEVC restringe los codificadores de modo que solo se puedan usar 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 se puede aplicar para admitir la coincidencia del tamaño de la Unidad de Transferencia Máxima (MTU), el procesamiento en paralelo y la reducción del retraso de extremo a extremo. La MTU

indica la cantidad máxima de datos que se pueden transmitir en un solo paquete. Si la carga útil de un paquete supera la MTU, esa carga útil se divide en dos paquetes mediante un proceso denominado segmentación.

Una sección normal, también denominada simplemente sección, es una parte dividida de una imagen que puede reconstruirse independientemente de otras secciones regulares dentro de la misma imagen, a pesar de algunas interdependencias debidas a las operaciones de filtrado de bucles. Cada segmento regular se encapsula en su propia unidad de capa de abstracción de red (NAL) para su transmisión. Además, la predicción en imagen (predicción intramuestra, predicción de información de movimiento, predicción del modo de codificación) y la dependencia de la codificación por entropía a través de los límites de los segmentos pueden deshabilitarse para admitir la reconstrucción independiente. Esta reconstrucción independiente apoya la paralelización. Por ejemplo, la paralelización normal basada en segmentos emplea una comunicación mínima entre procesadores o entre núcleos. Sin embargo, como cada segmento normal es independiente, cada segmento está asociado a una cabecera de segmento independiente. El uso de segmentos regulares puede incurrir en una sobrecarga de codificación sustancial debido al coste en bits de la cabecera de segmento para cada segmento y debido a la falta de predicción a través de los límites de los segmentos. Además, se pueden emplear segmentos regulares para permitir la coincidencia con los requisitos de tamaño de la MTU. Específicamente, dado que un segmento normal está encapsulado en una unidad NAL separada y puede codificarse de forma independiente, cada segmento regular debe ser más pequeño que la MTU en los esquemas de MTU para evitar dividir el segmento en múltiples paquetes. Por lo tanto, el objetivo de la paralelización y el objetivo de la coincidencia del tamaño de la MTU pueden imponer exigencias contradictorias a la disposición de los segmentos en una imagen.

Los segmentos dependientes son similares a los segmentos normales, pero tienen encabezados de segmento acortados y permiten particionar los límites del bloque de árbol de la imagen sin interrumpir la predicción en la imagen. En consecuencia, las secciones dependientes permiten que una sección normal se fragmente en múltiples unidades NAL, lo que proporciona un retraso de extremo a extremo reducido al permitir que una parte de una sección normal se envíe antes de que se complete la codificación de toda la sección regular.

Un mosaico es una parte particionada de una imagen creada por límites horizontales y verticales que crean columnas y filas de mosaicos. Los mosaicos se pueden codificar en orden de escaneo rasterizado (de derecha a izquierda y de arriba a abajo). El orden de escaneo de los CTB es local dentro de un mosaico. En consecuencia, las CTB de un primer mosaico se codifican en orden de escaneo rasterizado, antes de pasar a las CTB del siguiente mosaico. Al igual que los segmentos normales, los mosaicos rompen las dependencias de predicción en la imagen, así como las dependencias de decodificación por entropía. Sin embargo, es posible que los mosaicos no se incluyan en las unidades NAL individuales y, por lo tanto, los mosaicos no se pueden usar para igualar el tamaño de la MTU. Cada mosaico puede procesarse mediante un procesador/núcleo, y la comunicación entre procesadores/entre núcleos empleada para la predicción en imagen entre las unidades de procesamiento que decodifican los mosaicos vecinos puede limitarse a transmitir una cabecera de segmento compartida (cuando las mosaicos adyacentes están en el mismo segmento) y a realizar el filtrado de bucles relacionado con el intercambio de muestras y metadatos reconstruidos. Cuando se incluye más de un mosaico en un segmento, el desplazamiento de bytes del punto de entrada para cada mosaico que no sea el primer desplazamiento del punto de entrada del segmento puede señalarse en la cabecera del segmento. Para cada segmento y mosaico, debe cumplirse al menos una de las siguientes condiciones: 1) todos los bloques de árbol codificados de un segmento pertenecen al mismo mosaico; y 2) todos los bloques de árbol codificados de 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 usar datos de los CTB en otras filas. El procesamiento en paralelo es posible mediante la decodificación en paralelo de las filas de CTB. Por ejemplo, una fila actual puede decodificarse en paralelo con una fila anterior. Sin embargo, dos CTB retrasan la decodificación de la fila actual con respecto al proceso de decodificación de las filas anteriores. Este retraso garantiza que los datos relacionados con el CTB anterior y el CTB superior 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 tantos procesadores o núcleos como filas CTB contenga la imagen. Debido a que se permite la predicción en imagen entre filas de bloques de árbol vecinas dentro de una imagen, la comunicación entre procesadores/entre núcleos para permitir la predicción en imagen puede ser sustancial. La partición de WPP considera los tamaños de las unidades NAL. Por lo tanto, WPP no admite la coincidencia de tamaños de MTU. Sin embargo, los segmentos normales se pueden usar junto con el WPP, con cierta sobrecarga de codificación, para implementar la coincidencia de tamaños de MTU según se desee.

Los mosaicos también pueden incluir conjuntos de mosaicos con movimiento restringido. Un conjunto de mosaicos con restricción de movimiento (MCTS) es un conjunto de mosaicos diseñado de tal manera que los vectores de movimiento asociados están restringidos para apuntar a ubicaciones de muestras completas dentro del MCTS y a ubicaciones de muestras fraccionadas que solo requieren ubicaciones de muestras completas dentro del MCTS para la interpolación. Además, no se permite el uso de candidatos a vectores de movimiento para la predicción temporal de vectores de movimiento derivados de bloques fuera del MCTS. De esta manera, cada MCTS puede decodificarse independientemente sin la existencia de mosaicos no incluidos en el MCTS. Los mensajes de información de mejora complementaria (SEI) de los MCTS temporales pueden usarse para indicar la existencia de MCTS en el flujo de bits y

señalizar los MCTS. El mensaje SEI del MCTS proporciona información complementaria que se puede usar en la extracción del subflujo de bits del MCTS (especificada como parte de la semántica del mensaje del SEI) para generar un flujo de bits conforme para un conjunto de MCTS. La información incluye varios conjuntos de información de extracción, cada uno de los cuales define un número de conjuntos de MCTS y contiene bytes sin procesar, bytes de carga útil de secuencia (RBSP) de los conjuntos de parámetros de vídeo (VPS), conjuntos de parámetros de secuencia (SPS) y conjuntos de parámetros de imagen (PPS) de reemplazo que se utilizarán durante el proceso de extracción del subflujo de bits del MCTS. Al extraer un subflujo de bits de acuerdo con el proceso de extracción del subflujo de bits del MCTS, los conjuntos de parámetros (VPS, SPS y PPS) pueden reescribirse o reemplazarse, y las cabeceras de los segmentos pueden actualizarse porque uno o todos los elementos sintácticos relacionados con las direcciones de los segmentos (incluidos first_slice_segment_in_pic_flag y slice_segment_address) pueden emplear valores diferentes en el subflujo de bits extraído.

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

Además, la división de imágenes puede basarse en mosaicos a nivel de imagen y mosaicos a nivel de secuencia. Los mosaicos de nivel de secuencia pueden incluir la funcionalidad del MCTS y pueden implementarse como subimágenes. Por ejemplo, un mosaico a nivel de imagen puede definirse como una región rectangular de bloques de árbol de codificación dentro de una columna de mosaicos en particular y una fila de mosaicos en particular en una imagen. Un mosaico a nivel de secuencia puede definirse como un conjunto de regiones rectangulares de bloques de árbol de codificación incluidos en diferentes tramas, donde cada región rectangular comprende además uno o más mosaicos a nivel de imagen y el conjunto de regiones rectangulares de bloques de árbol de codificación se puede decodificar independientemente de cualquier otro conjunto de regiones rectangulares similares. Un conjunto de grupos de mosaicos de nivel de secuencia (STOPS) es un grupo de dichos mosaicos de nivel de secuencia. El STGPS puede señalizarse en una unidad NAL que no es de capa de codificación de vídeo (VCL) con un identificador (ID) asociado en la cabecera de la unidad NAL.

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

En otro ejemplo, cuando cualquier subimagen de una imagen se codifica como una subimagen con movimiento temporal limitado, todas las subimágenes de la imagen pueden codificarse como subimágenes con movimiento temporal restringido. Tal división de imágenes puede ser limitante. Por ejemplo, codificar una subimagen como una subimagen temporal con movimiento limitado puede reducir la eficiencia de codificación a cambio de una funcionalidad adicional. Sin embargo, en la región de las aplicaciones basadas en intereses, normalmente solo una o unas pocas de las subimágenes utilizan una funcionalidad basada en subimágenes con restricción de movimiento temporal. Por lo tanto, las subimágenes restantes sufren de una eficiencia de codificación reducida sin proporcionar ningún beneficio práctico.

En otro ejemplo, los elementos sintácticos para especificar el tamaño de una subimagen pueden especificarse en unidades de tamaños de CTU de luminancia. En consecuencia, tanto el ancho como el alto de la subimagen deben ser un múltiplo entero de CtbSizeY. Este mecanismo de especificar el ancho y el alto de la subimagen puede ocasionar varios problemas. Por ejemplo, la partición de subimágenes solo es aplicable a imágenes con un ancho de imagen y/o un alto de imagen que sean un múltiplo entero de CtbSizeY. Esto hace que la partición de subimágenes no esté disponible para las imágenes que contienen dimensiones que no son múltiplos enteros de CtbSizeY. Si la partición de subimágenes se aplicara al ancho y/o alto de la imagen cuando la dimensión de la imagen no es un múltiplo entero de CtbSizeY, la derivación del ancho y/o el alto de la subimagen en las muestras de luminancia para la subimagen situada más a la derecha y la subimagen más abajo sería incorrecta. Tal derivación incorrecta provocaría resultados erróneos en algunas herramientas de codificación.

En otro ejemplo, la ubicación de una subimagen en una imagen puede no estar señalizada. En su lugar, la ubicación se obtiene mediante la siguiente regla. La subimagen actual se coloca en la siguiente ubicación desocupada de este tipo en el orden de escaneo rasterizado de la CTU dentro de una imagen que es lo suficientemente grande como para contener la subimagen dentro de los límites de la imagen. La derivación de ubicaciones de subimágenes de esta manera puede provocar errores en algunos casos. Por ejemplo, si una subimagen se pierde en la transmisión, entonces las ubicaciones de otras subimágenes se derivan incorrectamente y las muestras descodificadas se colocan en ubicaciones erróneas. El mismo problema se aplica cuando las subimágenes llegan en un orden incorrecto.

En otro ejemplo, la decodificación de una subimagen puede requerir la extracción de subimágenes colocalizadas en imágenes de referencia. Esto puede imponer una complejidad adicional y las consiguientes cargas en términos de uso de los recursos del procesador y la memoria.

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

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

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

En otro ejemplo, una ID de STGPS puede señalizarse en una cabecera de unidad NAL de una unidad NAL de VCL usando ocho bits. Esto puede ayudar a la extracción de imágenes secundarias. Dicha señalización puede aumentar innecesariamente la longitud de la cabecera de la unidad NAL. Otro problema es que, a menos que los conjuntos de grupos de mosaicos a nivel de secuencia estén restringidos para evitar superposiciones, un grupo de mosaicos puede estar asociado a múltiples conjuntos de grupos de mosaicos a nivel de secuencia.

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

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

En un tercer ejemplo, las subimágenes están restringidas para cubrir una imagen sin espacios ni superposición. Como se ha indicado anteriormente, algunos sistemas de codificación de vídeo permiten que las subimágenes incluyan

huecos y superposiciones. Esto crea la posibilidad de que los grupos/segmentos de mosaicos se asocien a múltiples subimágenes. Si esto está permitido en el codificador, los descodificadores deben construirse para soportar dicho esquema de codificación incluso cuando el esquema de decodificación se usa raramente. Al no permitir los huecos y solapamientos de subimágenes, la complejidad del decodificador puede reducirse, ya que no es necesario que el decodificador tenga en cuenta los posibles huecos y superposiciones al determinar los tamaños y ubicaciones de las subimágenes. Además, no permitir las brechas y superposiciones de subimagen reduce la complejidad de los procesos de optimización de la velocidad de distorsión (RDO) en el codificador, ya que el codificador puede omitir la consideración de los casos de separación y superposición al seleccionar una codificación para una secuencia de vídeo. En consecuencia, evitar huecos y superposiciones puede reducir el uso de recursos de memoria y/o recursos de procesamiento en el codificador y el decodificador.

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

En un quinto ejemplo, se señaliza un conjunto completo de ID de subimagen en el SPS, y las cabeceras de los segmentos incluyen un ID de subimagen que indica la subimagen que contiene los segmentos correspondientes. Como se ha indicado anteriormente, algunos sistemas señalizan las posiciones de subimágenes en relación con otras subimágenes. Esto causa un problema si las subimágenes se pierden o se extraen por separado. Al designar cada subimagen mediante un ID, las subimágenes pueden posicionarse y dimensionarse sin hacer referencia a otras subimágenes. Esto, a su vez, admite la corrección de errores, así como aplicaciones que solo extraen algunas de las subimágenes y evitan transmitir otras subimágenes. Se puede enviar una lista completa de todos los ID de subimágenes en el SPS junto con la información de tamaño relevante. Cada cabecera de segmento puede contener un ID de subimagen que indique la subimagen que incluye el segmento correspondiente. De esta manera, las subimágenes y los segmentos correspondientes pueden extraerse y posicionarse sin hacer referencia a otras subimágenes. Por lo tanto, los ID de subimagen admiten una mayor funcionalidad y/o una mayor eficiencia de codificación, lo que reduce el uso de recursos de red, recursos de memoria y/o recursos de procesamiento en el codificador y el decodificador.

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

La Figura 1 es un diagrama de flujo de un método operativo de ejemplo 100 para codificar una señal de vídeo. Específicamente, una señal de vídeo se codifica en un codificador. El proceso de codificación comprime la señal de vídeo empleando diversos mecanismos para reducir el tamaño de archivo de vídeo. Un tamaño de archivo más pequeño permite transmitir el archivo de vídeo comprimido a un usuario, al tiempo que se reduce la sobrecarga de ancho de banda asociada. El decodificador decodifica entonces el archivo de vídeo comprimido para reconstruir la señal de vídeo original para mostrarse a un usuario final. El proceso de decodificación en general refleja el proceso de codificación para permitir que el decodificador reconstruya sistemáticamente la señal de vídeo.

En un paso 101, la señal de vídeo se introduce en el codificador. Por ejemplo, la señal de vídeo puede ser un archivo de vídeo no comprimido almacenado en la memoria. Como otro ejemplo, el archivo de vídeo puede ser captado por un dispositivo de captación de vídeo, tal como una cámara de vídeo, y codificarse para admitir la emisión en continuo 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 cuadros de imagen que, cuando se ven en secuencia, dan la impresión visual de movimiento. Los cuadros contienen píxeles que se expresan en términos de luz, denominados en la presente memoria componentes de luminancia (o muestras de luminancia), y de color, que se denominan componentes de

crominancia (o muestras de color). En algunos ejemplos, las tramas también pueden contener valores de profundidad para permitir la visualización tridimensional.

En un paso 103, el vídeo se divide en bloques. La partición incluye subdividir los píxeles que hay en cada cuadro en unos bloques cuadrados y/o rectangulares para la compresión. Por ejemplo, en la codificación de vídeo de alta eficiencia (HEVC, por sus siglas en inglés) (también conocida como H.265 y MPEG-H, parte 2), el cuadro se puede dividir primero en unas unidades de árbol de codificación (CTU, por sus siglas en inglés), que son bloques de un tamaño predefinido (p. ej., sesenta y cuatro píxeles por sesenta y cuatro píxeles). Las unidades CTU contienen tanto muestras de luminancia como de crominancia. Pueden emplearse árboles de codificación para dividir las unidades CTU en bloques y luego subdividir los bloques reiteradamente hasta que se obtengan configuraciones que admitan más codificación. Por ejemplo, los componentes de luminancia de un cuadro pueden subdividirse hasta que los bloques individuales contengan unos valores de iluminación relativamente homogéneos. Además, los componentes de crominancia de un cuadro pueden subdividirse hasta que los bloques individuales contengan unos valores de color relativamente homogéneos. Por consiguiente, los mecanismos de partición varían dependiendo del contenido de los cuadros de vídeo.

En un paso 105 se emplean diversos mecanismos de compresión para comprimir los bloques de imagen divididos en el paso 103. Por ejemplo, puede emplearse una interpredicción y/o una intrapredicción. La interpredicción está diseñada para aprovecharse del hecho de que los objetos en una escena común tienden a aparecer en cuadros sucesivos. Por consiguiente, un bloque que representa un objeto en un cuadro de referencia no necesita describirse repetidamente en cuadros adyacentes. En concreto, un objeto, tal como una mesa, puede permanecer en posición constante en múltiples cuadros. Por lo tanto, la tabla se describe una vez, y los cuadros adyacentes pueden remitirse al cuadro de referencia. Pueden emplearse unos mecanismos de coincidencia de patrones para hacer coincidir objetos que aparecen en múltiples cuadros. Además, en múltiples cuadros pueden representarse objetos en movimiento debido a, por ejemplo, un movimiento de objeto o un movimiento de cámara. Como ejemplo particular, un vídeo puede mostrar un automóvil que se mueve de un lado a otro de la pantalla en múltiples cuadros. Se pueden emplear vectores de movimiento para describir un movimiento de este tipo. Un vector de movimiento es un vector bidimensional que proporciona un desplazamiento de las coordenadas de un objeto en un cuadro a las coordenadas del objeto en un cuadro de referencia. Así, la interpredicción puede codificar un bloque de imágenes en un cuadro actual como un conjunto de vectores de movimiento que indican un desplazamiento de un bloque correspondiente en un cuadro de referencia.

La intrapredicción codifica bloques en un cuadro común. La intrapredicción se aprovecha del hecho de que los componentes de luminancia y de crominancia tienden a agruparse en un cuadro. Por ejemplo, un área verde en una parte de un árbol tiende a colocarse adyacente a áreas verdes similares. La intrapredicción emplea múltiples modos direccionales de predicción (p. ej., treinta y tres en HEVC), un modo plano y un modo de corriente continua (c.c.). Los modos direccionales indican que un bloque actual es parecido/igual a las muestras de un bloque vecino en una dirección correspondiente. El modo plano indica que una serie de bloques a lo largo de una fila/columna (p. ej., un plano) puede interpolarse basándose en bloques vecinos que se encuentran en los bordes de la fila. En efecto, el modo plano indica una transición suave de luz/color a través de una fila/columna empleando una pendiente relativamente constante en valores cambiantes. El modo de c.c. se emplea para el alisado de límites e indica que un bloque es parecido/igual a un valor promedio asociado a las muestras de todos los bloques vecinos que están asociadas a las direcciones angulares de los modos de predicción direccional. Por consiguiente, los bloques de intrapredicción pueden representar los bloques de imágenes como diversos valores de modo de predicción relacional en vez de los valores reales. Además, los bloques de interpredicción pueden representar los bloques de imágenes como valores de vector de movimiento en vez de los valores reales. En cualquiera de los casos, los bloques de predicción pueden en algunos casos no representar exactamente los bloques de imagen. Cualquier diferencia se almacena en unos bloques residuales. A los bloques residuales se les pueden aplicar transformadas para comprimir aún más el archivo.

En un paso 107 se pueden aplicar diversas técnicas de filtrado. En la codificación HEVC, los filtros se aplican según un plan de filtrado en bucle. La predicción basada en bloques comentada anteriormente puede dar como resultado la creación de imágenes en bloques en el decodificador. Además, el plan de predicción basado en bloques puede codificar un bloque y luego reconstruir el bloque codificado para su uso posterior como bloque de referencia. El plan de filtrado en bucle aplica iterativamente filtros de supresión de ruido, filtros antibloque, filtros de bucle adaptativo y filtros de desplazamiento adaptativo de muestras (SAO, por sus siglas en inglés) a los bloques/cuadros. Estos filtros mitigan tales artefactos de efecto bloque para que el archivo codificado pueda reconstruirse con precisión. Además, estos filtros mitigan artefactos en los bloques de referencia reconstruidos para que sea menos probable que los artefactos creen artefactos adicionales en bloques posteriores que se codifiquen basándose en los bloques de referencia reconstruidos.

Una vez que la señal de vídeo se ha dividido, comprimido y filtrado, los datos resultantes se codifican en un flujo de bits en un paso 109. El flujo de bits incluye los datos comentados anteriormente, así como cualquier dato de señalización que se desee para admitir una reconstrucción de señales de vídeo adecuada en el decodificador. Por ejemplo, tales datos pueden incluir datos de partición, datos de predicción, bloques residuales y diversos indicadores que proporcionen instrucciones de codificación al decodificador. El flujo de bits se puede almacenar en la memoria

para su transmisión a un decodificador a petición. El flujo de bits también puede difundirse y/o multidifundirse a una pluralidad de decodificadores. La creación del flujo de bits es un proceso iterativo. En consecuencia, los pasos 101, 103, 105, 107 y 109 pueden tener lugar continua y/o simultáneamente en muchos cuadros y bloques. El orden mostrado en la figura 1 se presenta para mayor claridad y facilidad de análisis y no pretende limitar el proceso de codificación de vídeo a un orden en particular.

El decodificador recibe el flujo de bits y comienza el proceso de decodificación en un paso 111. Específicamente, el decodificador emplea un plan de decodificación entrópica para convertir el flujo de bits en unos correspondientes datos de sintaxis y de vídeo. El decodificador emplea los datos de sintaxis del flujo de bits para determinar las particiones para los cuadros en el paso 111. La partición debe coincidir con los resultados de la partición en bloques del paso 103. Ahora se describirá la codificación/decodificación entrópica empleada en el paso 111. El codificador toma muchas decisiones durante el proceso de compresión, tales como seleccionar planes de partición en bloques a partir de varias opciones posibles basándose en el posicionamiento espacial de los valores en la o las imágenes de entrada. La señalización de las opciones precisas puede requerir un gran número de contenedores. Tal y como se usa en la presente memoria, un contenedor es un valor binario que se trata como una variable (p. ej., un valor de bit, que puede variar dependiendo del contexto). La codificación entrópica permite que el codificador descarte cualquier opción que sea claramente inviable para un caso particular, lo que deja un conjunto de opciones admisibles. Cada opción admisible se asigna entonces una palabra clave. La longitud de las palabras clave se basa en el número de opciones admisibles (p. ej., un contenedor para dos opciones, dos contenedores para tres o cuatro opciones, etc.). El codificador codifica entonces la palabra clave para la opción seleccionada. Este plan reduce el tamaño de las palabras clave, ya que éstas son tan grandes como se desee para indicar de manera exclusiva una selección de entre un pequeño subconjunto de opciones admisibles en vez de indicar de manera exclusiva la selección de entre un conjunto potencialmente grande de todas las opciones posibles. A continuación, el decodificador decodifica la selección determinando el conjunto de opciones admisibles de una manera parecida a la del codificador. Al determinar el conjunto de opciones admisibles, el decodificador puede leer la palabra clave y determinar la selección realizada por el codificador.

En un paso 113, el decodificador realiza una decodificación de bloques. En concreto, el decodificador emplea transformadas inversas para generar unos bloques residuales. Luego, el decodificador emplea los bloques residuales y los correspondientes bloques de predicción para reconstruir los bloques de imagen según la partición. Los bloques de predicción pueden incluir tanto bloques de intrapredicción como bloques de interpredicción tal y como se generan en el codificador en el paso 105. Los bloques de imagen reconstruidos se colocan luego en unos cuadros de una señal de vídeo reconstruida según los datos de partición determinados en el paso 111. La sintaxis para el paso 113 también puede señalizarse en el flujo de bits por codificación entrópica, tal y como se comentó anteriormente.

En un paso 115 se realiza un filtrado de los cuadros de la señal de vídeo reconstruida de una manera parecida a la del paso 107 realizado en el codificador. Por ejemplo, a los cuadros se les pueden aplicar filtros de supresión de ruido, filtros antibloque, filtros de bucle adaptativo y filtros SAO para eliminar los artefactos de efecto bloque. Una vez que se filtran las tramas, la señal de vídeo puede emitirse a una pantalla en la etapa 117 para que la vea un usuario final.

La figura 2 es un diagrama esquemático de un sistema 200 de codificación y decodificación (códec) ejemplar para realizar una codificación de vídeo. Específicamente, el sistema de códec 200 proporciona una funcionalidad para soportar la implementación del método operativo 100. El sistema códec 200 se ha generalizado para representar unos componentes empleados tanto en un codificador como en un decodificador. El sistema códec 200 recibe y divide una señal de vídeo tal y como se comentó con respecto a los pasos 101 y 103 del método 100 de funcionamiento, lo que tiene como resultado unas señales de vídeo divididas 201. Cuando actúa como codificador, el sistema códec 200 comprime entonces la señal de vídeo dividida 201 y forma un flujo de bits codificado, tal y como se comentó con respecto a los pasos 105, 107 y 109 del método 100. Cuando actúa como decodificador, el sistema códec 200 genera una señal de vídeo de salida del flujo de bits, tal y como se comentó con respecto a los pasos 111, 113, 115 y 117 del método 100 de funcionamiento. El sistema códec 200 incluye un componente 211 general de control de codificador, un componente 213 de cuantificación y cambio de escala de transformada, un componente 215 de estimación intraimagen, un componente 217 de predicción intraimagen, un componente 219 de compensación del movimiento, un componente 221 de estimación del movimiento, un componente 229 de cambio de escala y transformada inversa, un componente 227 de análisis de control de filtros, 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 al contexto (CABAC, por sus siglas en inglés) y formateo de cabecera. Tales componentes están acoplados tal y como se muestra. En la figura 2, las líneas negras indican el movimiento de datos que hay codificar/decodificar, mientras que las líneas discontinuas indican el movimiento de datos de control que controlan el funcionamiento de otros componentes. Todos los componentes del sistema códec 200 pueden estar presentes en el codificador. El decodificador puede incluir un subconjunto de los componentes del sistema códec 200. Por ejemplo, el decodificador puede incluir el componente 217 de predicción intraimagen, el componente 219 de compensación del movimiento, el componente 229 de cambio de escala y transformada inversa, el componente 225 de filtros en bucle y el componente 223 de memoria intermedia de imágenes decodificadas. Ahora se describirán estos componentes.

La señal 201 de vídeo dividida es una secuencia de vídeo captada que ha sido dividida en unos bloques de píxeles por un árbol de codificación. Un árbol de codificación emplea diversos modos de escisión para subdividir un bloque de

píxeles en bloques de píxeles más pequeños. Estos bloques pueden subdividirse todavía más en bloques más pequeños. Los bloques pueden denominarse nodos en el árbol de codificación. Los nodos principales más grandes se escinden en nodos hijos más pequeños. El número de veces que un nodo se ha subdividido se denomina profundidad del nodo/árbol de codificación. Los bloques divididos pueden incluirse en algunos casos en unas unidades de codificación (CU). Por ejemplo, una unidad CU puede ser una subparte de una unidad CTU que contiene un bloque de luminancia, uno o unos bloques de crominancia de diferencia roja (Cr) y uno o unos bloques de crominancia de diferencia azul (Cb) junto con las correspondientes instrucciones de sintaxis para la unidad CU. Los modos de escisión pueden incluir un árbol binario (BT), un árbol triple (TT) y un árbol cuádruple (QT), que se emplean para dividir un nodo en dos, tres o cuatro nodos hijos, respectivamente, de formas variables dependiendo de los modos de escisión empleados. La señal 201 de vídeo dividida se reenvía al componente 211 general de control de codificador, el componente 213 de cuantificación y cambio de escala de transformada, el componente 215 de estimación intraimagen, el componente 227 de análisis de control de filtros y el componente 221 de estimación del movimiento para su compresión.

El componente 211 general de control de codificador está configurado para tomar decisiones relacionadas con la codificación de las imágenes de la secuencia de vídeo para formar el flujo de bits según las restricciones de aplicación. Por ejemplo, el componente 211 general de control de codificador gestiona la optimización de la velocidad binaria y el tamaño de flujo de bits frente a la calidad de reconstrucción. Tales decisiones se pueden realizar basándose en una disponibilidad de espacio de almacenamiento/ancho de banda y unas solicitudes de resolución de imágenes. El componente 211 general de control de codificador también gestiona la utilización de la memoria intermedia a la luz de la velocidad de transmisión para mitigar problemas de agotamiento y desbordamiento de memoria intermedia. Para gestionar estos problemas, el componente 211 general de control de codificador gestiona la partición, la predicción y el filtrado realizados por los demás componentes. Por ejemplo, el componente 211 general de control de codificador puede incrementar dinámicamente la complejidad de compresión para aumentar la resolución y aumentar el uso de ancho de banda o disminuir la complejidad de compresión para reducir la resolución y el uso de ancho de banda. Por lo tanto, el componente 211 general de control de codificador controla los demás componentes del sistema códec 200 para compensar la calidad de reconstrucción de señales de vídeo con temas de velocidad binaria. El componente 211 general de control de codificador crea unos datos de control con los que se controla el funcionamiento de los demás componentes. Los datos de control también se reenvían al componente 231 de codificación CABAC y formateo de cabecera para codificarse en el flujo de bits para indicar parámetros que hay que decodificar en el decodificador.

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

El componente 221 de estimación del movimiento y el componente 219 de compensación del movimiento pueden estar muy integrados, pero se ilustran separados a efectos conceptuales. La estimación del movimiento, realizada por el componente 221 de estimación del movimiento, es el proceso de generación de vectores de movimiento, con los que se estima el movimiento para bloques de vídeo. Por ejemplo, un vector de movimiento puede indicar el desplazamiento de un objeto codificado en relación con un bloque predictivo. Un bloque predictivo es un bloque que se ha visto que coincide estrechamente con el bloque que hay que codificar en términos de diferencia de píxeles. Un bloque predictivo también puede denominarse bloque de referencia. Tal diferencia de píxeles pueden determinarse mediante una suma de las diferencias absolutas (SAD, por sus siglas en inglés), la suma de las diferencias cuadradas (SSD, por sus siglas en inglés) u otras métricas de diferencia. En la codificación HEVC se emplean varios objetos codificados, incluidos una unidad CTU, bloques de árbol de codificación (CTB, por sus siglas en inglés) y unidades CU. Por ejemplo, una unidad CTU se puede dividir en bloques CTB, que luego se pueden dividir en bloques CB para su inclusión en las unidades CU. Una unidad CU puede codificarse como una unidad de predicción (PU, por sus siglas en inglés) que contiene datos de predicción y/o como una unidad de transformación (TU) que contiene datos residuales transformados para la unidad CU. El componente 221 de estimación del movimiento genera vectores de movimiento, unidades PU y unidades TU usando un análisis de velocidad-distorsión como parte de un proceso de optimización de la distorsión de la velocidad. Por ejemplo, el componente 221 de estimación del movimiento puede determinar múltiples bloques de referencia, múltiples vectores de movimiento, etc., para un bloque/cuadro actual y puede seleccionar los bloques de referencia, vectores de movimiento, etc., que tienen las mejores características de velocidad-distorsión. Las mejores características de velocidad-distorsión compensan la calidad de reconstrucción de vídeo (p. ej., la cantidad de pérdida de datos por compresión) con el rendimiento de codificación (p. ej., el tamaño de la codificación final).

En algunos ejemplos, el sistema códec 200 puede calcular valores para unas posiciones de píxel subentero de unas imágenes de referencia almacenadas en el componente 223 de memoria intermedia de imágenes decodificadas. Por ejemplo, el sistema códec 200 puede interpolar los valores de posiciones de un cuarto de píxel, posiciones de un octavo de píxel u otras posiciones de píxel fraccionario de la imagen de referencia. Por lo tanto, el componente 221 de estimación del movimiento puede realizar una búsqueda de movimiento en relación con las posiciones de píxel

completo y las posiciones de píxel fraccionario y generar un vector de movimiento con una precisión de píxel fraccionaria. El componente 221 de estimación del movimiento calcula un vector de movimiento para una unidad PU de un bloque de vídeo en un trozo intercodificado comparando la posición de la unidad PU con la posición de un bloque predictivo de una imagen de referencia. El componente 221 de estimación del movimiento da salida al vector de movimiento calculado como datos de movimiento al componente 231 de CABAC y formateo de cabecera para codificarlos y moverlos al componente 219 de compensación del movimiento.

La compensación de movimiento, realizada por el componente 219 de compensación del movimiento, puede conllevar obtener o generar el bloque predictivo basándose en el vector de movimiento determinado por el componente 221 de estimación del movimiento. De nuevo, el componente 221 de estimación del movimiento y el componente 219 de compensación del movimiento pueden estar en algunos ejemplos integrados funcionalmente. Una vez recibido el vector de movimiento para la unidad PU del bloque de vídeo actual, el componente 219 de compensación del movimiento puede localizar el bloque predictivo al que apunta el vector de movimiento. Luego se forma un bloque de vídeo residual restando valores de píxel del bloque predictivo de los valores de píxel del bloque de vídeo actual que se está codificando, lo que da lugar a valores de diferencia de píxel. Generalmente, el componente 221 de estimación del movimiento realiza una estimación del movimiento en relación con componentes de luminancia, y el componente 219 de compensación del movimiento usa los vectores de movimiento calculados basándose en los componentes de luminancia tanto para los componentes de crominancia como para los componentes de luminancia. El bloque predictivo y el bloque residual se reenvían al componente 213 de cuantificación y cambio de escala de transformada.

La señal 201 de vídeo dividida también se envía al componente 215 de estimación intraimagen y al componente 217 de predicción intraimagen. Al igual que ocurre con el componente 221 de estimación del movimiento y el componente 219 de compensación del movimiento, el componente 215 de estimación intraimagen y el componente 217 de predicción intraimagen pueden estar muy integrados, pero se ilustran separados a efectos conceptuales. El componente 215 de estimación intraimagen y el componente 217 de predicción intraimagen intrapredicen un bloque actual en relación con bloques en un cuadro actual como alternativa a la interpredicción realizada por el componente 221 de estimación del movimiento y el componente 219 de compensación del movimiento entre cuadros, tal y como se describió anteriormente. En particular, el componente 215 de estimación intraimagen determina un modo de intrapredicción a usar para codificar un bloque actual. En algunos ejemplos, el componente 215 de estimación intraimagen selecciona un modo de intrapredicción apropiado para codificar un bloque actual de entre múltiples modos de intrapredicción probados. Los modos de intrapredicción seleccionados se reenvían luego al componente 231 de CABAC y formateo de cabecera para la codificación.

Por ejemplo, el componente 215 de estimación intraimagen calcula valores de velocidad-distorsión usando un análisis de velocidad-distorsión para los diversos modos de intrapredicción probados y selecciona el modo de intrapredicción que tiene las mejores características de velocidad-distorsión de entre los modos probados. El análisis de velocidad-distorsión generalmente determina una cantidad de distorsión (o de error) entre un bloque codificado y un bloque original no codificado que se codificó para producir el bloque codificado, así como una velocidad binaria (p. ej., un número de bits) usada para producir el bloque codificado. El componente 215 de estimación intraimagen puede calcular unas relaciones a partir de las distorsiones y las velocidades para los diversos bloques codificados para determinar qué modo de intrapredicción presenta el mejor valor de velocidad-distorsión para el bloque. Además, el componente 215 de estimación intraimagen puede configurarse para codificar bloques de profundidad de un mapa de profundidad usando un modo de modelado de profundidad (DMM, por sus siglas en inglés) basado en una optimización de velocidad-distorsión (RDO, por sus siglas en inglés).

El componente 217 de predicción intraimagen puede, cuando se implementa en un codificador, generar un bloque residual a partir del bloque predictivo basándose en los modos de intrapredicción seleccionados determinados por el componente 215 de estimación intraimagen o, cuando se implementa en un decodificador, leer el bloque residual del flujo de bits. El bloque residual incluye la diferencia de valores entre el bloque predictivo y el bloque original, representada como una matriz. El bloque residual se reenvía entonces al componente 213 de cuantificación y cambio de escala de transformada. El componente 215 de estimación intraimagen y el componente 217 de predicción intraimagen pueden trabajar tanto sobre componentes de luminancia como sobre componentes de crominancia.

El componente 213 de cuantificación y cambio de escala de transformada está configurado para comprimir todavía más el bloque residual. El componente 213 de cuantificación y cambio de escala de transformada aplica una transformada, tal como una transformada de coseno discreta (DCT, por sus siglas en inglés), una transformada de seno discreta (DST, por sus siglas en inglés) o una transformada conceptualmente similar, al bloque residual, lo que produce un bloque de vídeo que comprende valores de coeficiente de transformada residuales. También podrían usarse transformadas de ondícula, transformadas de números enteros, transformadas de subbanda u otros tipos de transformadas. La transformada puede convertir la información residual de un dominio de valor de píxel a un dominio de la transformada, tal como un dominio de la frecuencia. El componente 213 de cuantificación y cambio de escala de transformada también está configurado para cambiar la escala de la información residual transformada, por ejemplo, basándose en la frecuencia. Tal cambio de escala conlleva aplicar un factor de escala a la información residual para que una información de frecuencia diferente se cuantifique a distintas granularidades, lo que puede afectar a la calidad visual final del vídeo reconstruido. El componente 213 de cuantificación y cambio de escala de transformada también está configurado para cuantificar los coeficientes de transformada para reducir aún más la velocidad binaria. 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 cuantificación y cambio de escala de transformada puede entonces realizar una exploración de la matriz que incluye los coeficientes de transformada cuantificados. Los coeficientes de transformada cuantificados se reenvían al componente 231 de CABAC y formateo de cabecera para codificarse en el flujo de bits.

El componente 229 de cambio de escala y transformada inversa aplica una operación inversa a la del componente 213 de cuantificación y cambio de escala de transformada para admitir la estimación del movimiento. El componente 229 de cambio de escala y transformada inversa aplica una escala, transformación y/o cuantificación inversa(s) para reconstruir el bloque residual en el dominio de píxeles, por ejemplo, para su uso posterior como bloque de referencia que puede convertirse en un bloque predictivo para otro bloque actual. El componente 221 de estimación del movimiento y/o el componente 219 de compensación del movimiento puede(n) calcular un bloque de referencia añadiendo de nuevo el bloque residual a un correspondiente bloque predictivo para usarse en la estimación del movimiento de un bloque/cuadro posterior. A los bloques de referencia reconstruidos se les aplican unos filtros para mitigar los artefactos creados durante el cambio de escala, la cuantificación y las transformadas. De lo contrario, tales artefactos podrían causar una predicción imprecisa (y crear artefactos adicionales) cuando se predigan bloques posteriores.

El componente 227 de análisis de control de filtros y el componente 225 de filtros en bucle aplican los filtros a los bloques residuales y/o a bloques de imagen reconstruidos. Por ejemplo, el bloque residual transformado procedente del componente 229 de cambio de escala y transformada inversa puede combinarse con un correspondiente bloque de predicción procedente del componente 217 de predicción intraimagen y/o del componente 219 de compensación del movimiento para reconstruir el bloque de imagen original. Los filtros pueden entonces aplicarse al bloque de imagen reconstruido. En algunos ejemplos, los filtros pueden, en cambio, aplicarse a los bloques residuales. Al igual que ocurre con otros componentes en la figura 2, el componente 227 de análisis de control de filtros y el componente 225 de filtros en bucle están muy integrados y pueden implementarse juntos, pero se representan separados a efectos conceptuales. Los filtros aplicados a los bloques de referencia reconstruidos se aplican a unas regiones espaciales particulares e incluyen múltiples parámetros para ajustar cómo se aplican tales filtros. El componente 227 de análisis de control de filtros analiza los bloques de referencia reconstruidos para determinar dónde deberían aplicarse tales filtros y establece unos parámetros correspondientes. Tales datos se reenvían al componente 231 de CABAC y formateo de cabecera como datos de control de filtro para la codificación. El componente 225 de filtros en bucle aplica tales filtros basándose en los datos de control de filtro. Los filtros pueden incluir un filtro antibloque, un filtro de supresión de ruido, un filtro SAO y un filtro de bucle adaptativo. Dependiendo del ejemplo, tales filtros pueden aplicarse en el dominio del espacio/píxeles (p. ej., en un bloque de píxeles reconstruido) o en el dominio de la frecuencia.

Cuando funciona como 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 del movimiento, tal y como se comentó anteriormente. Cuando funciona como decodificador, el componente 223 de memoria intermedia de imágenes decodificadas almacena y reenvía los bloques reconstruidos y filtrados a un visualizador como parte de una señal de vídeo de salida. El componente 223 de memoria intermedia de imágenes decodificadas puede ser cualquier dispositivo de memoria que sea capaz de almacenar bloques de predicción, bloques residuales y/o bloques de imagen reconstruidos.

El componente 231 de CABAC y formateo de cabecera recibe los datos procedentes de los diversos componentes del sistema códec 200 y codifica tales datos en un flujo de bits codificado para su transmisión a un decodificador. En concreto, el componente 231 de CABAC y formateo de cabecera genera diversas cabeceras para codificar unos datos de control, tales como datos de control generales y datos de control de filtro. Además, los datos de predicción, incluidos los datos de intrapredicción y de movimiento, así como los datos residuales en forma de datos de coeficiente de transformada cuantificados se codifican todos en el flujo de bits. El flujo de bits final incluye toda la información que quiere el decodificador para reconstruir la señal 201 de vídeo dividida original. Tal información puede también incluir unas tablas de índice de modo de intrapredicción (también denominadas tablas de mapeo de palabras clave), definiciones de contextos de codificación para diversos bloques, indicaciones de los modos de intrapredicción más probables, una indicación de información de partición, etc. Tales datos pueden codificarse por codificación entrópica. Por ejemplo, la información puede codificarse por codificación de longitud variable adaptativa al contexto (CAVLC, por sus siglas en inglés), codificación CABAC, codificación aritmética binaria adaptativa al contexto y basada en la sintaxis (SBAC, por sus siglas en inglés), codificación entrópica de partición en intervalos probabilísticos (PIPE, por sus siglas en inglés) u otra técnica de codificación entrópica. Después de la codificación entrópica, el flujo de bits codificado puede transmitirse a otro dispositivo (p. ej., un decodificador de vídeo) o archivarse para su posterior transmisión o recuperación.

La figura 3 es un diagrama de bloques que ilustra un codificador de vídeo ejemplar 300. El codificador 300 de vídeo puede emplearse para implementar las funciones de codificación del sistema códec 200 y/o implementar los pasos 101, 103, 105, 107 y/o 109 del método de funcionamiento 100. El codificador 300 divide una señal de vídeo de entrada, lo que da como resultado una señal de vídeo dividida 301 que es sustancialmente similar a la señal de vídeo dividida 201. La señal 301 de vídeo dividida es comprimida y codificada a continuación por componentes del codificador 300 para formar un flujo de bits.

En concreto, la señal 301 de vídeo dividida se reenvía a un componente 317 de predicción intraimagen para una intrapredicción. El componente 317 de predicción intraimagen puede ser sustancialmente similar al componente 215 de estimación intraimagen y al componente 217 de predicción intraimagen. La señal 301 de vídeo dividida también se reenvía a un componente 321 de compensación del movimiento para una interpredicción basada en unos bloques de referencia que están en un componente 323 de memoria intermedia de imágenes decodificadas. El componente 321 de compensación del movimiento puede ser sustancialmente similar al componente 221 de estimación del movimiento y al componente 219 de compensación del movimiento. Los bloques de predicción y los bloques residuales procedentes del componente 317 de predicción intraimagen y del componente 321 de compensación del movimiento se reenvían a un componente 313 de transformada y cuantificación para transformar y cuantificar los bloques residuales. El componente 313 de transformada y cuantificación puede ser sustancialmente similar al componente 213 de escalado y cuantificación de transformada. Los bloques residuales transformados y cuantificados y los correspondientes bloques de predicción (junto con datos de control asociados) se reenvían a un componente 331 de codificación entrópica para codificarse en un flujo de bits. El componente 331 de codificación entrópica puede ser sustancialmente similar al componente 231 de CABAC y formateo de cabecera.

Los bloques residuales transformados y cuantificados y/o los correspondientes bloques de predicción también se reenvían del componente 313 de transformada y cuantificación a un componente 329 de transformada inversa y cuantificación para su reconstrucción en bloques de referencia para su uso por el componente 321 de compensación del movimiento. El componente 329 de transformada inversa y cuantificación puede ser sustancialmente similar al componente 229 de cambio de escala y transformada inversa. Dependiendo del ejemplo, a los bloques residuales y/o a los bloques de referencia reconstruidos también se les aplican unos filtros en bucle en un componente 325 de filtros en bucle. 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, tal y como se comentó con respecto al componente 225 de filtros en bucle. Los bloques filtrados se almacenan entonces en un componente 323 de memoria intermedia de imágenes decodificadas para su uso como bloques de referencia por el componente 321 de compensación del movimiento. El componente 323 de memoria intermedia de imágenes decodificadas puede ser sustancialmente similar al componente 223 de memoria intermedia de imágenes decodificadas.

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

El flujo de bits es recibido por un componente de decodificación entrópica 433. El componente 433 de decodificación entrópica está configurado para implementar un plan de decodificación entrópica, tal como una codificación CAVLC, CABAC, SBAC u PIPE u otras técnicas de codificación entrópica. Por ejemplo, el componente 433 de decodificación entrópica puede emplear información de cabecera para proporcionar un contexto para interpretar datos adicionales codificados como palabras clave en los flujos de bits. La información decodificada incluye cualquier información requerida para decodificar la señal de vídeo, tal como datos de control generales, datos de control de filtro, información de partición, datos de movimiento, datos de predicción y coeficientes de transformada cuantificados procedentes 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 intraimagen para su reconstrucción en bloques de imágenes basándose en operaciones de intrapredicción. El componente 417 de predicción intraimagen puede ser similar al componente 215 de estimación intraimagen y al componente 217 de predicción intraimagen. En concreto, el componente 417 de predicción intraimagen emplea modos de predicción para localizar un bloque de referencia en el cuadro y aplica un bloque residual al resultado para reconstruir unos bloques de imágenes intrapredichos. Los bloques de imagen intrapredichos reconstruidos y/o los bloques residuales y los correspondientes datos de interpredicción 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 imagen reconstruidos, los bloques residuales y/o los bloques de predicción, y tal información se almacena en el componente 423 de memoria intermedia de imágenes decodificadas. Los bloques de imagen reconstruidos procedentes del componente 423 de memoria intermedia de imágenes decodificadas se reenvían a un componente 421 de compensación del movimiento para la interpredicción. El componente 421 de compensación del movimiento puede ser sustancialmente similar al componente 221 de estimación del movimiento y/o al componente 219 de compensación del movimiento. En concreto, el componente 421 de compensación del movimiento emplea vectores de movimiento procedentes 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 sigue almacenando bloques de imagen reconstruidos adicionales, que pueden reconstruirse para formar cuadros a través de la información de partición. Tales cuadros también se pueden colocar formando una secuencia. Se da salida a la secuencia hacia un visualizador como una señal de vídeo de salida reconstruida.

La figura 5 es un diagrama esquemático que ilustra un flujo de bits 500 y un subflujo de bits 501 de ejemplo extraídos del flujo de bits 500. Por ejemplo, el flujo de bits 500 puede generarse mediante un sistema de códec 200 y/o un codificador 300 para su decodificación mediante un sistema de códec 200 y/o un decodificador 400. Como otro ejemplo, el flujo de bits 500 puede generarse mediante un codificador en la etapa 109 del procedimiento 100 para su uso por un decodificador en la etapa 111.

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

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

Cada imagen 521 puede contener un conjunto completo de datos visuales asociados con una secuencia de vídeo para un instante temporal correspondiente. Sin embargo, es posible que ciertas aplicaciones deseen mostrar solo una parte de una imagen 521 en algunos casos. Por ejemplo, un sistema de realidad virtual (VR) puede mostrar una región seleccionada por el usuario de la imagen 521, lo que crea la sensación de estar presente en la escena representada en la imagen 521. La región que un usuario puede desear ver no se conoce cuando se codifica el flujo de bits 500. En consecuencia, la imagen 521 puede contener cada región posible que un usuario pueda ver potencialmente como subimágenes 522, que pueden decodificarse y mostrarse por separado en función de la entrada del usuario. Otras aplicaciones pueden mostrar por separado una región de interés. Por ejemplo, un televisor con una imagen en una imagen puede desear mostrar una región particular y, por lo tanto, una subimagen 522, a partir de una secuencia de vídeo sobre una imagen 521 de una secuencia de vídeo no relacionada. En otro ejemplo más, los sistemas de teleconferencia pueden mostrar una imagen completa 521 de un usuario que está hablando actualmente y una subimagen 522 de un usuario que no está hablando actualmente. En consecuencia, una subimagen 522 puede contener una región definida de la imagen 521. Una subimagen 522 que está temporalmente restringida al movimiento puede descodificarse por separado del resto de la imagen 521. Específicamente, una subimagen de movimiento temporal limitado se codifica sin referencia a muestras fuera de la subimagen de movimiento temporal y, por lo tanto, contiene información suficiente para una decodificación completa sin referencia al resto de la imagen 521.

Cada sección 524 puede ser un rectángulo definido por una CTU en una esquina superior izquierda y una CTU en una esquina inferior derecha. En algunos ejemplos, un segmento 524 incluye una serie de mosaicos y/o CTU en un orden de escaneo de trama que avanza de izquierda a derecha y de arriba a abajo. En otros ejemplos, una sección 524 es una sección rectangular. Es posible que un segmento rectangular no atraviese todo el ancho de una imagen según el orden de escaneo rasterizado. En cambio, un segmento rectangular puede contener una región rectangular y/o cuadrada de una imagen 521 y/o una subimagen 522 definida en términos de una CTU y/o filas de mosaicos y una

CTU y/o columnas de mosaicos. Un segmento 524 es la unidad más pequeña que un decodificador puede mostrar por separado. Por lo tanto, los segmentos 524 de una imagen 521 pueden asignarse a diferentes subimágenes 522 para representar por separado las regiones deseadas de una imagen 521.

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

La presente descripción señala varios datos para soportar la codificación eficiente de las subimágenes 522 para la selección y visualización de las subimágenes 523 en el decodificador. El SPS 510 incluye un tamaño de subimagen 531, una ubicación de subimagen 532 y unos ID de subimagen 533 relacionados con el conjunto completo de subimágenes 522. El tamaño de subimagen 531 incluye una altura de subimagen en muestras de luminancia y una anchura de subimagen en muestras de luminancia para una subimagen 522 correspondiente. La ubicación 532 de la subimagen incluye una distancia de compensación entre una muestra superior izquierda de una subimagen 522 correspondiente y una muestra superior izquierda de la imagen 521. La ubicación 532 de la subimagen y el tamaño 531 de la subimagen definen un diseño de la subimagen 522 correspondiente. La ID de subimagen 533 contiene datos que identifican de manera única una subimagen 522 correspondiente. El ID de subimagen 533 puede ser un índice de escaneo rasterizado de subimagen 522 u otro valor definido. Por lo tanto, un decodificador puede leer el SPS 510 y determinar el tamaño, la ubicación y el ID de cada subimagen 522. En algunos sistemas de codificación de vídeo, los datos relacionados con las subimágenes 522 pueden incluirse en el PPS 512 porque una subimagen 522 se divide a partir de una imagen 521. Sin embargo, las particiones utilizadas para crear subimágenes 522 pueden ser utilizadas por aplicaciones, tales como aplicaciones basadas en ROI, aplicaciones de realidad virtual, etc., que dependen de particiones de subimágenes 522 consistentes en una secuencia/segmento de vídeo. Como tal, las particiones 522 de la subimagen generalmente no cambian por imagen. La colocación de la información de diseño para las subimágenes 522 en el SPS 510 garantiza que el diseño solo se señalice una vez para una secuencia/segmento en lugar de señalizarse de forma redundante para cada PPS 512 (que puede señalizarse para cada imagen 521 en algunos casos). Además, la señalización de la información de la subimagen 522, en lugar de depender del decodificador para obtener dicha información, reduce la posibilidad de error en caso de pérdida de paquetes y admite una funcionalidad adicional en términos de extraer las subimágenes 523. En consecuencia, el diseño de la subimagen 522 de señalización en la SPS 510 mejora la funcionalidad de un codificador y/o decodificador.

La SPS 510 también contiene indicadores 534 de subimágenes con restricción de movimiento relacionados con el conjunto completo de subimágenes 522. Los indicadores 534 de subimágenes de movimiento restringido indican si cada subimagen 522 es una subimagen de movimiento temporal restringido. Por lo tanto, el decodificador puede leer los indicadores 534 de subimágenes de movimiento restringido y determinar cuáles de las subimágenes 522 pueden extraerse y mostrarse por separado sin decodificar otras subimágenes 522. Esto permite que las subimágenes 522 seleccionadas se codifiquen como subimágenes con restricciones de movimiento temporal, al tiempo que permite que otras subimágenes 522 se codifiquen sin tales restricciones para aumentar la eficiencia de codificación.

Los ID de subimagen 533 también se incluyen en las cabeceras de segmento 514. Cada cabecera de segmento 514 contiene datos relevantes para un conjunto correspondiente de segmentos 524. En consecuencia, la cabecera de segmento 514 contiene solo los ID de subimagen 533 correspondientes a los segmentos 524 asociados a la cabecera de segmento 514. Como tal, un decodificador puede recibir un segmento 524, obtener un ID de subimagen 533 de la cabecera de segmento 514 y determinar qué subimagen 522 contiene el segmento 524. El decodificador también puede usar el ID de subimagen 533 de la cabecera de segmento 514 para correlacionar con los datos relacionados en el SPS 510. Como tal, el decodificador puede determinar cómo posicionar las subimágenes 522/523 y los segmentos 524/525 leyendo el SPS 510 y las cabeceras de segmento 514 relevantes. Esto permite decodificar las subimágenes 523 y los segmentos 525 incluso si algunas subimágenes 522 se pierden en la transmisión o se omiten deliberadamente para aumentar la eficiencia de codificación.

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

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

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

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

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

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

El mecanismo 700 se puede aplicar a los segmentos 724 en una subimagen 722, tal como los segmentos 524/525 y las subimágenes 522/523, respectivamente. En el ejemplo mostrado, la subimagen 722 incluye una primera sección 724a, una segunda sección 724b y una tercera sección 724c. Las cabeceras de segmento para cada uno de los segmentos 724 incluyen un ID de subimagen 733 para la subimagen 722. El decodificador puede hacer coincidir el ID de subimagen 733 de la cabecera del segmento con el ID de subimagen 733 en el SPS. El decodificador puede entonces determinar la ubicación 732 y el tamaño de la subimagen 722 a partir del SPS basándose en el ID de subimagen 733. Usando la ubicación 732, la subimagen 722 se puede colocar en relación con la muestra superior izquierda en la esquina superior izquierda 742 de la imagen. El tamaño se puede usar para establecer la altura y el ancho de la subimagen 722 con respecto a la ubicación 732. Los segmentos 724 se pueden incluir entonces en la subimagen 722. En consecuencia, los segmentos 724 se pueden colocar en la subimagen correcta 722 basándose en el ID de subimagen 733 sin hacer referencia a otras subimágenes. Esto admite la corrección de errores, ya que otras subimágenes perdidas no alteran la decodificación de la subimagen 722. Esto también admite aplicaciones que solo extraen una subimagen 722 y evitan transmitir otras subimágenes. Por lo tanto, los ID de subimagen 733 admiten una mayor funcionalidad y/o una mayor eficiencia de codificación, lo que reduce el uso de recursos de red, recursos de memoria y/o recursos de procesamiento en el codificador y el decodificador.

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

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

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

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

La información relacionada con las subimágenes que pueden estar presentes en la secuencia de vídeo codificada (CVS) puede señalizarse en un conjunto de parámetros de nivel de secuencia, tal como un SPS. Dicha señalización puede incluir la siguiente información. El número de subimágenes que están presentes en cada imagen del CVS puede señalizarse en el SPS. En el contexto del SPS o un CVS, las subimágenes colocadas para todas las unidades de acceso (AU) pueden denominarse colectivamente secuencia de subimágenes. También se puede incluir en el SPS un bucle para especificar adicionalmente la información que describe las propiedades de cada subimagen. Esta información puede comprender la identificación de la subimagen, la ubicación de la subimagen (por ejemplo, la distancia de compensación entre la muestra de luminancia de la esquina superior izquierda de la subimagen y la muestra de luminancia de la esquina superior izquierda de la imagen) y el tamaño de la subimagen. Además, el SPS puede indicar si cada subimagen es una subimagen con restricciones de movimiento (que contiene la funcionalidad de un MCTS). La información de perfil, nivel y nivel para cada subimagen también puede señalizarse o derivarse en el decodificador. Dicha información puede emplearse para determinar la información de perfil, nivel y nivel para un flujo de bits creado extrayendo subimágenes del flujo de bits original. El perfil y el nivel de cada subimagen pueden derivarse para que sean los mismos que el perfil y el nivel de todo el flujo de bits. El nivel para cada subimagen puede señalizarse explícitamente. Dicha señalización puede estar presente en el bucle contenido en el SPS. Los parámetros del decodificador de referencia hipotético (HRD) a nivel de secuencia pueden señalizarse en la sección de información de usabilidad de vídeo (VUI) del SPS para cada subimagen (o, de manera equivalente, cada secuencia de subimagen).

Cuando una imagen no se divide en dos o más subimágenes, las propiedades de la subimagen (por ejemplo, ubicación, tamaño, etc.), excepto el ID de subimagen, pueden no estar presentes o señalizadas en el flujo de bits. Cuando se extrae una subimagen de imágenes en una CVS, cada unidad de acceso del nuevo flujo de bits puede no contener subimágenes. En este caso, la imagen en cada AU del nuevo flujo de bits no se divide en múltiples subimágenes. Por lo tanto, no es necesario señalizar las propiedades de la subimagen, como la ubicación y el tamaño, en el SPS, ya que dicha información puede derivarse de las propiedades de la imagen. Sin embargo, la identificación de la subimagen aún puede señalizarse, ya que las unidades NAL/grupos de mosaicos de VCL que están incluidos en

la subimagen extraída pueden hacer referencia a la ID. Esto puede permitir que los ID de subimagen permanezcan iguales al extraer la subimagen.

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

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

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

Puede estar restringido el hecho de que la unión de subimágenes debe cubrir toda la imagen sin huecos y sin superposición. Cuando se aplica esta restricción, para cada subimagen, puede estar presente un indicador para especificar si la subimagen es una subimagen con movimiento restringido, lo que indica que la subimagen se puede extraer. Alternativamente, la unión de subimágenes puede no cubrir toda la imagen, pero es posible que no se permitan las superposiciones.

Los ID de subimagen pueden estar presentes inmediatamente después del encabezado de la unidad NAL para ayudar al proceso de extracción de subimágenes sin requerir que el extractor analice el resto de los bits de la unidad NAL. Para las unidades NAL de VCL, el ID de subimagen puede estar presente en los primeros bits de las cabeceras de los grupos de mosaicos. Para una unidad NAL que no sea VCL, puede aplicarse lo siguiente. Para el SPS, el ID de la subimagen no necesita estar presente inmediatamente después del encabezado de la unidad NAL. Para el PPS, si todos los grupos de mosaicos de la misma imagen están restringidos para hacer referencia al mismo PPS, el ID de subimagen no necesita estar presente inmediatamente después de la cabecera de su unidad NAL. Si se permite que

los grupos de mosaicos de la misma imagen hagan referencia a diferentes PPS, el ID de subimagen puede estar presente en los primeros bits del PPS (por ejemplo, inmediatamente después del encabezado de la unidad NAL). En este caso, se puede permitir que cualquier grupo de mosaicos de una imagen comparta el mismo PPS. Alternativamente, cuando se permite que los grupos de mosaicos de la misma imagen se refieran a diferentes PPS, y también se permite que diferentes grupos de mosaicos de la misma imagen compartan el mismo PPS, no puede estar presente ningún ID de subimagen en la sintaxis del PPS. Alternativamente, cuando se permite que los grupos de mosaicos de la misma imagen se refieran a diferentes PPS, y también se permite que diferentes grupos de mosaicos de la misma imagen compartan el mismo PPS, puede estar presente una lista de ID de subimágenes en la sintaxis del PPS. La lista indica las subimágenes a las que se aplica el PPS. Para otras unidades NAL no VCL, si la unidad no VCL se aplica al nivel de imagen o superior (por ejemplo, el delimitador de la unidad de acceso, el final de la secuencia, el final del flujo de bits, etc.), entonces el ID de subimagen puede no estar presente inmediatamente después del encabezado de la unidad NAL. De lo contrario, el ID de la subimagen puede estar presente inmediatamente después del encabezado de la unidad NAL.

Con la señalización SPS anterior, la división de mosaicos dentro de subimágenes individuales puede señalizarse en el PPS. Se puede permitir que los grupos de mosaicos dentro de la misma imagen hagan referencia a diferentes PPS. En este caso, la agrupación de mosaicos solo puede estar dentro de cada subimagen. El concepto de agrupación de mosaicos consiste en dividir una subimagen en mosaicos.

Alternativamente, se define un conjunto de parámetros para describir la división de mosaicos dentro de subimágenes individuales. Un conjunto de parámetros de este tipo puede denominarse conjunto de parámetros de subimagen (SPPS). El SPPS se refiere a SPS. Un elemento sintáctico que hace referencia al ID de SPS está presente en SPPS. El SPPS puede contener una ID de subimagen. Para fines de extracción de subimágenes, el elemento sintáctico que hace referencia al ID de subimagen es el primer elemento sintáctico en SPPS. El SPPS contiene una estructura de mosaicos (por ejemplo, un número de columnas, un número de filas, un espaciado uniforme de los mosaicos, etc.) El SPPS puede contener un indicador para indicar si un filtro de bucle está activado o no en los límites de las subimágenes asociadas. Alternativamente, las propiedades de subimagen para cada subimagen pueden señalizarse en el SPPS en lugar de en el SPS. La división de mosaicos dentro de subimágenes individuales aún puede señalizarse en el PPS. Los grupos de mosaicos dentro de la misma imagen pueden hacer referencia a diferentes PPS. Una vez que se activa un SPPS, el SPPS dura una secuencia de AU consecutivas en orden de decodificación. Sin embargo, el SPPS puede desactivarse/activarse en una AU que no sea el inicio de un CVS. En cualquier momento durante el proceso de decodificación de un flujo de bits de una sola capa con múltiples subimágenes en algunas AU, pueden estar activos múltiples SPPS. Un SPPS puede ser compartido por diferentes subimágenes de una AU. Alternativamente, SPPS y PPS se pueden combinar en un conjunto de parámetros. En tal caso, es posible que no sea necesario que todos los grupos de mosaicos de la misma imagen hagan referencia al mismo PPS. Se puede aplicar una restricción de modo que todos los grupos de mosaicos de la misma subimagen puedan hacer referencia al mismo conjunto de parámetros resultante de la fusión entre SPPS y PPS.

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

Al decodificar una subimagen, la ubicación de cada bloque del árbol de codificación (por ejemplo, XctB e YCTb) puede ajustarse a una ubicación de muestra de luminancia real en la imagen en lugar de a una ubicación de muestra de luminancia en la subimagen. De esta manera, se puede evitar la extracción de una subimagen coubicada de cada imagen de referencia, ya que el bloque del árbol de codificación se decodifica con referencia a la imagen en lugar de a la subimagen. Para ajustar la ubicación de un bloque de árbol de codificación, las variables SubpictureXOffset y SubpictureYOffset pueden derivarse en función de la posición de la subimagen (subpic_x_offset y subpic_y_offset). Los valores de las variables pueden añadirse a los valores de las coordenadas x e y de ubicación de la muestra de luminancia, respectivamente, de cada bloque del árbol de codificación en la subimagen.

Un proceso de extracción de subimágenes se puede definir de la siguiente manera. La entrada al proceso es la subimagen objetivo que se va a extraer. Esto puede ser en forma de ID de subimagen o ubicación de subimagen. Cuando la entrada es la ubicación de una subimagen, el ID de subimagen asociado se puede resolver analizando la información de subimagen en el SPS. Para las unidades NAL que no son VCL, se aplica lo siguiente. Los elementos sintácticos del SPS relacionados con el tamaño y el nivel de la imagen pueden actualizarse con la información de tamaño y nivel de la subimagen. Las siguientes unidades NAL que no son VCL se conservan sin cambios: PPS, delimitador de unidades de acceso (AUD), final de secuencia (EOS), final del flujo de bits (EOB) y cualquier otra unidad NAL que no sea de VCL que sea aplicable al nivel de imagen o superior. Se pueden eliminar las unidades NAL no VCL restantes con un ID de subimagen que no sea igual al ID de subimagen objetivo. También se pueden eliminar las unidades NAL de VCL con una ID de subimagen que no sea igual a la ID de subimagen objetivo.

Se puede usar un mensaje SEI de anidamiento de subimágenes a nivel de secuencia para anidar mensajes SEI a nivel de AU o a nivel de subimagen para un conjunto de subimágenes. Esto puede incluir un período de almacenamiento en búfer, la temporización de las imágenes y los mensajes SEI no HRD. La sintaxis y la semántica de este mensaje SEI de anidamiento de subimágenes pueden ser las siguientes. Para operaciones de sistemas, como en entornos de formato multimedia omnidireccional (OMAF), el reproductor OMAF puede solicitar y decodificar un conjunto de secuencias de subimágenes que cubran una ventana gráfica. Por lo tanto, el mensaje SEI a nivel de secuencia se usa para transportar información de un conjunto de secuencias de subimágenes que cubren colectivamente una región de imágenes rectangulares. Los sistemas pueden usar la información, y la información es indicativa de la capacidad de decodificación requerida, así como de la tasa de bits del conjunto de secuencias de subimágenes. La información indica el nivel del flujo de bits que incluye solo el conjunto de secuencias de subimágenes. Esta información también indica la velocidad de bits del flujo de bits que contiene solo el conjunto de secuencias de subimágenes. Opcionalmente, se puede especificar un proceso de extracción de subflujos de bits para un conjunto de secuencias de subimágenes. La ventaja de hacer esto es que el flujo de bits que incluye solo un conjunto de secuencias de subimágenes también puede ser conforme. Una desventaja es que, al considerar las diferentes posibilidades de tamaño de ventana gráfica, puede haber muchos conjuntos de este tipo además del ya elevado número posible de secuencias de subimágenes individuales.

En una modalidad ejemplar, uno o más de los ejemplos descritos pueden implementarse de la siguiente manera. Una subimagen puede definirse como una región rectangular de uno o más grupos de mosaicos dentro de una imagen. Un proceso de división binaria permitido se puede definir de la siguiente manera. Las entradas de este proceso son: un BTSplit en modo de división binario, un ancho de bloque de codificación CBWidth, una altura de bloque de codificación CBHeight, una ubicación (x0, y0) de la muestra de luminancia superior izquierda del bloque de codificación considerado en relación con la muestra de luminancia superior izquierda de la imagen, una profundidad de árbol multitipo mttDepth, una profundidad de árbol multitipo máxima con desplazamiento MaxMttDepth, un tamaño máximo de árbol binario MaxBTSize y una partición índice PartidX. El resultado de este proceso es la variable allowBTSplit.

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

Especificación de ParallelTTSplit y CBSize basada en BTSplit

Las variables parallelTTSplit y cbSize se derivan como se especificó anteriormente. La variable allowBtSplit se deriva de la siguiente manera: Si se cumple una o más de las siguientes condiciones, allowBtsplit se establece en FALSE: cbSize es menor o igual que minBtSizeY, cbWidth es mayor que maxBTSize, cbHeight es mayor que maxBTSize y mttDepth es mayor o igual que maxMttDepth. De lo contrario, si se cumplen todas las condiciones siguientes, allowBtsplit se establece en FALSE: btSplit es igual a SPLIT BT VER y y0 + cbHeight es mayor que SubpicBottomBorderInPic. De lo contrario, si se cumplen todas las condiciones siguientes, allowBtsplit se establece en FALSE, btSplit es igual a SPLIT_BT_HOR, x0 + cbWidth es mayor que subpicRightBorderInPic e y0 + cbHeight es menor o igual que subpicBottomBorderInPic. De lo contrario, si se cumplen todas las condiciones siguientes, allowBtsplit se establece en FALSE: mttDepth es mayor que cero, PartItx es igual a uno y mttSplitMode [x0] [y0] [mttDepth - 1] es igual a ParallelTTSplit. De lo contrario, si se cumplen todas las condiciones siguientes, allowBtsplit se establece igual a FALSE: btSplit es igual a SPLIT_BT_VER, cbWidth es menor o igual a maxTbSizeY y cbHeight es mayor que maxTbSizeY. De lo contrario, si se cumplen todas las condiciones siguientes, allowBtsplit se establece en FALSE: btSplit es igual a SPLIT_BT_HOR, cbWidth es mayor que maxTbSizeY y cbHeight es menor o igual que maxTbSizeY. De cualquier otra manera, allowBtSplit se establece igual a TRUE.

Un proceso de división ternaria permitido se puede definir de la siguiente manera. Las entradas de este proceso son: un TTSplit en modo de división ternario, un ancho de bloque de codificación CBWidth, una altura de bloque de codificación CBHeight, una ubicación (x0, y0) de la muestra de luminancia superior izquierda del bloque de codificación considerado en relación con la muestra de luminancia superior izquierda de la imagen, una profundidad de árbol multitipo mttDepth, una profundidad máxima de árbol multitipo con offset MaxMttDepth y un tamaño máximo de árbol binario MaxTTSize. El resultado de este proceso es la variable allowTTSplit.

	ttSplit = = SPLIT_TT_VER	ttSplit = = SPLIT_TT_HOR
cbSize	cbWidth	cbHeight

Especificación de CBSize basada en TTSplit.

La variable cbSize se deriva como se especificó anteriormente. La variable allowTtSplit se deriva de la siguiente manera: Si se cumplen una o más de las siguientes condiciones, allowTTSplit se establece en FALSE: cbSize es menor o igual que 2 * minttSizeY, cbWidth es mayor que Min (maxTbSizeY, maxTTSize), cbHeight es mayor que Min (maxTbSizeY, maxTTSize), mt- tDepth es mayor o igual que maxMttDepth, x0 + cbWidth es mayor que

subPicRightBorderInPic e y0 + cbHeight es mayor que SubPicBottomBorderInPic. De cualquier otra manera, allowTtSplit se establece igual a TRUE.

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

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

El pic_width_in_luma_samples especifica el ancho de cada imagen decodificada en unidades de muestras de luminancia. pic_width_in_luma_samples no debe ser igual a cero y debe ser un múltiplo entero de minCBSizeY. El pic_height_in_luma_samples especifica la altura de cada imagen decodificada en unidades de muestras de luminancia. pic_height_in_luma_samples no debe ser igual a cero y debe ser un múltiplo entero de minCBSizeY. El num_subpicture_minus1 más 1 especifica el número de subimágenes divididas en imágenes codificadas que pertenecen a la secuencia de vídeo codificada. El subpic_id_len_minus1 más 1 especifica el número de bits utilizados para representar el elemento sintáctico subpic_id [i] en el SPS, spps_subpic_id en el SPPS que hace referencia al SPS y tile_group_subpic_id en los encabezados de los grupos de mosaicos que hacen referencia al SPS. El valor de subpic_id_len_minus1 estará comprendido entre Ceil (Log2 (num_subpic_minus1 + 2) y ocho, ambos inclusive. El subpic_id [i] especifica el ID de subimagen de la i-ésima subimagen de imágenes que hacen referencia al SPS. La longitud de subpic_id [i] es subpic_id_len_minus1 + 1 bits. El valor de subpic_id [i] debe ser mayor que cero. El subpic_level_idc [i] indica un nivel en el que el CVS resultante de la extracción de las subimágenes i-ésimas se ajusta a los requisitos de recursos especificados. Los flujos de bits no deben contener valores de subpic_level_idc [i] distintos de los especificados. Se reservan otros valores de subpic_level_idc [i]. Cuando no está presente, se deduce que el valor de subpic_level_idc [i] es igual al valor de general_level_idc.

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

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

que la i-ésima subimagen puede o no ser una subimagen con movimiento limitado en el tiempo. Cuando no está presente, se deduce que el valor de subpic_motion_constrained_flag es igual a cero.

Las variables SubpicWidthInCtbSy, SubpicHeightInCTBSY, SubpicSizeInCTBSY, SubpicWidthInMincBSY, SubpicHeightInMincBSY, SubpicSizeInSamplesY, subpicWidthInSamplesC y subpicHeightIn- SamplesC se derivan de la siguiente manera:

SubpicWidthInLumaSamples[i] = subpic_width_in_luma_samples[i]

SubpicHeightInLumaSamples[i] = subpic_height_in_luma_samples[i]

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

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

SubpicWidthInCtbsY[i] = Ceil(SubpicWidthInLumaSamples[i] 4 CtbSizeY)

SubpicHeightInCtbsY[i] = Ceil(SubpicHeightInLumaSamples[i] 4 CtbSizeY)

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

SubpicWidthInMinCbsY[i] = SubpicWidthInLumaSamples[i] / MinCbSizeY

SubpicHeightInMinCbsY[i] = SubpicHeightInLumaSamples[i] / MinCbSizeY

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

SubpicSizeInSamplesY[i] = SubpicWidthInLumaSamples[i] *

SubpicHeightInLumaSamples[i]

SubpicWidthInSamplesC[i] = SubpicWidthInLumaSamples[i] / SubWidthC

SubpicHeightInSamplesC[i] = SubpicHeightInLumaSamples[i] / SubHeightC

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

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

El spps_subpic_id identifica la subimagen a la que pertenece el SPPS. La longitud de spps_subpic_id es subpic_id_len_minus1 + 1 bits. El spps_subpic_parameter_set_id identifica el SPPS para que otros elementos

sintácticos puedan hacer referencia a él. El valor de spps_subpic_parameter_set_id estará comprendido entre cero y sesenta y tres, ambos inclusive. El spps_seq_parameter_set_id especifica el valor de sps_seq_parameter_set_id para el SPS activo. El valor de spps_seq_parameter_set_id debe estar comprendido entre cero y quince, ambos inclusive. El single_tile_in_subpic_flag igual a uno especifica que solo hay un mosaico en cada subimagen que hace referencia al SPPS. El single_tile_in_subpic_flag igual a cero especifica que hay más de un mosaico en cada subimágen que hace referencia al SPPS. El valor num_tile_columns_minus 1 más 1 especifica el número de columnas de mosaicos que dividen la subimagen. El num_tile_columns_minus1 debe estar en el rango de cero a picWidthinctbsy [spps_subpic_id] - 1, ambos inclusive. Cuando no está presente, se deduce que el valor de num_tile_columns_minus1 es igual a cero. El num_tile_rows_minus1 más 1 especifica el número de filas de mosaicos que dividen la subimagen. El num_tile_rows_minus1 debe estar comprendido entre cero y picHeightincTB- sY [spps_subpic_id] - 1, ambos inclusive. Cuando no está presente, se deduce que el valor de num_tile_rows_minus1 es igual a cero. La variable numTilesInPic se establece igual a (num_tile_columns_minus1 + 1) * (num_tile_rows_minus1 + 1).

Cuando single_tile_in_subpic_flag es igual a cero, numTilesInPic será mayor que cero. El uniform_tile_spacing_flag igual a uno especifica que los límites de las columnas de las mosaicos y, del mismo modo, los límites de las filas de las mosaicos se distribuyen uniformemente en la subimagen. El uniform_tile_spacing_flag igual a cero especifica que los límites de las columnas de las mosaicos y, del mismo modo, los límites de las filas de las mosaicos no se distribuyen uniformemente en la subimágen, sino que se señalizan explícitamente mediante los elementos sintácticos tile_column_width_minus1 [i] y tile_row_height_minus1 [i]. Cuando no está presente, se deduce que el valor de uniform_tile_spacing_flag es igual a uno. El tile_column_width_minus1 [i] más 1 especifica el ancho de la i-ésima columna del mosaico en unidades de CTB. El tile_row_height_minus1 [i] más 1 especifica la altura de la i-ésima fila de mosaicos en unidades de CTB.

Las siguientes variables se derivan invocando el proceso de conversión del escaneo de mosaicos y rásteres de CTB: la lista colWidth [i] para i va desde cero hasta num_tile_columns_minus1, inclusive, que especifica el ancho de la i-ésima columna de mosaicos en unidades de CTB; La lista RowHeight [j] para j, que va desde cero hasta num_tile_rows_minus1, inclusive, especificando la altura de la hilera de mosaicos j en unidades de CTB; La lista colBd [i] para i, que va desde cero hasta num_tile_columns_minus1 + 1, ambos inclusive, especificando la ubicación del límite de la i-ésima columna del mosaico en unidades de CTB; La lista RowBD [j] para j, comprendida entre cero y num_tile_rows_minus1 + 1, ambos inclusive, especificando la ubicación del límite de la hilera de mosaicos enésima en unidades de CTB; La lista ctBaddrrsTots [ctBaddrrs] para CTBaddrrs que van desde cero hasta PicSizeInCTBSy - 1, ambos inclusive, que especifica la conversión de una dirección CTB en el escaneo rasterizado CTB de una imagen a una dirección CTB en el escaneo de mosaicos; La lista ctBaddrtStors [ct- BadDrts] para CTBaddrts que van desde cero hasta PicSizeInCTBSy - 1, inclusive, que especifica la conversión de una dirección CTB en el escaneo de mosaicos a una dirección CTB en el escaneo rasterizado CTB de una imagen; La lista TileID [ctBaddrts] para ctBaddrts, que va desde cero hasta PicSizeInCTBsy - 1, inclusive, que especifica la conversión de una dirección CTB en el escaneo de mosaicos a un ID de mosaico; La lista NumCtuInTile [TileIdX] para TileIdX, que va desde cero hasta PicSizeInCTBSy - 1, inclusive, que especifica la conversión de un índice de mosaico al número de CTU en el mosaico; La lista FirstCTBaddRts [TileIdx] para TileIdX, que va desde cero hasta numTilesInPic - 1, inclusive, que especifica la conversión de un ID de mosaico a la dirección CTB en el escaneo de mosaicos del primer CTB del mosaico; La lista ColumnWidthinLumaSamples [i] para i oscila entre cero y num_tile_columns_minus1, ambos inclusive, especificando el ancho de la i-ésima columna del mosaico en unidades de muestras de luminancia; Y la lista RowHeightInLumaSamples [j] para j, que va desde cero hasta num_tile_rows_minus1, ambos inclusive, especificando la altura de la hilera de mosaicos enésima en unidades de muestras de luminancia. Los valores de ColumnWidthinLumaSamples [i] para i, que van desde cero hasta num_tile_columns_minus1, inclusive, y RowHeightInLuma- Samples [j] para j, que van desde cero hasta num_tile_rows_minus1, inclusive, deberán ser todos mayores que cero.

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

La sintaxis y la semántica generales de los encabezados de los grupos de mosaicos son las siguientes.

tile_group_header() {	Descriptor

tile_group_subpic_id	u(v)
tile_group_subpic_parameter_set_id	u(v)
...	
}	

El valor del elemento sintáctico de cabecera de grupo de mosaicos tile_group_pic_parameter_set_id y tile_group_pic_order_cnt_lsb será el mismo en todas las cabeceras de grupo de mosaicos de una imagen codificada. El valor del elemento sintáctico de cabecera de grupo de mosaicos tile_group_subpic_id será el mismo en todas las cabeceras de grupo de mosaicos de una subimagen codificada. El tile_group_subpic_id identifica la subimagen a la que pertenece el grupo de mosaicos. La longitud de tile_group_subpic_id es subpic_id_len_minus1 + 1 bits. El tile_group_subpic_parameter_set_id especifica el valor de spps_subpic_parameter_set_id para el SPPS en uso. El valor de tile_group_spps_parameter_set_id estará comprendido entre cero y sesenta y tres, ambos inclusive.

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

PicWidthInLumaSamples = SubpicWidthInLumaSamples[tile_group_subpic_id]

PicHeightInLumaSamples = PicHeightInLumaSamples[tile_group_subpic_id]

SubPicRightBorderInPic = SubPicRightBorderInPic[tile_group_subpic_id]

SubPicBottomBorderInPic = SubPicBottomBorderInPic[tile_group_subpic_id]

PicWidthInCtbsY = SubPicWidthInCtbsY[tile_group_subpic_id] PicHeightInCtbsY = SubPicHeightInCtbsY[tile_group_subpic id] PicSizeInCtbsY =

SubPicSizeInCtbsY[tile_group_subpic_id] PicWidthInMinCbsY =

SubPicWidthInMinCbsY[tile_group_subpic_id] PicHeightInMinCbsY =

SubPicHeightInMinCbsY[tile_group_subpic_id] PicSizeInMinCbsY =

SubPicSizeInMinCbsY[tile_group_subpic_id] PicSizeInSamplesY =

SubPicSizeInSamplesY[tile_groupsubpic_id] PicWidthInSamplesC =

SubPicWidthInSamplesC[tile_group_subpic_id] PicHeightInSamplesC =

SubPicHeightInSamplesC[tile_group_subpic_id]

La sintaxis de la unidad de árbol de codificación es la siguiente.

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

}	
}	

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

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

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

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

La sintaxis y la semántica del árbol multitipo son las siguientes.

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

IsCuQpDeltaCoded = 0	
CuQpDeltaVal = 0	
CuQgTopLeftX = x0	
CuQgTopLeftY = y0	
}	
si (mtt_split_cu_flag) {	
si ((allowSplitBtHor || allowSplitTtHor) && allowSplitBtVer | allowSplitTtVer))	
mtt_split_cu_vertical_flag	ae(v)
si((allowSplitBtVer && allowSplitTtVer && mtt_split_cu_vertical_flag) || (allowSplitBtHor && allowSplitTtHor && !mtt_split_cu_vertical_flag))	
mtt_split_cu_binary_flag	ae(v)
si (MttSplitMode[x0][y0][mttDepth] = = SPLIT_BT_VER){	
depthOffset += (x0 + cbWidth > SubPicRightBorderInPic) ? 1: 0	
x1= x0 + (cbWidth / 2)	
multi_type_tree(x0, y0, cbWidth / 2, cbHeight, cqtDepth, mttDepth + 1, depthOffset, 0, treeType)	
si (x1 < SubPicRightBorderInPic)	
multi_type_tree(x1, y0, cbWidth / 2, cbHeightY, cqtDepth, mttDepth + 1, depthOffset, 1, treeType)	
} o ifSplitMode[x0][y0][mttDepth] = = SPLIT_BT_HOR) {	
depthOffset += (y0 + cbHeight > SubPicBottomBorderInPic) ? 1: 0	
y1 = y0 + (cbHeight / 2)	
multi_type_treex0, y0, cbWidth, cbHeight / 2, cqtDepth, mttDepth + 1, depthOffset, 0, treeType)	
si (y1 < SubPicBottomBorderInPic)	
multi_type_tree(x0, y1, cbWidth, cbHeight / 2, cqtDepth, mttDepth + 1, depthOffset, 1, treeType)	
+ o si (MttSplitMode[x0][y0][mttDepth] = = SPLIT_TT_VER) {	
x1 = x0 + (cbWidth / 4)	
x2 = x0 + (3 * cbWidth / 4)	
muti_type_tree(x0, y0, cbWidth / 4, cbHeight, cqtDepth, mttDepth + 1, depthoffset, 0, treeType)	
multi_type_tree(x1, y0, cbWidth / 2, cbHeight, cqtDepth, mttDepth + 1, depthoffset, 1, treeType)	
multi_type_tree(x2, y0, cbWidth / 4, cbHeight, cqtDepth, mttDepth + 1, depthoffset, 2, treeType)	
} o {/* SPLIT_TT_HOR */	
y1 = y0 + (cbHeight / 4)	
y2 = y0 + (3 * cbHeight / 4)	
multi_type_tree(x0, y0, cbWidth, cbHeight / 4, cqtDepth, mttDepth + 1, depthoffset, 0, treeType)	
multi_type_tree(x0, y1, cbWidth, cbHeight / 2, cqtDepth, mttDepth + 1, depthoffset, 1, treeType) cqtDepth, mttDepth + 1,	
multi_type_tree(x0, y2, cbWidth, cbHeight / 4, cqtDepth, mttDepth + 1, depthoffset, 2, treeType) cqtDepth, mttDepth + 1,	
}	
} o	
coding_unit(x0, y0, cbWidth, cbHeight, treeType)	
}	

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

mtt_split_cu_flag es igual a 1: x0 + cbWidth es mayor que subpicRightBorderInPic e y0 + cbHeight es mayor que subpicBottomBorderInPic. De lo contrario, se deduce que el valor de mtt_split_cu_flag es igual a cero.

El proceso de derivación para la predicción del vector de movimiento de luminancia temporal es el siguiente. Los resultados de este proceso son: la predicción vectorial de movimiento mvlxCol con una precisión de muestra fraccionada de 1/16 y el indicador de disponibilidad AvailableFlagLxCol. La variable currCB especifica el bloque de codificación de luminancia actual en la ubicación de luminancia (xcB, ycB). Las variables mvLXCol y availableFlagLXCol se derivan de la siguiente manera: Si tile_group_temporal_mvp_enabled_flag es igual a cero, o si la imagen de referencia es la imagen actual, ambos componentes de mvlxCol se establecen en cero y AvailableFlagLxCol se establece en cero. De lo contrario (tile_group_temporal_mvp_enabled_flag es igual a uno y la imagen de referencia no es la imagen actual), se aplican los siguientes pasos ordenados. El vector de movimiento colocado de la parte inferior derecha se obtiene de la siguiente manera:

xColBr = xCb + cbWidth (8-355)

yColBr = yCb + cbHeight (8-356)

Si yCb >> CtbLog2SizeY is equal to yColBr >> CtbLog2SizeY, yColBr es menor que SubPicBottomBorderInPic and xColBr is less than SubPicRightBorderInPic, se aplica lo siguiente. La variable colCb especifica el bloque de codificación de luminancia que cubre la ubicación modificada indicada por ((xColBr >> 3) << 3, (yColBr >> 3) << 3) dentro de la imagen colocalizada especificada por ColPic. La ubicación de luminancia (xColCB, yColCB) se establece igual a la muestra superior izquierda del bloque de codificación de luminancia colocado especificado por colCB en relación con la muestra de luminancia superior izquierda de la imagen colocada especificada por colPic. El proceso de derivación de los vectores de movimiento colocados se invoca con CurrCB, colCB, (xColCB, yColCB), ReflDxlx y sbFlag establecidos en cero como entradas, y la salida se asigna a mvlxCol y AvailableFlagLxCol. De lo contrario, ambos componentes de mvlxCol se establecen en cero y AvailableFlagLxCol se establece en cero.

El proceso de derivación para los candidatos que fusionan triángulos temporales es el siguiente. Las variables mvLXColC0, mvLXColC1, availableFlagLXColC0 y availableFlagLXColC1 se derivan de la siguiente manera: Si tile_group_temporal_mvp_enabled_flag es igual a cero, los dos componentes de mvlxColc0 y mvlxColc1 se establecen en cero y AvailableFlagLxColc0 y AvailableFlagLxColc1 se establecen en cero. De cualquier otra manera, (tile_group_temporal_mvp_enabled_flag is equal to 1) se aplican las siguientes etapas ordenadas: El vector de movimiento colocado de la parte inferior derecha mVLxColc0 se obtiene de la siguiente manera:

xColBr = xCb + cbWidth (8-392)

yColBr = yCb + cbHeight (8-393)

Si yCb >> CtbLog2SizeY is equal to yColBr >> CtbLog2SizeY, yColBr es menor que SubPicBottomBorderInPic and xColBr is less than SubPicRightBorderInPic, se aplica lo siguiente. La variable colCb especifica el bloque de codificación de luminancia que cubre la ubicación modificada indicada por ((xColBr >> 3) << 3, (yColBr >> 3) << 3) dentro de la imagen colocalizada especificada por ColPic. La ubicación de luminancia (xColCB, yColCB) se establece igual a la muestra superior izquierda del bloque de codificación de luminancia colocado especificado por colCB en relación con la muestra de luminancia superior izquierda de la imagen colocada especificada por colPic. El proceso de derivación de los vectores de movimiento colocados se invoca con CurrCB, colCB, (xColCB, yColCB), refidXLxc0 y sbFlag iguales a cero como entradas, y la salida se asigna a mvlxColc0 y AvailableFlagLxColc0. De lo contrario, ambos componentes de mvlxColc0 se establecen en cero y AvailableFlagLxColc0 se establece en cero.

El proceso de derivación para los candidatos a la fusión de vectores de movimiento de puntos de control afines construidos es el siguiente. El cuarto vector de movimiento del punto de control cpmVLxCorner [3] (colocado en la parte inferior derecha), el índice de referencia RefidXLXCorner [3], el indicador de utilización de la lista de predicción PredFlagLxCorner [3] y el indicador de disponibilidad AvailableFlagCorner [3], donde X es 0 y 1, se derivan de la siguiente manera. Los índices de referencia del candidato a la fusión temporal, refidXLXCorner [3], donde X es cero o uno, se establecen en cero. Las variables mvlxCol y AvailableFlagLxCol, donde X es cero o uno, se derivan de la siguiente manera. Si tile_group_temporal_mvp_enabled_flag es igual a cero, ambos componentes de mvlxCol se establecen en cero y AvailableFlagLxCol se establece en cero. De cualquier otra manera (tile_group_temporal_mvp_enabled_flag es igual a cero), se aplica lo siguiente:

xColBr = xCb + cbWidth (8-566)

yColBr = yCb + cbHeight (8-567)

Si yCb >> CtbLog2SizeY is equal to yColBr >> CtbLog2SizeY, yColBr es menor que SubPicBottomBorderInPic and xColBr is less than SubPicRightBorderInPic, se aplica lo siguiente. La variable colCb especifica el bloque de codificación de luminancia que cubre la ubicación modificada indicada por ((xColBr >> 3) << 3, (yColBr >> 3) << 3) dentro de la imagen colocalizada especificada por ColPic. La ubicación de luminancia (xColCB, yColCB) se establece

igual a la muestra superior izquierda del bloque de codificación de luminancia colocado especificado por colCB en relación con la muestra de luminancia superior izquierda de la imagen colocada especificada por colPic. El proceso de derivación de los vectores de movimiento colocados se invoca con CurrCB, colCB, (xColCB, yColCB), RefIDxlx y sbFlag establecidos en cero como entradas, y la salida se asigna a mvlxCol y AvailableFlagLxCol. De lo contrario, ambos componentes de mvlxCol se establecen en 0 y AvailableFlagLxCol se establece en cero. Sustituya todas las apariciones de pic_width_in_luma_samples por PicWidthInLumaSamples. Sustituya todas las apariciones de pic_height_in_luma_samples por PicHeightInLumaSamples.

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

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

El subpic_id_len_minus1 más uno especifica el número de bits utilizados para representar el elemento sintáctico subpic_id [i] en el SPS, spps_subpic_id en el SPPS que hace referencia al SPS y tile_group_subpic_id en los encabezados de los grupos de mosaicos que hacen referencia al SPS. El valor de subpic_id_len_minus1 estará comprendido entre Ceil (Log2 (num_subpic_minus1 + 3) y ocho, ambos inclusive. Es un requisito para la conformidad del flujo de bits que no haya superposición entre la subimagen [i] para i de 0 a num_subpic_minus1, inclusive. Cada subimagen puede ser una subimagen con movimiento temporal restringido.

La semántica general de los encabezados de los grupos de mosaicos es la siguiente. El tile_group_subpic_id identifica la subimagen a la que pertenece el grupo de mosaicos. La longitud de tile_group_subpic_id es subpic_id_len_minus1 + 1 bits. El tile_group_subpic_id igual a uno indica que el grupo de mosaicos no pertenece a ninguna subimagen.

En una tercera modalidad ejemplar, la sintaxis y la semántica de la cabecera de la unidad NAL son las siguientes.

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

El nuh_subpicture_id_len especifica el número de bits utilizados para representar el elemento sintáctico que especifica el ID de la subimagen. Cuando el valor de nuh subpicture_id_len es mayor que cero, los primeros bits nuh_subpicture_id_len después de nuh_reserved_zero_4bits especifican el ID de la subimagen a la que pertenece la carga útil de la unidad NAL. Cuando nuh_subpicture_id_len es mayor que cero, el valor de nuh subpicture_id_len será igual al valor de subpic_id_len_minus1 en el SPS activo. El valor de nuh_subpicture_id_len para las unidades NAL que no son de VCL se restringe de la siguiente manera. Si nal_unit_type es igual a SPS_NUT o PPS_NUT, nuh_subpicture_id_len será igual a cero. El nuh_reserved_zero_3bits será igual a “000”. Los decodificadores ignorarán (p. ej., eliminarán del flujo de bits y descartarán) las unidades NAL con valores de nuh_reserved_zero_3bits distintos de “000”.

En una cuarta modalidad ejemplar, la sintaxis de anidamiento de subimágenes es la siguiente.

sub-picture_nesting(payloadSize) {	Descriptor
all_sub_pictures_flag	u(1)
if(!all_sub_pictures_flag) {	
nesting_num_sub_pictures_minus 1	ue(v)
para (i = 0; i <= nesting_num_sub_pictures_minus1; i++)	
nesting_sub_picture_id[i]	u(v)
}	
mientras (!byte_aligned())	
sub_picture_nesting_zero_bit /* equal to 0 */	u(1)
do	
sei_message()	
mientras (more_rbsp_data())	
}	

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

La figura 9 es un diagrama esquemático de un dispositivo 900 de codificación de vídeo ejemplar. El dispositivo 900 de codificación de vídeo es apto para implementar los ejemplos/realizaciones divulgados tal y como se describen en la presente memoria. El dispositivo 900 de codificación de vídeo comprende unos puertos de recepción 920, unos puertos 950 de envío y/o unas unidades transceptoras (Tx/Rx) 910, que incluyen unos transmisores y/o receptores para comunicar datos de envío y/o de recepción por una red. El dispositivo 900 de codificación de vídeo también incluye un procesador 930, que incluye una unidad lógica y/o una 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 electrónicos optoeléctricos (OE), componentes electroópticos (EO) y/o componentes de comunicación inalámbrica acoplados a los puertos 950 de envío y/o puertos 920 de recepción para la comunicación de datos a través de redes de comunicaciones eléctricas, ópticas o inalámbricas. El dispositivo 900 de codificación de vídeo también puede incluir unos dispositivos 960 de entrada y/o salida (E/S) para comunicar datos a y de un usuario. Los dispositivos 960 de E/S pueden incluir dispositivos de salida tales como una pantalla para visualizar datos de vídeo, unos altavoces para dar salida a datos de audio, etc. Los dispositivos 960 de E/S también pueden incluir unos dispositivos de entrada, tales como un teclado, un ratón, una bola de seguimiento, etc., y/o unas correspondientes interfaces para interactuar con tales 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 (p. ej., como un procesador multinúcleo), matrices de compuertas 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 de recepción, el Tx/Rx 910, los puertos 950 de envío y la memoria 932. El procesador 930 comprende un módulo 914 de codificación. El módulo de codificación 914 implementa las realizaciones descritas anteriormente, tales como los métodos 100, 1000, 1100 y/o el mecanismo 700, que pueden emplear un flujo de bits 500, una imagen 600 y/o una imagen 800. El módulo de codificación 914 también puede implementar cualquier otro método/mecanismo descrito en el presente documento. Además, el módulo de codificación 914 puede implementar un sistema de códec 200, un codificador 300 y/o un decodificador 400. Por ejemplo, el módulo de codificación 914 puede emplearse para señalizar y/u obtener ubicaciones y tamaños de subimágenes en un SPS. En otro ejemplo, el módulo de codificación 914 puede restringir las anchuras y alturas de subimagen para que sean múltiplos del tamaño de la CTU, a menos que dichas subimágenes estén posicionadas en el borde derecho de la imagen o en el borde inferior de la imagen, respectivamente. En otro ejemplo, el módulo de codificación 914 puede restringir las subimágenes para cubrir una imagen sin espacios ni superposición. En otro ejemplo, el módulo de codificación 914 puede emplearse para señalizar y/u obtener datos que indiquen que algunas subimágenes son subimágenes con movimiento temporal restringido y otras subimágenes no lo son. En otro ejemplo, el módulo de codificación 914 puede señalizar un conjunto completo de ID de subimagen en el SPS e incluir un ID de subimagen en cada cabecera de segmento para indicar la subimagen que contiene los segmentos correspondientes. En otro ejemplo, el módulo de codificación 914 puede señalizar los niveles para cada subimagen. Como tal, el módulo de codificación 914 hace que el dispositivo de codificación de vídeo 900 proporcione una funcionalidad adicional, evite cierto procesamiento para reducir la sobrecarga de procesamiento y/o aumente la eficiencia de codificación al

particionar y codificar datos de vídeo. Así, el módulo de codificación 914 mejora la funcionalidad del dispositivo 900 de codificación de vídeo, al tiempo que aborda problemas que son específicos de las técnicas de codificación de vídeo. Además, el módulo de codificación 914 efectúa una transformación del dispositivo 900 de codificación de vídeo a un estado diferente. Alternativamente, el módulo de codificación 914 puede implementarse como instrucciones almacenadas en la memoria 932 y ejecutadas por el procesador 930 (p. ej., 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, una memoria de solo lectura (ROM), una memoria de acceso aleatorio (RAM), una memoria flash, una memoria ternaria de contenido direccionable (TCAM), una memoria estática de acceso aleatorio (SRAM), etc. La memoria 932 puede usarse como dispositivo de almacenamiento de datos de desbordamiento para almacenar programas cuando tales programas se seleccionan para su ejecución y para almacenar instrucciones y datos que se leen durante la ejecución de programa.

La figura 10 es un diagrama de flujo de un método 1000 de ejemplo para codificar indicadores de nivel de subimágenes en un flujo de bits, tal como el flujo de bits 500, para soportar la decodificación de subimágenes, tal como las subimágenes 522, 523, 622, 722 y/u 822. El método 1000 puede ser empleado por un codificador, tal como un sistema de códec 200, un codificador 300 y/o un dispositivo de codificación de vídeo 900 cuando se realiza el método 100.

El procedimiento 1000 puede comenzar cuando un codificador recibe una secuencia de vídeo que incluye una pluralidad de imágenes y determina codificar esa secuencia de vídeo en un flujo de bits, por ejemplo, en función de la entrada del usuario. La secuencia de vídeo se divide en imágenes/imágenes/fotogramas para su posterior división antes de la codificación. En la etapa 1001, una imagen se divide en una pluralidad de subimágenes. Una o más de la pluralidad de subimágenes se codifican entonces en el flujo de bits.

En la etapa 1003, el codificador determina los requisitos de recursos para decodificar cada una de las una o más subimágenes. Esto se puede lograr como parte del proceso de RDO. Por ejemplo, el codificador puede emplear un decodificador de referencia hipotético (HRD) como parte del proceso de codificación. El HRD puede emplear recursos para decodificar múltiples codificaciones potenciales con el fin de determinar una codificación óptima. El codificador puede determinar los recursos utilizados por el HRD al decodificar la codificación seleccionada como la codificación óptima.

En la etapa 1005, los indicadores de nivel de subimagen se codifican en el flujo de bits. Los indicadores de nivel de subimagen indican los requisitos de recursos para decodificar cada una de las una o más subimágenes según lo determinado en la etapa 1003. En algunos ejemplos, los indicadores de nivel de subimagen se codifican en uno o más mensajes SEI en el flujo de bits. En otros ejemplos, los indicadores de nivel de subimagen se codifican en un SPS en el flujo de bits. Los indicadores de nivel de subimagen pueden indicar el tamaño de la subimagen, el recuento de píxeles de la subimagen, la velocidad de bits de la subimagen o combinaciones de los mismos.

En la etapa 1007, se puede codificar un SPS en el flujo de bits. Cuando los indicadores de nivel de subimagen se codifican en el SPS, las etapas 1005 y 1007 se pueden combinar. El SPS también puede comprender ID de subimágenes para cada una de las subimágenes. El SPS también puede comprender una ubicación de subimágenes para cada una de las subimágenes. El SPS también puede comprender un tamaño de subimagen para cada una de las subimágenes.

En la etapa 1009, el flujo de bits se almacena para la comunicación hacia un decodificador. El flujo de bits puede transmitirse entonces hacia el decodificador según se desee. En algunos ejemplos, se puede extraer un subflujo de bits del flujo de bits codificado. En tal caso, el flujo de bits transmitido es un subflujo de bits. En otros ejemplos, el flujo de bits codificado puede transmitirse para la extracción del subflujo de bits en el decodificador. En otros ejemplos más, el flujo de bits codificado puede decodificarse y mostrarse sin la extracción del subflujo de bits. En cualquiera de estos ejemplos, los indicadores de nivel de subimagen pueden transmitirse en el flujo de bits. De esta manera, cada subimagen puede codificarse independientemente de otras subimágenes. Esto asegura que los requisitos de decodificación se establezcan por subimagen en lugar de establecer los requisitos de decodificación para toda la imagen basándose en la sección más compleja de la codificación. Como tal, los indicadores de nivel de subimágenes aseguran que el decodificador no esté dirigido a establecer requisitos de decodificación demasiado altos para subimágenes codificadas de acuerdo con mecanismos menos complejos. La información a nivel de subimagen señalizada admite una mayor funcionalidad y/o una mayor eficiencia de codificación, lo que reduce el uso de recursos de red, recursos de memoria y/o recursos de procesamiento en el codificador y el decodificador.

La figura 11 es un diagrama de flujo de un método 1100 de ejemplo para decodificar un flujo de bits, tal como el flujo de bits 500 y/o el subflujo de bits 501, de subimágenes, tales como las subimágenes 522, 523, 622, 722 y/u 822, basándose en indicadores de nivel de subimágenes. El método 1100 puede ser empleado por un decodificador, tal como un sistema de códec 200, un decodificador 400 y/o un dispositivo de codificación de vídeo 900 cuando se realiza el método 100. Por ejemplo, el método 1100 se puede aplicar para decodificar un flujo de bits creado como resultado del método 1000.

El método 1100 puede comenzar cuando un decodificador comienza a recibir un flujo de bits que contiene subimágenes. El flujo de bits puede incluir una secuencia de vídeo completa o el flujo de bits puede ser un subflujo de bits que contiene un conjunto reducido de subimágenes para una extracción separada. En la etapa 1101, se recibe un flujo de bits. El flujo de bits comprende una o más subimágenes divididas a partir de una imagen. El flujo de bits también comprende un indicador de nivel de subimagen que indica los requisitos de recursos para decodificar una subimagen actual.

En la etapa 1103, el flujo de bits se analiza para obtener el indicador de nivel de subimagen y la subimagen actual. En algunos ejemplos, el indicador de nivel de subimagen se incluye en un mensaje SEI en el flujo de bits. En otros ejemplos, el indicador de nivel de subimagen se incluye en un SPS en el flujo de bits. Por lo tanto, analizar el flujo de bits incluye analizar el SPS, los mensajes SEI y/u otros conjuntos de parámetros como el PPS, las cabeceras de segmento, etc. El indicador de nivel de subimagen puede indicar el tamaño de la subimagen, el recuento de píxeles de la subimagen, la velocidad de bits de la subimagen o combinaciones de los mismos. El SPS también puede comprender información adicional de subimagen, tal como los ID de subimagen para cada una de las subimágenes, una ubicación de subimagen para cada una de las subimágenes, un tamaño de subimagen para cada una de las subimágenes, etc. Por lo tanto, el análisis del SPS también puede obtener dicha información.

En la etapa 1105, se asignan recursos para decodificar la subimagen actual basándose en el indicador de nivel de subimagen. La subimagen actual se decodifica entonces en la etapa 1107 para crear una secuencia de vídeo empleando los recursos asignados. La secuencia de vídeo puede reenviarse entonces para su visualización en la etapa 1109. En consecuencia, un indicador de nivel de subimagen puede transmitirse en un flujo de bits y un decodificador puede asignar por separado recursos de hardware y/o software para cada subimagen que se va a decodificar. De esta manera, cada subimagen puede codificarse independientemente de otras subimágenes. Además, los recursos no están sobreasignados como ocurre cuando el nivel se asigna a nivel de imagen. Además, los recursos se pueden asignar de manera apropiada cuando un subconjunto de las subimágenes se extrae por separado. Por lo tanto, los indicadores de nivel de subimagen admiten una mayor funcionalidad y/o una mayor eficiencia de codificación, lo que reduce el uso de recursos de red, recursos de memoria y/o recursos de procesamiento en el codificador y el decodificador.

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

El sistema 1200 incluye un codificador de vídeo 1202. El codificador de vídeo 1202 comprende un módulo de partición 1201 para dividir una imagen en una pluralidad de subimágenes. El codificador de vídeo 1202 comprende además un módulo de determinación 1203 para determinar los requisitos de recursos para decodificar cada una de las una o más subimágenes. El codificador de vídeo 1202 comprende además un módulo de codificación 1205 para codificar en un flujo de bits indicadores de nivel de subimágenes que indican los requisitos de recursos para decodificar una o más subimágenes; y codificar una o más de la pluralidad de subimágenes en el flujo de bits. El codificador de vídeo 1202 comprende además un módulo de almacenamiento 1207 para almacenar el flujo de bits para la comunicación hacia un decodificador. El codificador de vídeo 1202 comprende además un módulo de transmisión 1209 para transmitir el flujo de bits que incluye la (s) subimagen y los indicadores de nivel hacia el decodificador. El codificador de vídeo 1202 puede configurarse además para realizar cualquiera de las etapas del procedimiento 1000.

El sistema 1200 también incluye un decodificador de vídeo 1210. El decodificador de vídeo 1210 comprende un módulo de recepción 1211 para recibir un flujo de bits que comprende una o más subimágenes divididas a partir de una imagen y un indicador del nivel de subimagen que indica los requisitos de recursos para decodificar una subimagen actual. El decodificador de vídeo 1210 comprende además un módulo de análisis 1213 para analizar el flujo de bits para obtener el indicador de nivel de subimagen y la subimagen actual. El decodificador de vídeo 1210 comprende además un módulo de asignación 1215 para asignar recursos para decodificar la subimagen actual basándose en el indicador de nivel de subimagen. El decodificador de vídeo 1210 comprende además un módulo de decodificación 1217 para decodificar la subimagen actual para crear una secuencia de vídeo empleando los recursos asignados. El decodificador de vídeo 1110 comprende además un módulo de reenvío 1219 para reenviar la secuencia de vídeo para su visualización. El decodificador de vídeo 1210 puede configurarse además para realizar cualquiera de las etapas del procedimiento 1100.

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

También debe entenderse que no es necesario que las etapas de los métodos ejemplares expuestos en el presente documento se realicen en el orden descrito, y el orden de las etapas de dichos métodos debe entenderse como meramente ejemplar. Del mismo modo, se pueden incluir etapas adicionales en dichos métodos, y ciertas etapas se pueden omitir o combinar, en métodos consistentes con diversas modalidades de la presente descripción.

Si bien se han proporcionado varias modalidades en la presente divulgación, puede entenderse que los sistemas y métodos divulgados podrían incorporarse de muchas otras formas específicas sin apartarse del alcance de la presente divulgación. Los presentes ejemplos deben considerarse ilustrativos y no restrictivos, y la intención no se limita a los detalles ofrecidos en la presente memoria. Por ejemplo, los diversos elementos o componentes se pueden combinar o integrar en otro sistema o se pueden omitir o no implementarse determinadas características.

Además, las técnicas, sistemas, subsistemas, y métodos descritos e ilustrados en las diversas modalidades como discretos o separados se pueden combinar o integrar con otros sistemas, componentes, técnicas o métodos sin apartarse del alcance de la presente divulgación. Un experto en la técnica puede comprobar otros ejemplos de cambios, sustituciones y alteraciones, y estos podrían realizarse sin apartarse del alcance divulgado en el presente documento.

REIVINDICACIONES

1. Un método (1110) implementado en un decodificador, comprendiendo el método:

recibir (1101), por parte de un receptor del decodificador, un flujo de bits que comprende datos de una o más subimágenes divididas a partir de una imagen y un mensaje de información de mejora complementaria, SEI, en donde el mensaje SEI comprende información de un conjunto de secuencias de subimágenes que cubren colectivamente una región de imágenes rectangulares, en donde la información indica la velocidad de bits de un flujo de bits que contiene solo el conjunto de secuencias de subimágenes; en donde cada una de las una o más subimágenes es una región rectangular de uno o más grupos de mosaicos dentro de una imagen;

analizar (1103), mediante un procesador del decodificador, el flujo de bits para obtener la información del mensaje SEI;

asignar (1105), mediante el procesador, una velocidad de bits para decodificar la subimagen actual basándose en la información del mensaje SEI;

decodificar (1107), mediante el procesador, la subimagen actual para crear una secuencia de vídeo empleando la velocidad de bits asignada.

2. El método de la reivindicación 1, en donde el flujo de bits incluye un conjunto de parámetros de secuencia, SPS, que comprende identificadores de subimagen, ID, para cada una de las subimágenes.

3. El método de la reivindicación 2, en donde el SPS comprende además una ubicación de subimágenes para cada una de las subimágenes.

4. El método de la reivindicación 2, en donde el SPS comprende además un tamaño de subimagen para cada una de las subimágenes.

5. Un método implementado en un decodificador (1000), comprendiendo el método:

dividir (1001), mediante un procesador del codificador, una imagen en una pluralidad de subimágenes; en donde cada una de la pluralidad de subimágenes es una región rectangular de uno o más grupos de mosaicos dentro de una imagen;

codificar (1001), mediante el procesador, una o más de la pluralidad de subimágenes en un flujo de bits;

codificar en un flujo de bits (1005), mediante el procesador, un mensaje de información de mejora complementaria, SEI, en donde el mensaje SEI comprende información de un conjunto de secuencias de subimágenes que, en conjunto, consisten en una región de imágenes rectangular, en donde la información indica una velocidad de bits de un flujo de bits que contiene solo el conjunto de secuencias de subimágenes.

6. El método de la reivindicación 5 que comprende además:

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

7. Un decodificador (1210) que comprende:

una unidad de recepción (1211), configurada para recibir un flujo de bits que comprende datos de una o más subimágenes divididas a partir de una imagen y un mensaje de información de mejora complementaria, SEI, en donde el mensaje SEI comprende información de un conjunto de secuencias de subimágenes que, en conjunto, consisten en una región de imágenes rectangular, en donde la información indica la velocidad de bits de un flujo de bits que contiene solo el conjunto de secuencias de subimágenes; en donde cada una de las una o más subimágenes es una región rectangular de uno o más grupos de mosaicos dentro de una imagen;

una unidad de análisis (1213), configurada para analizar el flujo de bits para obtener la información del mensaje SEI;

una unidad de asignación (1215), configurada para asignar una velocidad de bits para decodificar la subimagen actual basándose en la información del mensaje SEI;

una unidad de decodificación (1217), configurada para decodificar la subimagen actual para crear una secuencia de vídeo empleando la velocidad de bits asignada.

8. Un codificador (1202) que comprende:

una unidad de partición (1201), configurada para dividir una imagen en una pluralidad de subimágenes; en donde cada una de la pluralidad de subimágenes es una región rectangular de uno o más grupos de mosaicos dentro de una imagen;

una unidad de codificación (1205), configurada para codificar para:

codificar en un flujo de bits un mensaje de información de mejora complementaria, SEI, en donde el mensaje SEI comprende información de un conjunto de secuencias de subimágenes que, en conjunto, consisten en una región de imágenes rectangular, en donde la información indica una velocidad de bits de un flujo de bits que contiene solo el conjunto de secuencias de subimágenes; y

codificar una o más de la pluralidad de subimágenes en el flujo de bits.

9. El codificador de la reivindicación 8, que comprende además:

una unidad de almacenamiento (1207), configurada para almacenar el flujo de bits para la comunicación hacia un decodificador.

10. Un flujo de bits codificado, comprendiendo el flujo de bits una o más subimágenes divididas a partir de una imagen, y que comprende un mensaje de información de mejora complementaria, SEI, en donde el mensaje SEI comprende información de un conjunto de secuencias de subimágenes que, en conjunto, consisten en una región de imágenes rectangulares, en donde la información indica la velocidad de bits de un flujo de bits que contiene solo el conjunto de secuencias de subimágenes; en donde cada una de las una o más subimágenes es una región rectangular de uno o más grupos de mosaicos dentro de una imagen.

11. Un producto de programa informático que comprende un código de programa para realizar el método según una cualquiera de las reivindicaciones 1 a 6 cuando se ejecuta en un ordenador o un procesador.

12. Un medio legible por ordenador no transitorio que transporta un código de programa que, cuando es ejecutado por un ordenador, hará que el ordenador realice el método de cualquiera de las reivindicaciones 1 a 6.

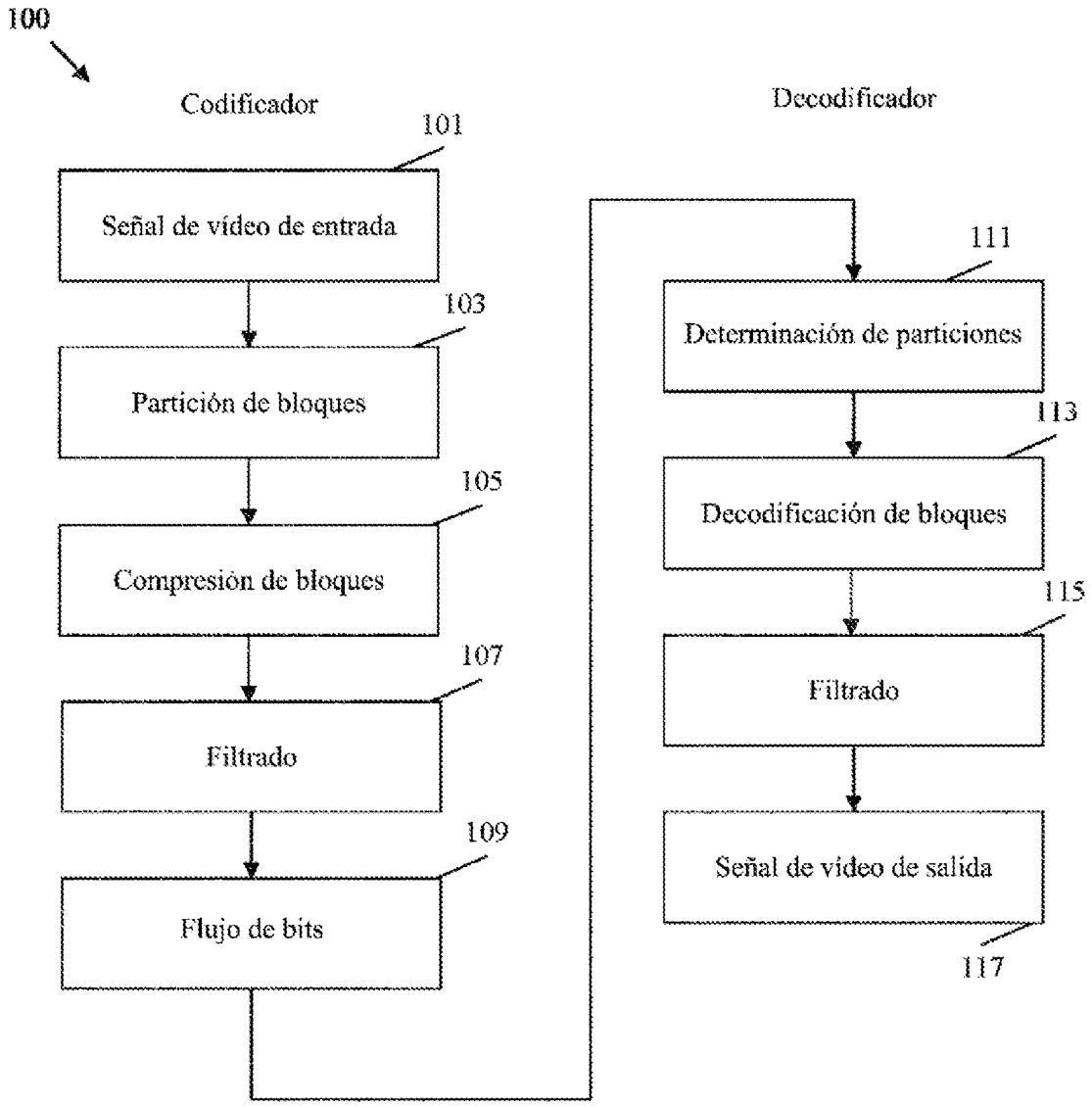
100
Codificador
Decodificador
101
Señal de vídeo de entrada
103
Partición de bloques
105
Compresión de bloques
107
Filtrado
109
Flujo de bits
111
Determinación de particiones
113
Decodificación de bloques
115
Filtrado
Señal de vídeo de salida
117

Figura 1

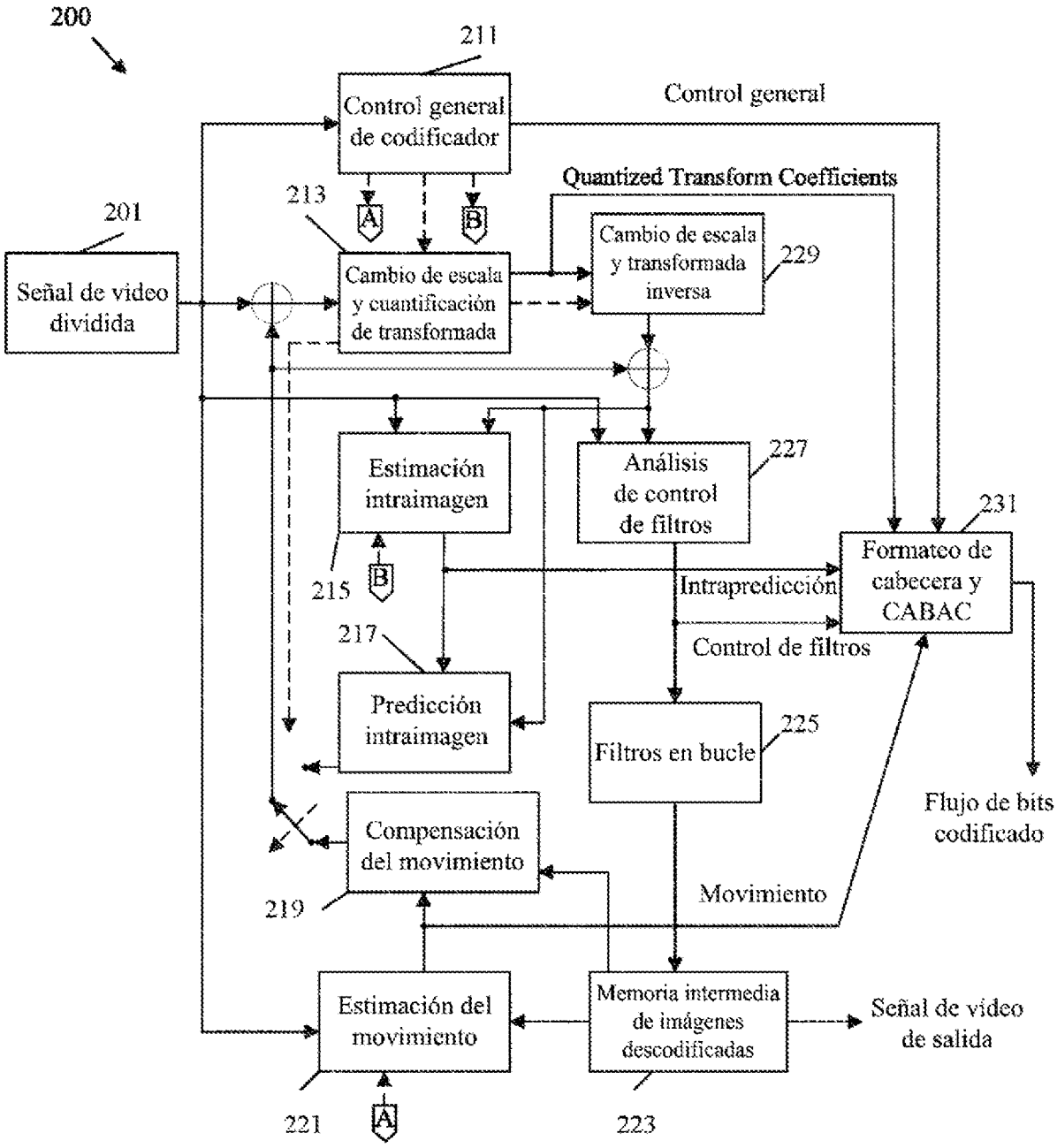
200
211
Control general de codificador
Control general
Quantized Transform Coefficients
213
A
B
201
Señal de vídeo dividida
Cambio de escala y cuantificación de transformada
Cambio de escala y transformada inversa
229
Estimación intraimagen
Análisis de control de filtros
227
231
215
B
Formateo de cabecera y CABAC
Intrapredicción
217
Control de filtros
Predicción intraimagen
Filtros en bucle
225
Flujo de bits codificado
Compensación del movimiento
219
Movimiento
Estimación del movimiento
Memoria intermedia de imágenes descodificadas
Señal de vídeo de salida
221
A
223

Figura 2

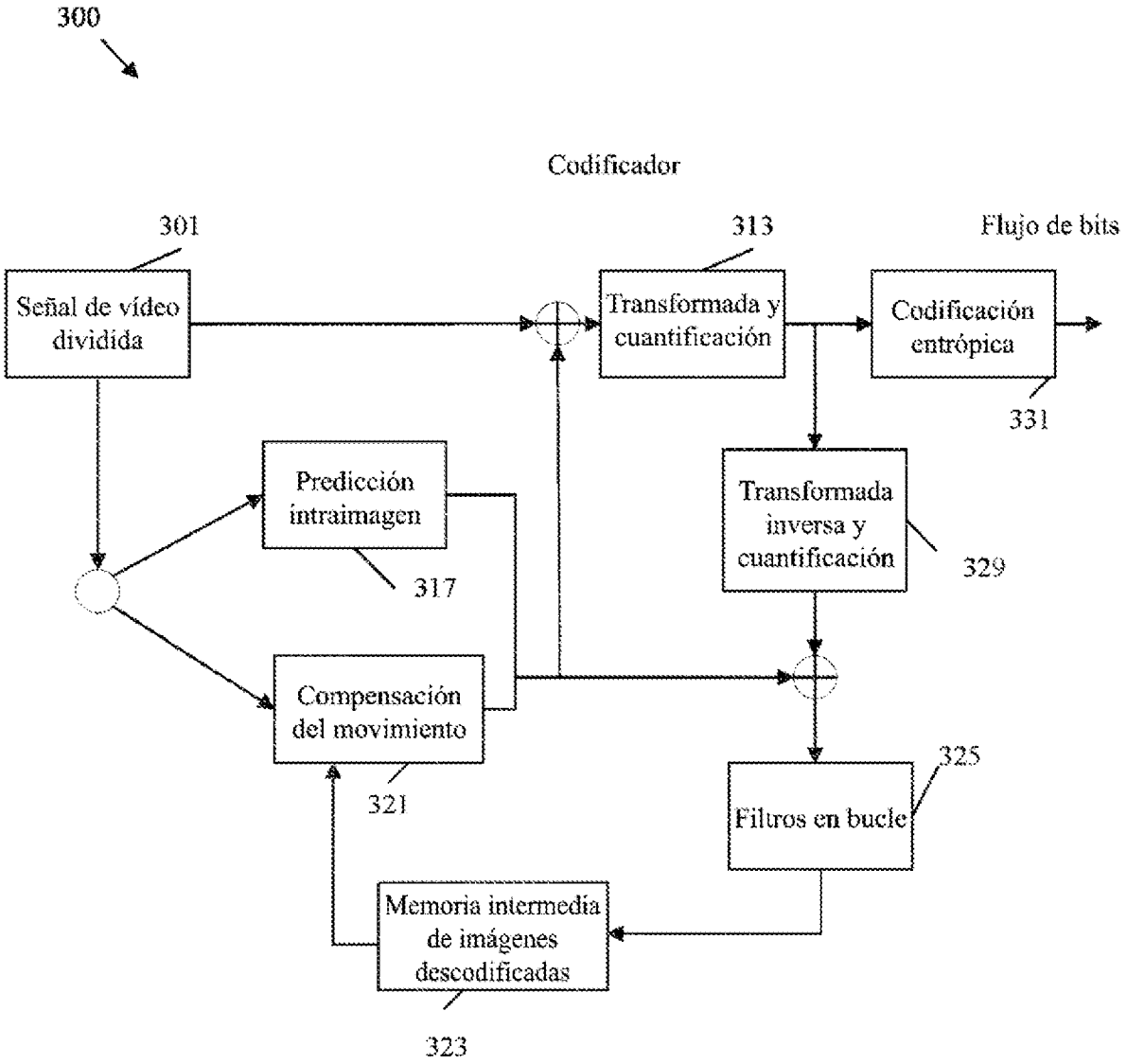
300
Codificador
301
Señal de vídeo dividida
313
Transformada y cuantificación
Flujo de bits
Codificación entrópica
331
Predicción intraimagen
317
Transformada inversa y cuantificación
329
Compensación del movimiento
321
325
Filtros en bucle
Memoria intermedia de imágenes descodificadas
323

Figura 3

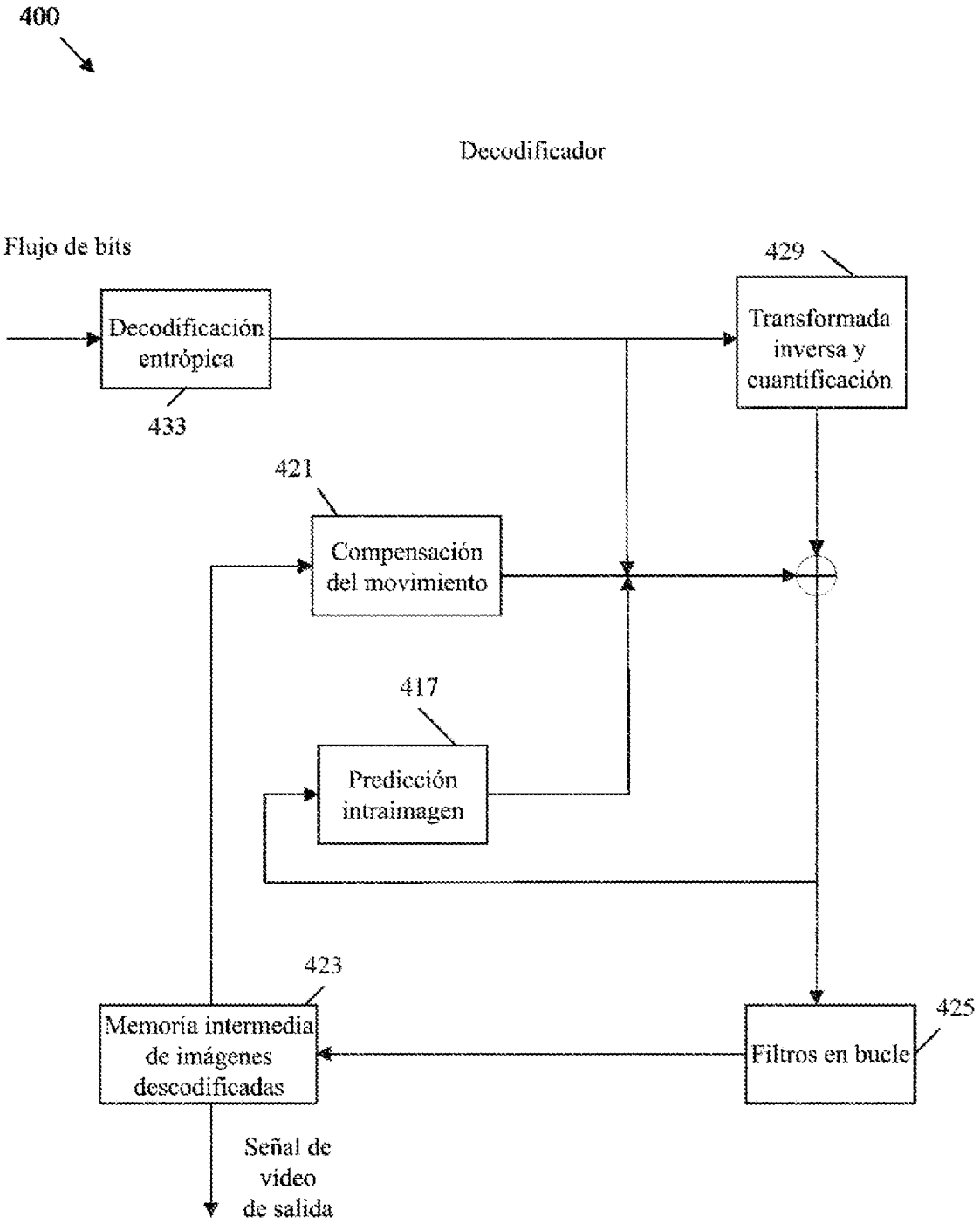
400
Decodificador
Flujo de bits
Decodificación entrópica
433
429
Transformada inversa y cuantificación
421
Compensación del movimiento
417
Predicción intraimagen
423
Memoria intermedia de imágenes descodificadas
425
Filtros en bucle
Señal de vídeo de salida

Figura 4

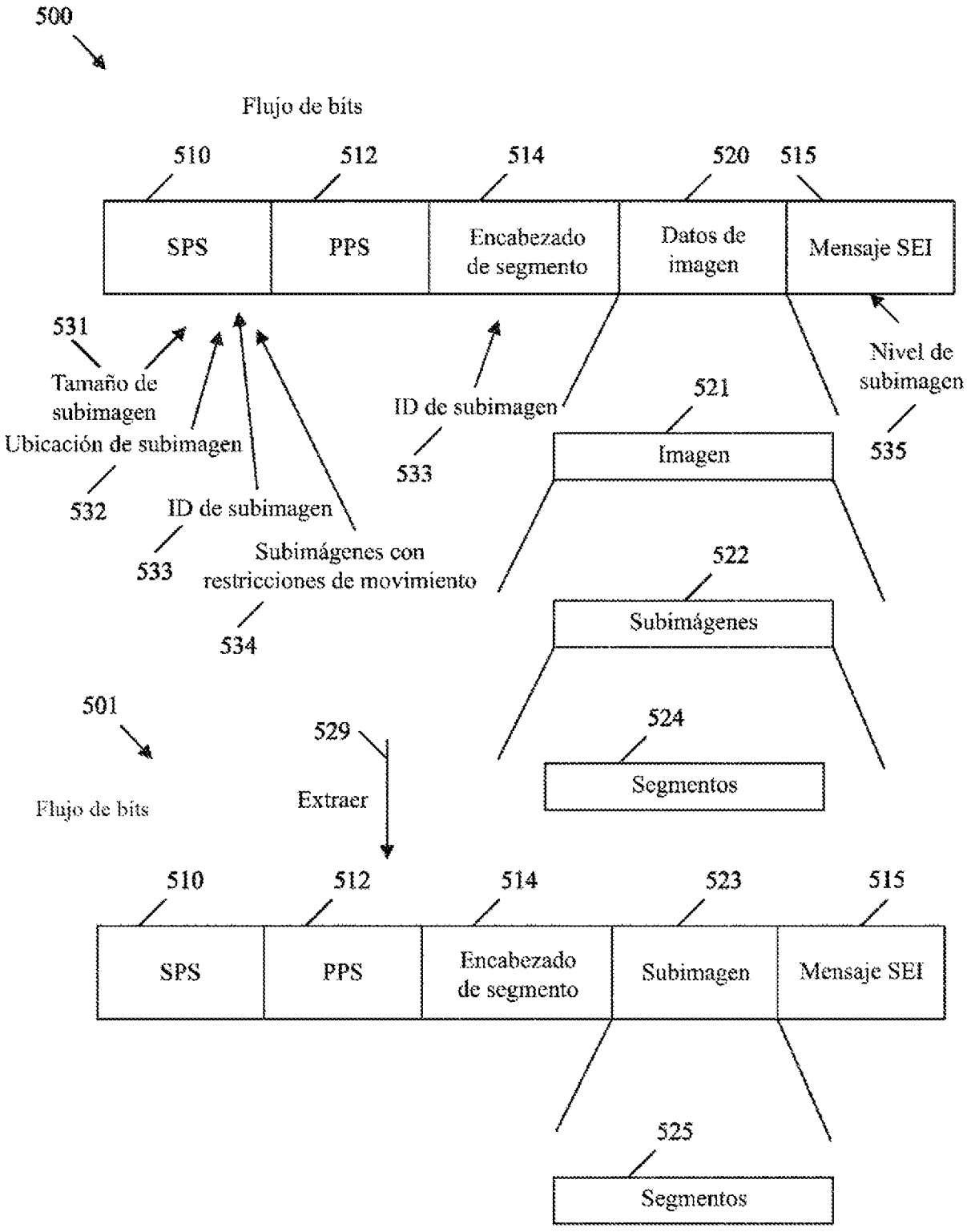
500
Flujo de bits
510
512
514
520
515
SPS
PPS
Encabezado de segmento
Datos de imagen
Mensaje SEI
531
Tamaño de subimagen
Ubicación de subimagen
532
ID de subimagen
533
Subimágenes con restricciones de movimiento
534
ID de subimagen
533
Nivel de subimagen
535
521
Imagen
522
Subimágenes
524
Segmentos
501
Flujo de bits
529
Extraer
510
512
514
523
515
SPS
PPS
Encabezado de segmento
Subimagen
Mensaje SEI
525
Segmentos

Figura 5

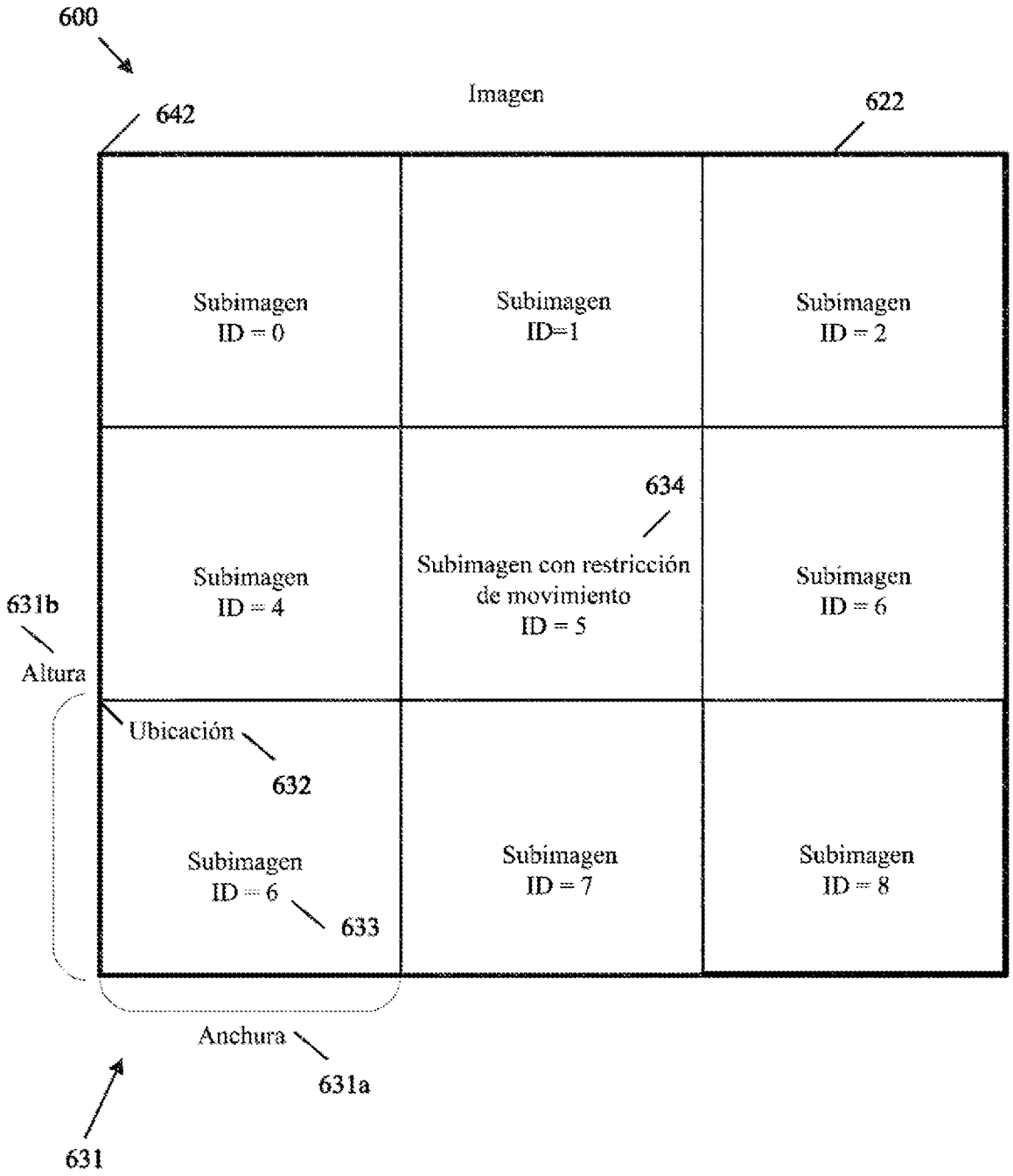
600
642
Imagen
622
Subimagen
ID = 0
Subimagen
ID=1
Subimagen
ID = 2
Subimagen
ID = 4
634
Subimagen con restricción
de movimiento
ID = 5
Subimagen
ID = 6
631b
Altura
Ubicación
632
Subimagen
ID = 6
633
Subimagen
ID = 7
Subimagen
ID = 8
Anchura
631a
631

Figura 6

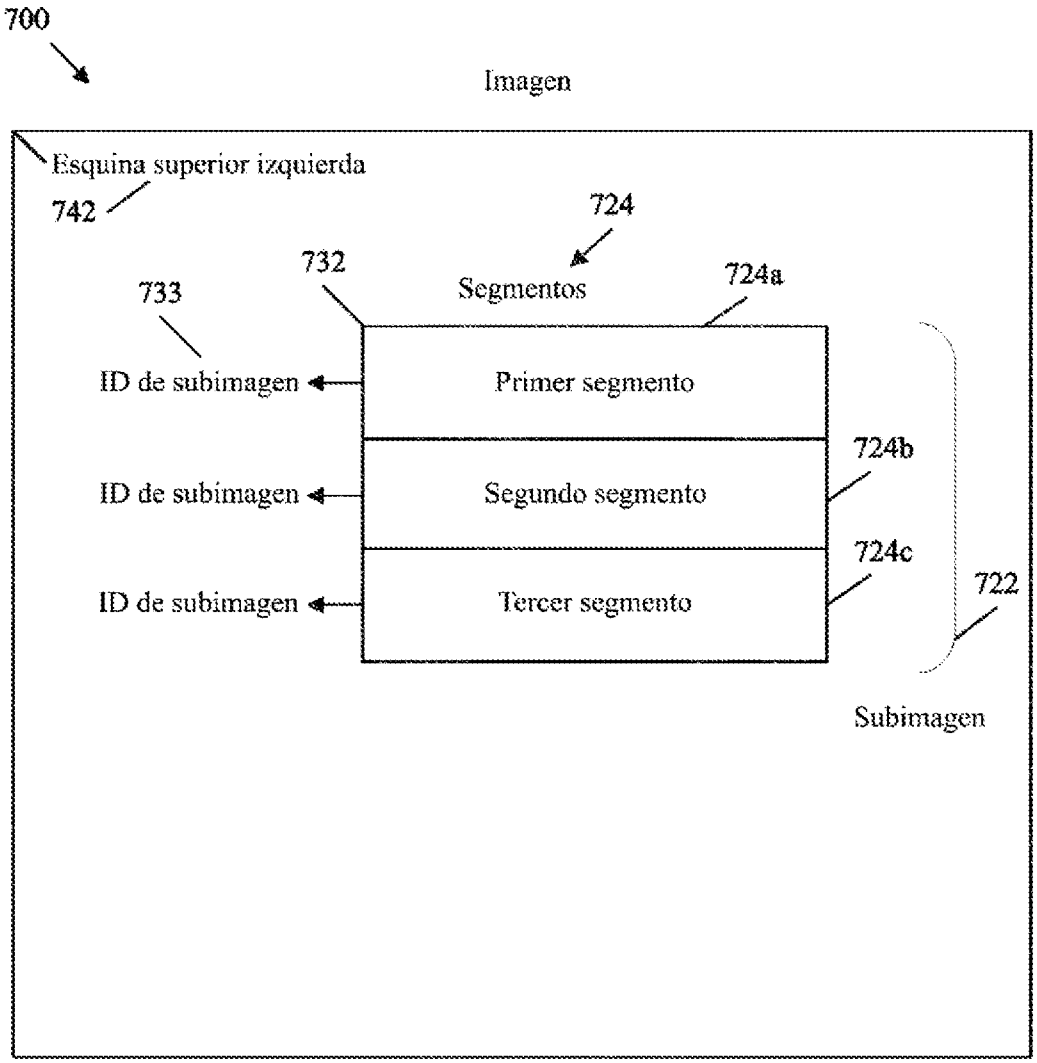
700
Imagen
Esquina superior izquierda
742
724
732
Segmentos
724a
733
ID de subimagen
Primer segmento
ID de subimagen
Segundo segmento
724b
ID de subimagen
Tercer segmento
724c
722
Subimagen

Figura 7

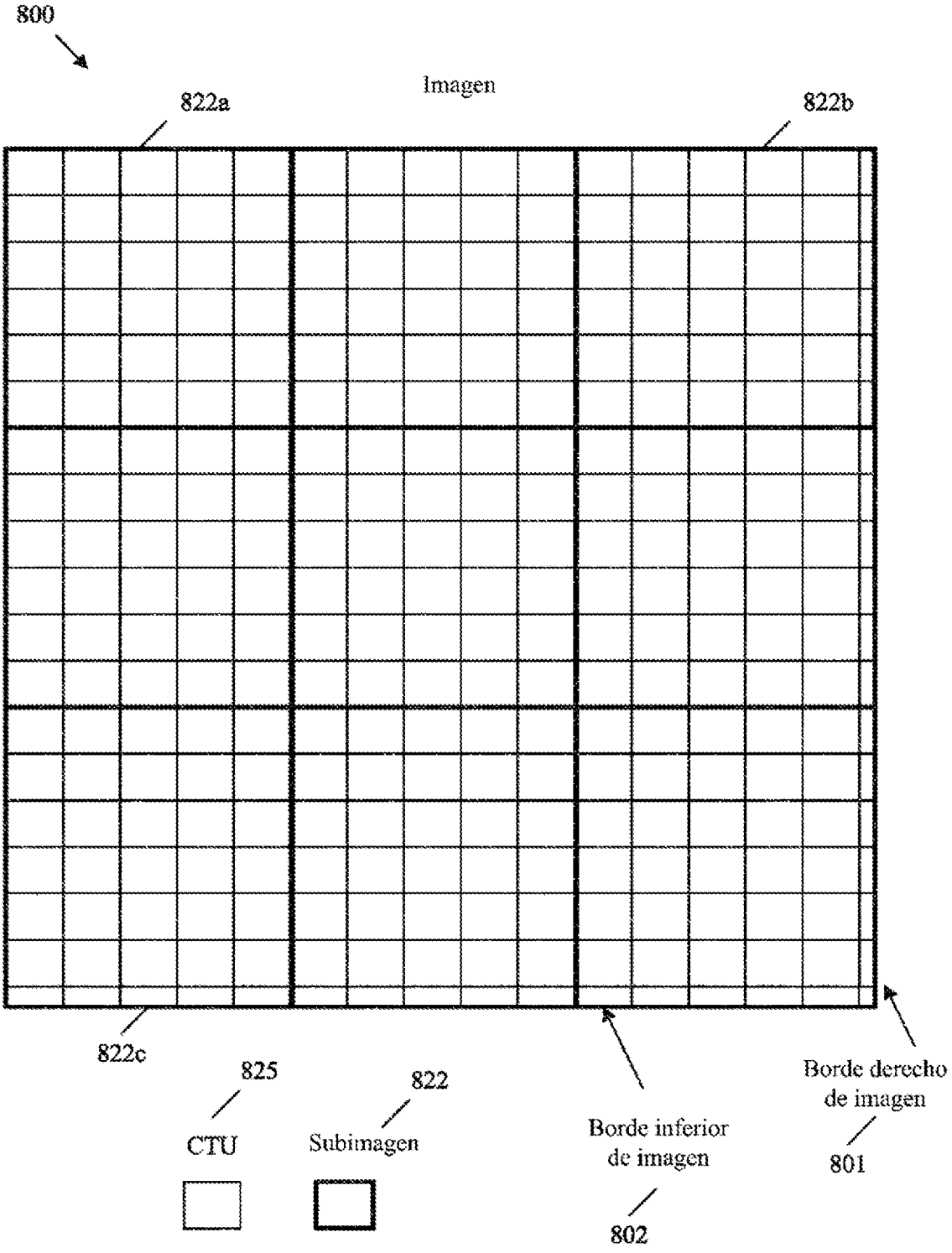
800
Imagen
822a
822b
822c
825
CTU
822
Subimagen
Borde inferior de imagen
802
Borde derecho de imagen
801

Figura 8

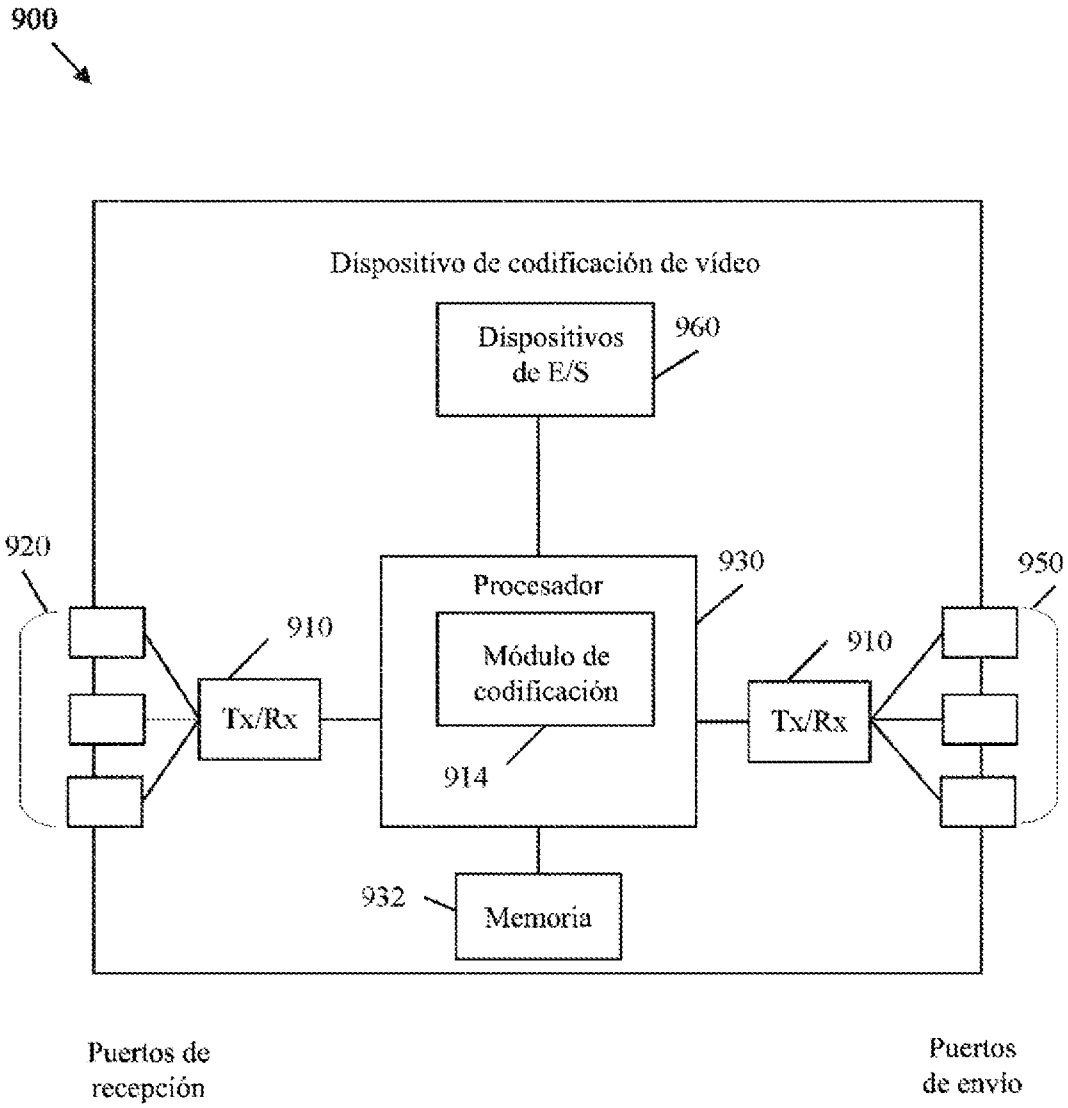
900
Dispositivo de codificación de vídeo
Dispositivos de E/S
960
920
910
Tx/Rx
Procesador
930
Módulo de codificación
914
910
Tx/Rx
950
932
Memoria
Puertos de recepción
Puertos de envío

Figura 9

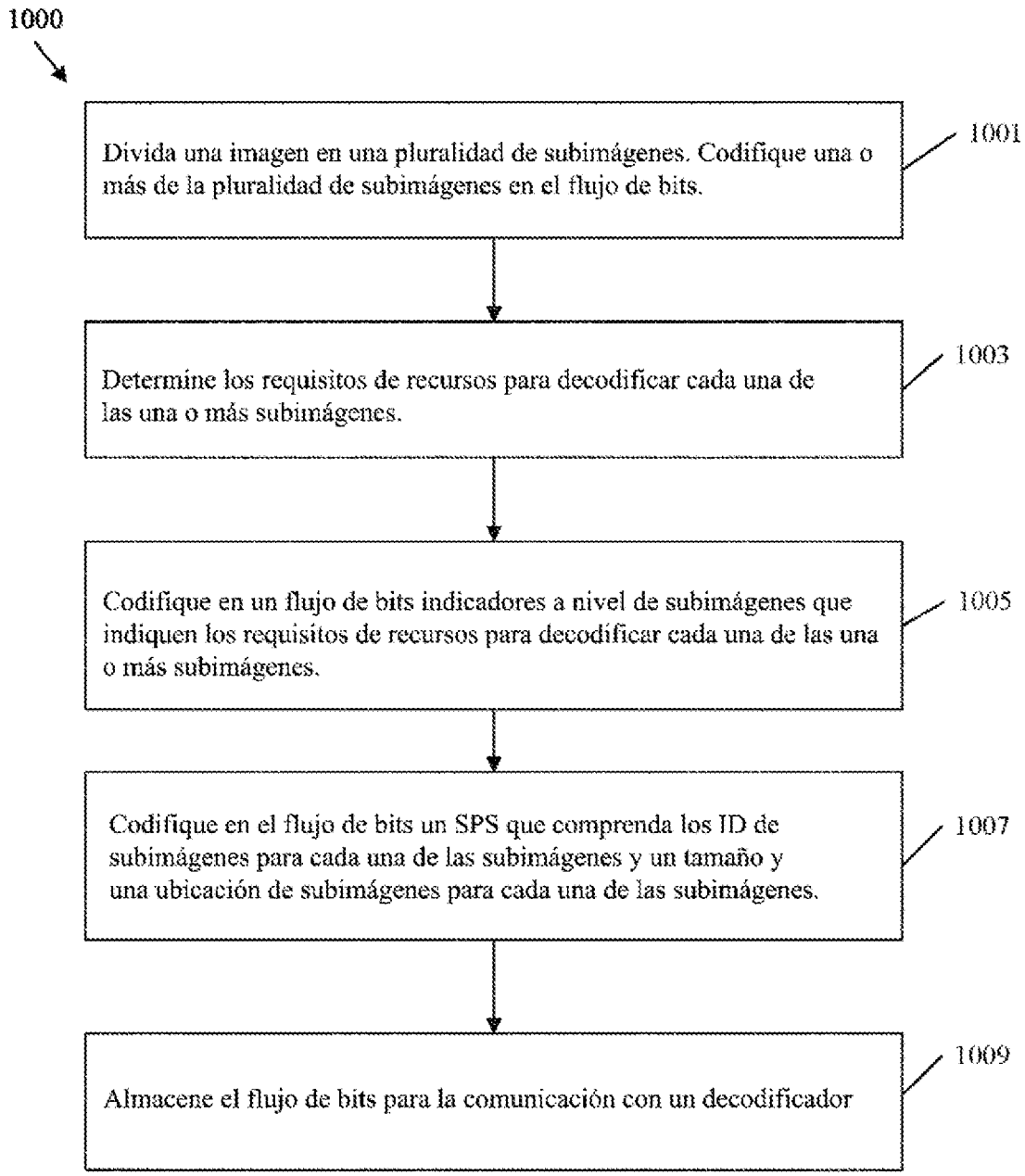
1000
Divida una imagen en una pluralidad de subimágenes. Codifique una o más de la pluralidad de subimágenes en el flujo de bits.
1001
Determine los requisitos de recursos para decodificar cada una de las una o más subimágenes.
1003
Codifique en un flujo de bits indicadores a nivel de subimágenes que indiquen los requisitos de recursos para decodificar cada una de las una o más subimágenes.
1005
Codifique en el flujo de bits un SPS que comprenda los ID de subimágenes para cada una de las subimágenes y un tamaño y una ubicación de subimágenes para cada una de las subimágenes.
1007
Almacene el flujo de bits para la comunicación con un decodificador
1009

Figura 10

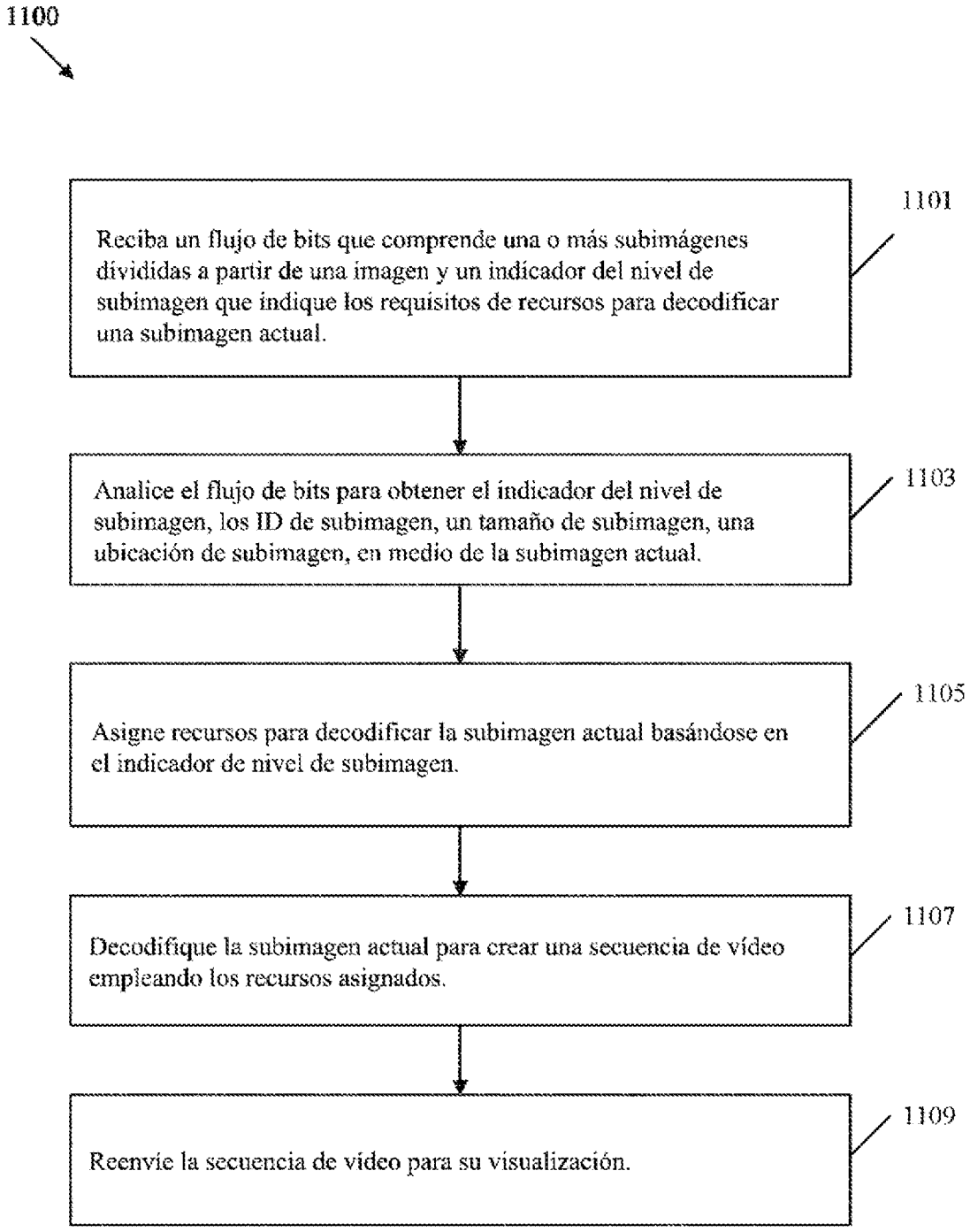
1100
Reciba un flujo de bits que comprende una o más subimágenes divididas a partir de una imagen y un indicador del nivel de subimagen que indique los requisitos de recursos para decodificar una subimagen actual.
1101
Analice el flujo de bits para obtener el indicador del nivel de subimagen, los ID de subimagen, un tamaño de subimagen, una ubicación de subimagen, en medio de la subimagen actual.
1103
Asigne recursos para decodificar la subimagen actual basándose en el indicador de nivel de subimagen.
1105
Decodifique la subimagen actual para crear una secuencia de vídeo empleando los recursos asignados.
1107
Reenvíe la secuencia de vídeo para su visualización.
1109

Figura 11

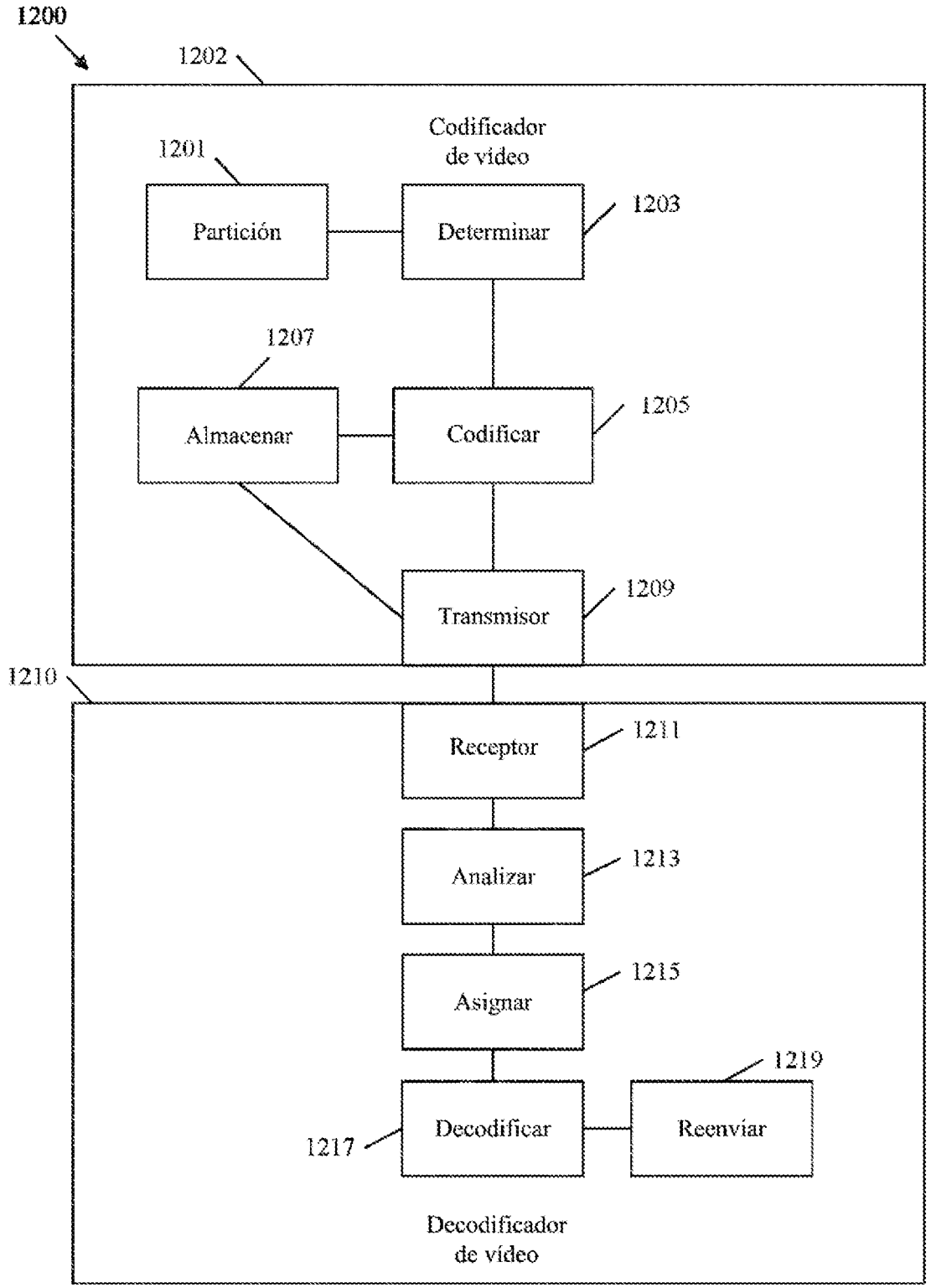
1200
1202
Codificador de vídeo
1201
Partición
Determinar
1203
1207
Almacenar
Codificar
1205
Transmisor
1209
1210
Receptor
1211
Analizar
1213
Asignar
1215
1219
Decodificar
Reenviar
1217
Decodificador de vídeo

Figura 12