(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **3 007 437**

(51) Int. CI.:

H04N 19/00 (2014.01)
H04N 19/102 (2014.01)
H04N 19/119 (2014.01)
H04N 19/156 (2014.01)
H04N 19/176 (2014.01)
H04N 19/70 (2014.01)
H04N 19/96 (2014.01)
H04N 19/436 (2014.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **27.07.2020 PCT/CN2020/104786**

(87) Fecha y número de publicación internacional: **04.02.2021 WO21018083**

(96) Fecha de presentación y número de la solicitud europea: **27.07.2020 E 20847404 (9)**

(97) Fecha y número de publicación de la concesión europea: **08.01.2025 EP 3987776**

(54) Título: **Uso dependiente de tamaño de bloque de modo de codificación de vídeo**

(30) Prioridad:

26.07.2019 WO PCT/CN2019/097926
31.08.2019 WO PCT/CN2019/103892

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
20.03.2025

(73) Titular/es:

BEIJING BYTEDANCE NETWORK TECHNOLOGY CO., LTD. (50.00%)
Room B-0035, 2/F, No.3 Building No. 30, Shixing Road Shijingshan District
Beijing 100041, CN y
BYTEDANCE INC. (50.00%)

(72) Inventor/es:

DENG, ZHIPIN;
ZHANG, LI;
ZHANG, KAI y
LIU, HONGBIN

(74) Agente/Representante:

VALLEJO LÓPEZ, Juan Pedro

DESCRIPCIÓN

Uso dependiente de tamaño de bloque de modo de codificación de vídeo

Referencia cruzada a solicitudes relacionadas

Esta solicitud se basa en la solicitud de patente internacional n.° PCT/CN2020/104786, presentada el 27 de julio de 2020, que reivindica la prioridad y los beneficios de la solicitud de patente internacional n.° PCT/CN2019/097926 presentada el 26 de julio de 2019 y la solicitud de patente internacional n.° PCT/CN2019/103892 presentada el 31 de agosto de 2019.

Campo técnico

Este documento está relacionado con tecnologías de codificación y descodificación de vídeo e imagen.

Antecedentes

El vídeo digital representa el mayor uso de ancho de banda en Internet y otras redes de comunicación digital. A medida que aumenta el número de dispositivos de usuario conectados capaces de recibir y mostrar vídeo, se espera que la demanda de ancho de banda para el uso de vídeo digital continúe creciendo. Bibliografía no de patente S-T HSIANG ET AL: "CE1.6.1: Coding large size CUs",11. REUNIÓN de JVET; 20180711 - 20180718; LJUBLJANA; (EL EQUIPO MIXTO DE EXPLORACIÓN DE VÍDEO DE ISO/IEC JTC1/SC29/WG11 E ITU-T SG.16), n.° JVET-K0227 10 de julio de 2018 (10-07-2018), URL:http://phenix.int-evry.fr/jvet/doc_end_user/documents/11_Ljubljana/wg11/JVET-K0227-v2.zip JVET-K0227-v1.docx divulga una técnica para codificar las CU de gran tamaño;

Bibliografía no de patente SUZUKI (SONY) T ET AL: "Description of SDR and HDR video coding technology proposal by Sony",10. REUNIÓN de JVET; 20180410 - 20180420; SAN DIEGO; (EL EQUIPO MIXTO DE EXPLORACIÓN DE VÍDEO DE ISO/IEC JTC1/SC29/WG11 E ITU-T SG.16), n.° JVET-J0028 12 de abril de 2018 (12-04-2018), URL:http://phenix.int-evry.fr/jvet/doc_end_user/documents/10_San %20Diego/wg11/JVET-J0028-v2.zip JVET-J0028.docx divulga una técnica relacionada con la selección de estructura de partición;

Sumario

Las técnicas divulgadas pueden usarse por realizaciones de codificador o descodificador de vídeo o imagen para realizar codificación o descodificación de vídeo en las que el modo de partición de imagen se determina basándose en tamaño de bloque.

En un aspecto de ejemplo, se divulga un método de procesamiento de vídeo. El método incluye usar una dimensión de una unidad de datos de canalización virtual (VPDU) usada para una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo y una representación de flujo de bits del vídeo para realizar una determinación de si se habilita una partición de árbol ternario (TT) o árbol binario (BT) de un bloque de vídeo de los uno o más bloques de vídeo, y realizar, basándose en la determinación, la conversión, en donde la dimensión es igual a VSize en muestras de luma, en donde las dimensiones del bloque de vídeo son CtbSizeY en muestras de luma, en donde VSize = mín(M, CtbSizeY), y en donde M es un número entero positivo.

En otro aspecto de ejemplo, se divulga un método de procesamiento de vídeo. El método incluye usar, para una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo y una representación de flujo de bits del vídeo, una dimensión de un bloque de vídeo de los uno o más bloques de vídeo para realizar una determinación de si se habilita una partición de árbol ternario (TT) o de árbol binario (BT) del bloque de vídeo, y realizar, basándose en la determinación, la conversión.

En otro aspecto de ejemplo más, se divulga un método de procesamiento de vídeo. El método incluye usar una altura o una anchura de un bloque de vídeo para realizar una determinación de si se habilita una herramienta de codificación para una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo que comprenden el bloque de vídeo y una representación de flujo de bits del vídeo, y realizar, basándose en la determinación, la conversión, en donde la determinación se basa en una comparación entre la altura o la anchura con un valor N, donde N es un número entero positivo.

En otro aspecto de ejemplo más, se divulga un método de procesamiento de vídeo. El método incluye usar la comparación entre una altura o una anchura de un bloque de vídeo y un tamaño de un bloque de transformada para realizar una determinación de si se habilita una herramienta de codificación para una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo que comprenden el bloque de vídeo y una representación de flujo de bits del vídeo, y realizar, basándose en la determinación, la conversión.

En otro aspecto de ejemplo más, se divulga un método de procesamiento de vídeo. El método incluye usar una altura o una anchura de un bloque de vídeo para realizar una determinación de si se habilita una herramienta de codificación

para una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo que comprenden el bloque de vídeo y una representación de flujo de bits del vídeo, y realizar, basándose en la determinación, la conversión.

En otro aspecto de ejemplo más, se divulga un método de procesamiento de vídeo. El método incluye usar una comparación entre una dimensión de una subpartición de un bloque de vídeo y un tamaño de transformada máximo para realizar (a) una determinación de si se habilita un modo de predicción intra subpartición (ISP) para una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo que comprenden el bloque de vídeo, y (b) una selección de uno o más tipos de partición permitidos para la conversión, y realizar, basándose en la determinación y la selección, la conversión, en donde, en el modo de ISP, un bloque de vídeo de los uno o más bloques de vídeo se particiona en múltiples subparticiones antes de la aplicación de una intra-predicción y transformada.

En otro aspecto de ejemplo más, se divulga un método de procesamiento de vídeo. El método incluye realizar una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo y una representación de flujo de bits del vídeo, en donde la conversión comprende una herramienta de codificación que se ha deshabilitado, y en donde los elementos de sintaxis relacionados con la herramienta de codificación se excluyen de la representación de flujo de bits y se infiere que son un valor predeterminado que especifica que la herramienta de codificación está deshabilitada.

En otro aspecto de ejemplo más, se divulga un método de procesamiento de vídeo. El método incluye realizar una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo y una representación de flujo de bits del vídeo, en donde la conversión comprende una herramienta de codificación que se ha deshabilitado, y en donde la representación de flujo de bits comprende elementos de sintaxis relacionados con la herramienta de codificación que se infiere que son un valor predeterminado basándose en la herramienta de codificación que está deshabilitada.

En otro aspecto de ejemplo más, se divulga un método de procesamiento de vídeo. El método incluye usar una dimensión de una unidad de datos de canalización virtual (VPDU) y/o un tamaño de transformada máximo usados para una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo y una representación de flujo de bits del vídeo para realizar una determinación de si se habilita una partición implícita (QT) de un bloque de vídeo de los uno o más bloques de vídeo, y realizar, basándose en la determinación, la conversión.

En otro aspecto de ejemplo más, se divulga un método de procesamiento de vídeo. El método incluye realizar una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo y una representación de flujo de bits del vídeo, en donde la conversión comprende una transformada de subbloque (SBT), en donde una altura máxima o una anchura máxima de la SBT se basa en un tamaño de transformada máximo, y en donde la SBT comprende una o más transformadas que se aplican de manera separada a una o más particiones de un bloque de vídeo de los uno o más bloques de vídeo.

En otro aspecto de ejemplo más, se divulga un método de procesamiento de vídeo. El método incluye realizar una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo y una representación de flujo de bits del vídeo, en donde la conversión comprende un modo de omisión de transformada y/o un modo de modulación de código de pulso diferencial basada en intra bloque (BDPCM), en donde un tamaño de bloque máximo usado para el modo de omisión de transformada se basa en un tamaño de transformada máximo, en donde el modo de omisión de transformada comprende omitir procesos de transformada y transformada inversa para una herramienta de codificación correspondiente, y en donde, en el modo de BDPCM, un residual de una intra predicción del bloque de vídeo actual se codifica de manera predictiva usando una operación de modulación de codificación de pulso diferencial.

En otro aspecto de ejemplo más, se divulga un método de procesamiento de vídeo. El método incluye usar una comparación entre una altura o una anchura de un bloque de vídeo y un tamaño de transformada máximo para realizar una determinación de si se habilita un modo de inter intra predicción combinado (CIIP) para una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo que comprenden el bloque de vídeo y una representación de flujo de bits del vídeo, y realizar, basándose en la determinación, la conversión, en donde, en el modo de CIIP, una predicción final del bloque de vídeo se basa en una suma ponderada de una inter predicción del bloque de vídeo y una intra predicción del bloque de vídeo.

En otro aspecto de ejemplo más, se divulga un método de procesamiento de vídeo. El método incluye hacer una determinación, para una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo y una representación de flujo de bits del vídeo, con respecto a la partición de un bloque de vídeo de los uno o más bloques de vídeo codificados con inter intra predicción combinada (CIIP), y realizar, basándose en la determinación, la conversión, en donde, en el modo de CIIP, una predicción final del bloque de vídeo se basa en una suma ponderada de una inter predicción del bloque de vídeo y una intra predicción del bloque de vídeo.

En otro aspecto de ejemplo más, se divulga un método de procesamiento de vídeo. El método incluye realizar una conversión entre un vídeo que comprende una región de vídeo que comprende múltiples bloques de vídeo y una representación de flujo de bits del vídeo de acuerdo con una regla, en donde la regla especifica que un tamaño de bloque máximo de los múltiples bloques de vídeo en la región de vídeo que se codifican en la representación de flujo de bits usando una codificación de transformada determina un tamaño de bloque máximo de los múltiples bloques de vídeo en la región de vídeo que se codifican en la representación de flujo de bits sin usar codificación de transformada.

En otro aspecto de ejemplo más, se divulga un método de procesamiento de vídeo. El método incluye realizar una conversión entre un vídeo que comprende una región de vídeo que comprende múltiples bloques de vídeo y una representación de flujo de bits del vídeo de acuerdo con una regla, en donde la regla especifica que un proceso de mapeo de luma con escalado de croma (LMCS) está deshabilitado para la región de vídeo cuando la codificación sin pérdidas está habilitada para la región de vídeo, en donde la región de vídeo es una secuencia, una imagen, una subimagen, un corte, un grupo de mosaicos, un mosaico, un ladrillo, una fila de unidad de árbol de codificación (CTU), una CTU, una unidad de codificación (CU), una unidad de predicción (PU), una unidad de transformada (TU) o un subbloque, y en donde el proceso de LMCS comprende muestras de luma de la región de vídeo que se vuelven a conformar entre un primer dominio y un segundo dominio y un residual de croma que se escala de una manera dependiente de luma.

En otro aspecto de ejemplo más, el método descrito anteriormente puede implementarse mediante un aparato codificador de vídeo que comprende un procesador.

En otro aspecto de ejemplo más, estos métodos pueden realizarse en forma de instrucciones ejecutables por procesador y almacenarse en un medio de programa legible por ordenador.

Estos y otros aspectos se describen adicionalmente en el presente documento.

Breve descripción de los dibujos

La figura 1 muestra ejemplos de división de árbol binario (BT) y árbol ternario (TT) dependiendo del tamaño de bloque de vídeo.
La figura 2 es un diagrama de bloques de un ejemplo de una plataforma de hardware usada para implementar técnicas descritas en el presente documento.
La figura 3 es un diagrama de bloques de un sistema de procesamiento de vídeo de ejemplo en el que pueden implementarse técnicas divulgadas.
La figura 4 es un diagrama de flujo para un método de ejemplo de procesamiento de vídeo.
La figura 5 es un diagrama de flujo para otro método de ejemplo de procesamiento de vídeo.
La figura 6 es un diagrama de flujo para otro método de ejemplo más de procesamiento de vídeo.
La figura 7 es un diagrama de flujo para otro método de ejemplo más de procesamiento de vídeo.
La figura 8 es un diagrama de flujo para otro método de ejemplo más de procesamiento de vídeo.
La figura 9 es un diagrama de flujo para otro método de ejemplo más de procesamiento de vídeo.
La figura 10 es un diagrama de flujo para otro método de ejemplo más de procesamiento de vídeo.
La figura 11 es un diagrama de flujo para otro método de ejemplo más de procesamiento de vídeo.
La figura 12 es un diagrama de flujo para otro método de ejemplo más de procesamiento de vídeo.
La figura 13 es un diagrama de flujo para otro método de ejemplo más de procesamiento de vídeo.
La figura 14 es un diagrama de flujo para otro método de ejemplo más de procesamiento de vídeo.
La figura 15 es un diagrama de flujo para otro método de ejemplo más de procesamiento de vídeo.
La figura 16 es un diagrama de flujo para otro método de ejemplo más de procesamiento de vídeo.
La figura 17 es un diagrama de flujo para otro método de ejemplo más de procesamiento de vídeo.
La figura 18 es un diagrama de flujo para otro método de ejemplo más de procesamiento de vídeo.

Descripción detallada

El presente documento proporciona diversas técnicas que pueden usarse por un descodificador de flujos de bits de imagen o vídeo para mejorar la calidad de vídeo o imágenes digitales descomprimidos o descodificados. Por razones de brevedad, el término "vídeo" se usa en el presente documento para incluir tanto una secuencia de fotos (llamadas tradicionalmente vídeo) como imágenes individuales. Además, un codificador de vídeo también puede implementar estas técnicas durante el proceso de codificación para reconstruir fotogramas descodificados usados para codificación adicional.

Los encabezamientos de sección se usan en el presente documento para facilitar la comprensión y no limitan las realizaciones y técnicas a las secciones correspondientes. Como tal, las realizaciones de una sección se pueden combinar con realizaciones de otras secciones.

1. Sumario

Este documento está relacionado con tecnologías de codificación de vídeo. Específicamente, se trata de reglas para controlar el tamaño de la unidad de árbol de codificación o la unidad de transformada en la codificación y descodificación de vídeo. Puede aplicarse a la norma de codificación de vídeo existente como HEVC, o la norma (codificación de vídeo versátil) a finalizar. También puede ser aplicable a futuras normas de codificación de vídeo o códec de vídeo.

2. Análisis inicial

Las normas de codificación de vídeo han evolucionado principalmente a través del desarrollo de las bien conocidas normas ITU-T e ISO/IEC. La UIT-T produjo H.261 y H.263, ISO/IEC produjo MPEG-1 y MPEG-4 Visual, y las dos organizaciones produjeron conjuntamente las normas H.262/MPEG-2 Video y H.264/MPEG-4 codificación de vídeo avanzada (AVC) y H.265/HEVC. Desde H.262, las normas de codificación de vídeo se basan en la estructura de codificación de vídeo híbrida en donde se usa predicción temporal más codificación de transformada. Para explorar las futuras tecnologías de codificación de vídeo más allá de HEVC, el Equipo Mixto de Exploración de Vídeo (JVET) fue fundado por VCEG y MPEG conjuntamente en 2015. Desde entonces, JVET ha adoptado muchos métodos nuevos y los ha puesto en el software de referencia denominado Modelo de Exploración Mixta (JEM). La reunión de JVET se lleva a cabo simultáneamente una vez cada trimestre, y la nueva norma de codificación tiene como objetivo una reducción de tasa de bits del 50 % en comparación con HEVC. La nueva norma de codificación de vídeo se denominó oficialmente codificación de vídeo versátil (VVC) en la reunión del JVET de abril de 2018, y la primera versión del modelo de prueba de VVC (VTM) se liberó en ese momento. Como hay un esfuerzo continuo que contribuye a la normalización de VVC, se están adoptando nuevas técnicas de codificación para la norma de VVC en cada reunión de JVET. El borrador de trabajo de VVC y el modelo de prueba VTM se actualizan entonces después de cada reunión. El proyecto de VVC ahora tiene como objetivo la finalización técnica (FDIS) en la reunión de julio de 2020.

2.1 Tamaño de CTU en VVC

El software VTM-5.0 permite 4 tamaños de CTU diferentes: 16x16, 32x32, 64x64 y 128x128. Sin embargo, en la reunión de JVET de julio de 2019, el tamaño mínimo de CTU se redefinió a 32x32 debido a la adopción de JVET-O0526. Y el tamaño de CTU en el borrador de trabajo de VVC 6 se codifica en la cabecera de SPS en un elemento de sintaxis codificado por UE denominado log2_ctu_size_minus_5.

A continuación, se encuentran las modificaciones de especificación correspondientes en el borrador 6 de VVC con la definición de unidades de datos de canalización virtual (VPDU) y la adopción de JVET-O0526.

7.3.2.3. Sintaxis de RBSP de conjunto de parámetros de secuencia

seq_parameter_set_rbsp() {	***Descriptor***
...	
log2_ctu_size_minus5	*u(2)*
...	

7.4.3.3. Semántica de RBSP de conjunto de parámetros de secuencia

*... **log2_ctu_size_minus5** plus 5 especifica el tamaño de bloque de árbol de codificación de luma de cada CTU. Es un requisito de conformidad de flujo de bits que el valor de log2_ctu_size_minus5 sea menor o igual que 2. **log2_min_luma_coding_block_size_minus2** plus 2 especifica el tamaño de bloque de codificación de luma mínimo. Las variables CtbLog2SizeY, CtbSizeY, MinCbLog2SizeY, MinCbSizeY, IbcBufWidthY, IbcBufWidthC y Vsize se derivan como sigue:*

$$CtbLog2SizeY = log2\_ctu\_size\_minus5 + 5 \quad (7\text{-}15)$$

$$CtbSizeY = 1 << CtbLog2SizeY \quad (7\text{-}16)$$

$$MinCbLog2SizeY = log2\_min\_luma\_coding\_block\_size\_minus2 \quad 2 \quad (7\text{-}17)$$

$$MinCbSizeY = 1 << MinCbLog2SizeY \quad (7\text{-}18)$$

$$IbcBufWidthY = 128 * 128 / CtbSizeY \quad (7\text{-}19)$$

$$IbcBufWidthC = IbcBufWidthY / SubWidthC \quad (7\text{-}20)$$

$$VSize = Min(64, CtbSizeY) \quad (7\text{-}21)$$

Las variables Ctb WidthC y CtbHeightC, que especifican la anchura y la altura, respectivamente, de la matriz para cada croma CTB, se derivan como sigue:

- *Si chroma_format_idc es igual a 0 (monocromo) o separate_colour_plane_flag es igual a 1, Ctb WidthC y CtbHeightC son ambas iguales a 0.*

- *De lo contrario, CtbWidthC y CtbHeightC se derivan como sigue:*

$$CtbWidthC = CtbSizeY / SubWidthC \quad (7\text{-}22)$$

$$CtbHeightC = CtbSizeY / SubHeightC \quad (7\text{-}23)$$

Para log2BlockWidth que varía de 0 a 4 y para log2BlockHeight que varía de 0 a 4, inclusive, se invoca el proceso de inicialización de matriz de orden de exploración de ráster y diagonal vertical como se especifica en la cláusula 6.5.2 con 1 << log2BlockWidth y 1 << log2BlockHeight como entradas, y la salida se asigna a

DiagScanOrder[log2BlockWidth][log2BlockHeight] y
RasterScanOrder[log2BlockWidth][log2BlockHeight].

slice_log2_diff_max_bt_min_qt_luma *especifica la diferencia entre el logaritmo de base 2 del tamaño máximo (anchura o altura) en muestras de luma de un bloque de codificación de luma que se puede dividir usando una división binaria y el tamaño mínimo (anchura o altura) en muestras de luma de un bloque hoja de luma resultante de la división de árbol cuádruple de una CTU en el corte actual. El valor de slice_log2_diff_max_bt_min_qt_luma deberá estar en el intervalo de 0 a CtbLog2SizeY - MinQtLog2SizeY, inclusive. Cuando no está presente, el valor de slice_log2_diff_max_bt_min_qt_luma se infiere como sigue:*

- *Si slice_type igual a 2 (I), el valor de slice_log2_diff_max_bt_min_qt_luma se infiere que es igual a sps_log2_diff_max_bt_min_qt_intra_slice_luma*
- *De lo contrario (slice _type igual a 0 (B) o 1 (P)), se infiere que el valor de slice_log2_diff_max_bt_min_qt_luma es igual a sps_log2_diff_max_bt_min_qt_inter_slice.*

slice_log2_diff_max_tt_min_qt_luma *especifica la diferencia entre el logaritmo de base 2 del tamaño máximo (anchura o altura) en muestras de luma de un bloque de codificación de luma que se puede dividir usando una división ternaria y el tamaño mínimo (anchura o altura) en muestras de luma de un bloque hoja de luma resultante de la división de árbol cuádruple de una CTU en el corte actual. El valor de slice log2_diff_max_tt_min_qt_luma deberá estar en el intervalo de 0 a CtbLog2SizeY - MinQtLog2SizeY, inclusive. Cuando no está presente, el valor de slice_log2_diff_max_tt_min_qt_luma se infiere como sigue:*

- *Si slice_type igual a 2 (I), se infiere que el valor de slice_log2_diff_max_tt_min_qt_luma es igual a sps_log2_diff_max_tt_min_qt_intra_slice_luma*
- *De lo contrario (slice_type igual a 0 (B) o 1 (P)), se infiere que el valor de slice_log2_diff_max_tt_min_qt_luma es igual a sps_log2_diff_max_tt_min_qt_inter_slice.*

slice_log2_diff_min_qt_min_cb_chroma *especifica la diferencia entre el logaritmo de base 2 del tamaño mínimo en muestras de luma de un bloque hoja de croma resultante de la división de árbol cuádruple de una CTU de croma con treeType igual a DUAL_TREE_CHROMA y el logaritmo de base 2 del tamaño de bloque de codificación mínimo en muestras de luma para CU de croma con treeType igual a DUAL_TREE_CHROMA en el corte actual. El valor de slice log2_diff_min_qt_min_cb_chroma deberá estar en el intervalo de 0 a CtbLog2SizeY - MinCbLog2SizeY, inclusive. Cuando no está presente, se infiere que el valor de slice_log2_diff_min_qt_min_cb_chroma es igual a sps log2_diff_min_qt_min_cb_intra_slice_chroma.*

slice_max_mtt_hierarchy_depth_chroma *especifica la profundidad de jerarquía máxima para unidades de codificación resultantes de la división de árbol de múltiples tipos de una hoja de árbol cuádruple con tipo de árbol igual a DUAL_TREE_CHROMA en el corte actual. El valor de slice_max_mtt_hierarchy_depth_chroma deberá estar en el intervalo de 0 a CtbLog2SizeY - MinCbLog2SizeY, inclusive. Cuando no está presente, se infiere que los valores de slice_max_mtt_hierarchy_depth_chroma son iguales a sps_max_mtt_hierarchy_depth_intra_slices_chroma.*

slice_log2_diff_max_bt_min_qt_chroma *especifica la diferencia entre el logaritmo de base 2 del tamaño máximo (anchura o altura) en muestras de luma de un bloque de codificación de croma que se puede dividir usando una división binaria y el tamaño mínimo (anchura o altura) en muestras de luma de un bloque hoja de croma resultante de la división de árbol cuádruple de una CTU de croma con tipo de árbol igual a DUAL_TREE_CHROMA en el corte actual. El valor de slice_log2_diff_max_bt_min_qt_chroma deberá estar en el intervalo de 0 a CtbLog2SizeY - MinQtLog2SizeC, inclusive. Cuando no está presente, se infiere que el valor de corte log2_diff_max_bt_min_qt_chroma es igual a sps log2_diff_max_bt_min_qt_intra_slice_chroma*

slice_log2_diff_max_tt_min_qt_chroma *especifica la diferencia entre el logaritmo de base 2 del tamaño máximo*

(anchura o altura) en muestras de luma de un bloque de codificación de croma que se puede dividir usando una división ternaria y el tamaño mínimo (anchura o altura) en muestras de luma de un bloque hoja de croma resultante de la división de árbol cuádruple de una CTU de croma con tipo de árbol igual a DUAL_TREE_CHROMA en el corte actual. El valor de slice_log2_diff_max_tt_min_qt_chroma deberá estar en el intervalo de 0 a CtbLog2SizeY - MinQtLog2SizeC, inclusive. Cuando no está presente, se infiere que el valor de slice_log2_diff_max_tt_min_qt_chroma es igual a sps_log2_diff_max_tt_min_qt_intra_slice_chroma

Las variables MinQtLog2SizeY, MinQtLog2SizeC, MinQtSizeY, MinQtSizeC, MaxBtSizeY, MaxBtSizeC, MinBtSizeY, MaxTtSizeY, MaxTtSizeC, MinTtSizeY, MaxMttDepthY y MaxMttDepthC se derivan como sigue:

$$MinQtLog2SizeY = MinCbLog2SizeY + slice\_log2\_diff\_min\_qt\_min\_cb\_luma \quad (7\text{-}86)$$

$$MinQtLog2SizeC = MinCbLog2SizeY + slice\_log2\_diff\_min\_qt\_min\_cb\_chroma \quad (7\text{-}87)$$

$$MinQtSizeY = 1 << MinQtLog2SizeY \quad (7\text{-}88)$$

$$MinQtSizeC = 1 << MinQtLog2SizeC \quad (7\text{-}89)$$

$$MaxBtSizeY = 1 << (MinQtLog2SizeY + slice\_log2\_diff\_max\_bt\_min\_qt\_luma) \quad (7\text{-}90)$$

$$MaxBtSizeC = 1 << (MinQtLog2SizeC + slice\_log2\_diff\_max\_bt\_min\_qt\_chroma) \quad (7\text{-}91)$$

$$MinBtSizeY = 1 << MinCbLog2SizeY \quad (7\text{-}92)$$

$$MaxTtSizeY = 1 << (MinQtLog2SizeY + slice\_log2\_diff\_max\_tt\_min\_qt\_luma) \quad (7\text{-}93)$$

$$MaxTtSizeC = 1 << (MinQtLog2SizeC + slice\_log2\_diff\_max\_tt\_min\_qt\_chroma) \quad (7\text{-}94)$$

$$MinTtSizeY = 1 << MinCbLog2SizeY \quad (7\text{-}95)$$

$$MaxMttDepthY = slice\_max\_mtt\_hierarchy\_depth\_luma \quad (7\text{-}96)$$

$$MaxMttDepthC = slice\_max\_mtt\_hierarchy\_depth\_chroma \quad (7\text{-}97)$$

2.2 Tamaño de transformada máximo en VVC

En el Borrador 5 de VVC, el tamaño de transformada máximo se señaliza en el SPS, pero se fija como de longitud de 64 y no es configurable. Sin embargo, en la reunión de JVET de julio de 2019, se decidió habilitar que el tamaño de transformada de luma máxima fuera 64 o 32 únicamente con una bandera en el nivel de SPS. El tamaño de transformada de croma máximo se deriva de la relación de muestreo de croma en relación con el tamaño de transformada de luma máximo.

A continuación, se encuentran las modificaciones de especificación correspondientes en el borrador 6 de VVC con la adopción de JVET-O05xxx.

7.3.2.3. Sintaxis de RBSP de conjunto de parámetros de secuencia

seq_parameter_set_rbsp() {	***Descriptor***
...	
sps_max_luma_transform_size_64_flag	*u(1)*
...	

7.4.3.3. Semántica de RBSP de conjunto de parámetros de secuencia

... sps_max_luma_transform_size_64_flag igual a 1 especifica que el tamaño de transformada máximo en muestras de luma es igual a 64. sps_max_luma_transform_size_64_flag igual a 0 especifica que el tamaño de transformada máximo en muestras de luma es igual a 32.

Cuando CtbSizeY es menor que 64, el valor de la bandera sps_max_luma_transform_size_64 deberá ser igual a 0. *Las variables MinTbLog2SizeY, MaxTbLog2SizeY, MinTbSizeY y MaxTbSizeY se derivan como sigue:*

MinTbLog2SizeY = 2 (7-27)

MaxTbLog2SizeY = sps_max_luma_transform_size_64_flag ? 6 : 5 (7-28)

MinTbSizeY = 1 << MinTbLog2SizeY (7-29)

MaxTbSizeY = 1 << MaxTbLog2SizeY (7-30)

sps_sbt_max_size_64_flag *igual a 0 especifica que la anchura y altura de CU máximas para permitir la transformada de subbloque es de 32 muestras de luma. sps_sbt_max_size_64_flag igual a 1 especifica que la anchura y altura de CU máximas para permitir la transformada de subbloque es 64 muestras de luma.*

MaxSbtSize = Min(MaxTbSizeY, sps_sbt_max_size_64_flag ? 64 : 32) (7-31)

...

3. Ejemplos de problemas técnicos abordados por las soluciones técnicas divulgadas

Hay varios problemas en el último borrador de trabajo de VVC JVET-O2001-v11, que se describen a continuación.

1) En el borrador 6 de VVC actual, el tamaño de transformada máximo y el tamaño de CTU se definen independientemente. Por ejemplo, el tamaño de la CTU podría ser 32, mientras que, el tamaño de transformada podría ser 64. Es deseable que el tamaño de transformada máximo sea igual o menor que el tamaño de CTU.
2) En el borrador 6 de VVC actual, el proceso de partición de bloques depende del tamaño de bloque de transformada máximo distinto del tamaño de VPDU. Por lo tanto, si el tamaño de bloque de transformada máximo es 32x32, además de prohibir la división de TT de 128x128 y la división de BT vertical de 64x128, y la división de BT horizontal de 128x64 para obedecer la regla de VPDU, además prohíbe la división de TT para el bloque de 64x64, prohíbe la división de BT vertical para el bloque de codificación de 32x64/16x64/8x64, y también prohíbe la división de BT horizontal para el bloque de codificación de 64x8/64x16/64x32, lo que puede no ser eficiente para la eficiencia de codificación.
3) El borrador 6 de VVC actual permite un tamaño de CTU igual a 32, 64 y 128. Sin embargo, es posible que el tamaño de la CTU pueda ser mayor que 128. Por lo tanto, necesitan modificarse algunos elementos de sintaxis.

a) Si se permite un tamaño de CTU mayor, la estructura de partición de bloque y la señalización de banderas de división de bloque pueden rediseñarse.
b) Si se permite un tamaño de CTU mayor, entonces puede rediseñarse parte del diseño actual (por ejemplo, derivación de parámetros afines, predicción de IBC, tamaño de memoria intermedia de IBC, predicción de triángulo de fusión, CIIP, modo de fusión normal, etc.).

4) En el borrador 6 de VVC actual, el tamaño de CTU se señaliza a nivel de SPS. Sin embargo, dado que la adopción del remuestreo de imágenes de referencia (también conocido como cambio de resolución adaptativo) permite que las imágenes puedan codificarse con diferentes resoluciones en un flujo de bits, el tamaño de la CTU puede ser diferente a través de múltiples capas.
5) En WD6, el tamaño de bloque máximo usado para MIP e ISP depende del tamaño de transformada máximo, distinto del tamaño de VPDU o 64x64, lo que puede no ser eficiente para la eficiencia de codificación.
6) En WD6, el tamaño de bloque máximo usado para omisión de transformada e intra BDPCM depende del tamaño de omisión de transformada máximo, que está restringido por el tamaño de transformada máximo.
7) En WD6, el tamaño de bloque máximo usado para SBT depende del tamaño de SBT máximo, que está restringido por el tamaño de transformada máximo.
8) En WD6, el tamaño de bloque de codificación usado para IBC y PLT está limitado a 64x64, que puede ajustarse por el tamaño de transformada máximo, tamaño de CTU y/o tamaño de VPDU.
9) En WD6, el tamaño de bloque de codificación usado para CIIP podría ser mayor que el tamaño de transformada máximo.
10) En WD6, la bandera de LMCS habilitado no está condicionada por la bandera de omisión de cuantificación de transformada.

4. Realizaciones y técnicas de ejemplo

El listado de soluciones a continuación debe considerarse como ejemplos para explicar algunos conceptos. Estos elementos no deben interpretarse de manera restringida. Además, estos elementos se pueden combinar de cualquier manera.

En este documento, C=mín(a,b) indica que la C es igual al valor mínimo entre a y b.

En este documento, el tamaño/dimensión de la unidad de vídeo puede ser la altura o la anchura de una unidad de vídeo (por ejemplo, anchura o altura de una imagen/subimagen/corte/ladrillo/mosaico/CTU/CU/CB/TU/TB). Si un

tamaño de unidad de vídeo se indica por MxN, entonces M indica la anchura y N indica la altura de la unidad de vídeo.

En este documento, "un bloque de codificación" puede ser un bloque de codificación de luma y/o un bloque de codificación de croma. El tamaño/dimensión en muestras de luma para un bloque de codificación puede usarse en esta invención para representar el tamaño/dimensión medido en muestras de luma. Por ejemplo, un bloque de codificación de 128x128 (o un tamaño de bloque de codificación de 128x128 en muestras de luma) puede indicar un bloque de codificación de luma de 128x128 y/o un bloque de codificación de croma de 64x64 para formato de color 4:2:0. De manera similar, para formato de color 4:2:2, puede referirse a un bloque de codificación de luma de 128x128 y/o a un bloque de codificación de croma de 64x128. Para formato de color 4:4:4, puede referirse a un bloque de codificación de luma de 128x128 y/o a un bloque de codificación de croma de 128x128.

Tamaño de CTU configurable relacionado

1. Se propone que pueden permitirse diferentes dimensiones de CTU (tales como anchura y/o altura) para diferentes unidades de vídeo tales como capas/imágenes/subimágenes/cortes/mosaicos/ladrillos.

a) En un ejemplo, uno o múltiples conjuntos de dimensiones de CTU pueden señalizarse explícitamente a un nivel de unidad de vídeo tal como nivel de VPS/DPS/SPS/PPS/APS/imagen/subimagen/corte/cabecera de corte/mosaico/ladrillo.
b) En un ejemplo, cuando se permite el remuestreo de imágenes de referencia (también conocido como cambio de resolución adaptativa), las dimensiones de la CTU pueden ser diferentes a través de las capas de diferencia.

i. Por ejemplo, las dimensiones de CTU de una imagen inter-capa pueden derivarse implícitamente de acuerdo con el factor de escalado de submuestreo/sobremuestreo.

1. Por ejemplo, si las dimensiones de CTU señalizadas para una capa base son M × N (tal como M = 128 y N = 128) y la imagen codificada inter-capa se remuestrea por un factor de escala S en anchura y un factor de escala T en altura, que puede ser mayor o menor que 1 (tal como S = 1/4 y T = 1/2 que indica que la imagen codificada inter-capa se submuestrea 4 veces en anchura y se submuestrea 2 veces en altura), entonces, las dimensiones de CTU en la imagen codificada inter-capa pueden derivarse a (M×S) × (N×T), o (M/S) × (N/T).

ii. Por ejemplo, pueden señalizarse explícitamente diferentes dimensiones de CTU para múltiples capas a nivel de unidad de vídeo, por ejemplo, para imágenes/subimágenes de remuestreo inter-capa, las dimensiones de CTU pueden señalizarse a nivel de VPS/DPS/SPS/PPS/APS/imagen/subimagen/corte/cabecera de corte/mosaico/ladrillo que es diferente del tamaño de CTU de capa base.

2. Se propone que, si se permite o no la división de TT o BT puede depender de las dimensiones de VPDU (tales como anchura y/o altura). Supóngase que, la VPDU tiene una dimensión VSize en muestras de luma, y el bloque de árbol de codificación tiene una dimensión CtbSizeY en muestras de luma.

a) En un ejemplo, VSize = mín(M, CtbSizeY). M es un valor entero tal como 64.
b) En un ejemplo, si se permite o no la división de TT o BT puede ser independiente del tamaño de transformada máximo.
c) En un ejemplo, la división de TT puede deshabilitarse cuando una anchura o altura de bloque de codificación en muestras de luma es mayor que mín(VSize, maxTtSize).

i. En un ejemplo, cuando el tamaño de transformada máximo es igual a 32x32, pero VSize es igual a 64x64, la división de TT puede deshabilitarse para el bloque de codificación de 128x128/128x64/64x128.
ii. En un ejemplo, cuando el tamaño de transformada máximo es igual a 32x32, pero VSize es igual a 64x64, Puede permitirse una división de TT para un bloque de codificación de 64x64.

d) En un ejemplo, la división de BT vertical puede deshabilitarse cuando una anchura de bloque de codificación en muestras de luma es menor o igual que VSize, pero su altura en muestras de luma es mayor que VSize.

i. En un ejemplo, en caso de tamaño de transformada máximo de 32x32, pero tamaño de VPDU igual a 64x64, la división de BT vertical puede deshabilitarse para el bloque de codificación de 64x128.
ii. En un ejemplo, en caso de tamaño de transformada máximo de 32x32, pero tamaño de VPDU igual a 64x64, se puede permitir una división de BT vertical para un bloque de codificación de 32x64/16x64/8x64.

e) En un ejemplo, la división de BT vertical puede deshabilitarse cuando un bloque de codificación supera la anchura de imagen/subimagen en muestras de luma, pero su altura en muestras de luma es mayor que VSize.

i. Como alternativa, la división de BT horizontal puede permitirse cuando un bloque de codificación supera la anchura de imagen/subimagen en muestras de luma.

f) En un ejemplo, la división de BT horizontal puede deshabilitarse cuando una anchura de bloque de codificación en muestras de luma es mayor que VSize, pero su altura en muestras de luma es menor o igual que VSize.

i. En un ejemplo, en caso de tamaño de transformada máximo de 32x32, pero tamaño de VPDU igual a 64x64, la división de BT vertical puede deshabilitarse para el bloque de codificación de 128x64.
ii. En un ejemplo, en caso de tamaño de transformada máximo de 32x32, pero tamaño de VPDU igual a 64x64, se puede permitir una división de BT horizontal para un bloque de codificación de 64x8/64x16/64x32.

g) En un ejemplo, la división de BT horizontal puede deshabilitarse cuando un bloque de codificación supera la altura de imagen/subimagen en muestras de luma, pero su anchura en muestras de luma es mayor que VSize.

i. Como alternativa, la división de BT vertical puede permitirse cuando un bloque de codificación supera la altura de imagen/subimagen en muestras de luma.

h) En un ejemplo, cuando la división TT o BT está deshabilitada, la bandera de división de TT o BT puede no señalizarse y derivarse implícitamente para que sea cero.

i. Como alternativa, cuando la división de TT y/o BT está habilitada, la bandera de división de TT y/o BT puede señalizarse explícitamente en el flujo de bits.
ii. Como alternativa, cuando la división TT o BT está deshabilitada, la bandera de división de TT o BT puede señalizarse, pero ignorarse por el descodificador.
iii. Como alternativa, cuando la división TT o BT está deshabilitada, la bandera de división de TT o BT puede señalizarse, pero debe ser cero en un flujo de bits de conformidad.

3. Se propone que las dimensiones de la CTU (tales como anchura y/o altura) puedan ser mayores que 128.

a) En un ejemplo, las dimensiones de la CTU señalizadas pueden ser 256 o incluso mayores (por ejemplo, **log2_ctu_size_minus5** puede ser igual a 3 o mayor).
b) En un ejemplo, las dimensiones de CTU derivadas pueden ser 256 o incluso mayores.

i. Por ejemplo, las dimensiones de CTU derivadas para remuestrear imágenes/subimágenes pueden ser mayores que 128.

4. Se propone que cuando se permiten dimensiones de CTU más grandes (tal como la anchura y/o altura de la CTU es mayor que 128), entonces se puede inferir que la bandera de división de QT es verdadera y la división de QT se puede aplicar de manera recursiva hasta que la dimensión del bloque de codificación de división alcance un valor especificado (por ejemplo, un valor especificado puede establecerse al tamaño de bloque de transformada máximo, o 128, o 64, o 32).

a) En un ejemplo, la división de QT recursiva puede realizarse implícitamente sin señalización, hasta que el tamaño de bloque de codificación dividido alcance el tamaño de bloque de transformada máximo.
b) En un ejemplo, cuando la CTU 256x256 se aplica al árbol dual, entonces la bandera de división de QT puede no señalizarse para un bloque de codificación mayor que el tamaño de bloque de transformada máximo, y la división de QT puede forzarse a usarse para el bloque de codificación hasta que el tamaño de bloque de codificación de división alcance el tamaño de bloque de transformada máximo.

5. Se propone que la bandera de división de TT puede señalizarse condicionalmente para dimensiones de CU/PU (anchura y/o altura) mayores que 128.

a) En un ejemplo, tanto las banderas de división de TT horizontal como vertical pueden señalizarse para una CU de 256x256.
b) En un ejemplo, puede señalizarse una división de TT vertical, pero no una división de TT horizontal para una CU/PU de 256x128/256x64.
c) En un ejemplo, puede señalizarse una división de TT horizontal pero no una división de TT vertical para una CU/PU de 128x256/64x256.
d) En un ejemplo, cuando la bandera de división de TT está prohibida para dimensiones de CU mayores que 128, entonces puede no señalizarse y derivarse implícitamente como cero.

i. En un ejemplo, la división de TT horizontal puede prohibirse para CU/PU de 256x128/256x64.
ii. En un ejemplo, la división de TT vertical puede prohibirse para CU/PU de 128x256/64x256.

6. Se propone que la bandera de división de BT puede señalizarse condicionalmente para dimensiones de CU/PU (anchura y/o altura) mayores que 128.

a) En un ejemplo, tanto las banderas de división de BT horizontal como vertical pueden señalizarse para CU/PU de 256x256/256x128/128x256.
b) En un ejemplo, la bandera de división de BT horizontal puede señalizarse para CU/PU de 64x256.
c) En un ejemplo, la bandera de división de BT vertical puede señalizarse para CU/PU de 256x64.
d) En un ejemplo, cuando la bandera de división de BT está prohibida para dimensión de CU mayor que 128, entonces puede no señalizarse y derivarse implícitamente como cero.

i. En un ejemplo, la división de BT vertical puede prohibirse para CU/PU de Kx256 (tal como K es igual o menor que 64 en muestras de luma), y la bandera de división de BT vertical puede no señalizarse y derivarse como cero.

1. Por ejemplo, en el caso anterior, la división de BT vertical puede prohibirse para CU/PU de 64x256.
2. Por ejemplo, en el caso anterior, la división de BT vertical puede prohibirse para evitar 32x256 CU/PU en límites de imagen/subimagen.

ii. En un ejemplo, la división de BT vertical puede prohibirse cuando un bloque de codificación supera la anchura de imagen/subimagen en muestras de luma, pero su altura en muestras de luma es mayor que M (tal como M=64 en muestras de luma).
iii. En un ejemplo, la división de BT horizontal puede prohibirse para el bloque de codificación de 256xK (tal como K es igual o menor que 64 en muestras de luma), y la bandera de división de BT horizontal puede no señalizarse y derivarse como cero.

1. Por ejemplo, en el caso anterior, la división de BT horizontal puede prohibirse para el bloque de codificación de 256x64.
2. Por ejemplo, en el caso anterior, la división de BT horizontal puede prohibirse para evitar el bloque de codificación de 256x32 en los límites de imagen/subimagen.

iv. En un ejemplo, la división de BT horizontal puede prohibirse cuando un bloque de codificación supera la altura de imagen/subimagen en muestras de luma, pero su anchura en muestras de luma es mayor que M (tal como M=64 en muestras de luma).

7. Se propone que el cálculo de parámetros de modelo afín puede depender de las dimensiones de CTU.

a) En un ejemplo, la derivación de vectores de movimiento escalados y/o vectores de movimiento de punto de control en predicción afín puede depender de las dimensiones de CTU.

8. Se propone que la memoria intermedia de copia intra bloque (IBC) puede depender de las dimensiones máximas de CTU configurables/permitidas.

a) Por ejemplo, la anchura de memoria intermedia de IBC en muestras de luma puede ser igual a NxN dividido por la anchura (o altura) de CTU en muestras de luma, en donde N puede ser el tamaño máximo de CTU configurable en muestras de luma, tal como N = 1 << (**log2_ctu_size_minus5** + 5).

9. Se propone que un conjunto de herramienta o herramientas de codificación especificadas pueda deshabilitarse para una CU/PU grande, donde la CU/PU grande se refiere a una CU/PU donde la anchura de CU/PU o la altura de CU/PU es mayor que N (tal como N=64 o 128).

a) En un ejemplo, la herramienta o herramientas de codificación especificadas anteriormente mencionadas pueden ser paleta, y/o copia de intra bloque (IBC), y/o modo de intra omisión, y/o modo de predicción de triángulo, y/o modo de CIIP, y/o modo de fusión normal, y/o derivación de movimiento del lado del descodificador, y/o flujo óptico bidireccional, y/o flujo óptico basado en perfeccionamiento de predicción, y/o predicción afín, y/o TMVP basado en subbloque, y etc.

i. Como alternativa, la herramienta o herramientas de codificación de contenido de pantalla tales como paleta y/o modo de copia intra bloque (IBC) pueden aplicarse a CU/PU grandes.

b) En un ejemplo, pueden usar explícitamente restricción de sintaxis para deshabilitar la herramienta o herramientas de codificación especificadas para una CU/PU grande.

i. Por ejemplo, La bandera de paleta/IBC puede señalizar explícitamente una CU/PU que no es una CU/PU grande.

c) En un ejemplo, puede usar restricción de flujo de bits para deshabilitar la herramienta o herramientas de codificación especificadas para una CU/PU grande.

10. Si se permite o no la división de TT o BT puede depender de las dimensiones de bloque.

a) En un ejemplo, La división de TT puede deshabilitarse cuando una anchura o altura de bloque de codificación en muestras de luma es mayor que N (por ejemplo, N=64).

i. En un ejemplo, cuando el tamaño de transformada máximo es igual a 32x32, la división de TT puede deshabilitarse para el bloque de codificación de 128x128/128x64/64x128.
ii. En un ejemplo, cuando el tamaño de transformada máximo es igual a 32x32, Puede permitirse una división de TT para un bloque de codificación de 64x64.

b) En un ejemplo, la división de BT vertical puede deshabilitarse cuando una anchura de bloque de codificación en muestras de luma es menor o igual que N (por ejemplo, N=64), pero su altura en muestras de luma es mayor que N (por ejemplo, N=64).

i. En un ejemplo, en caso de tamaño de transformada máximo 32x32, la división de BT vertical puede deshabilitarse para el bloque de codificación de 64x128.
ii. En un ejemplo, en caso de tamaño de transformada máximo 32x32, se puede permitir una división de BT vertical para un bloque de codificación de 32x64/16x64/8x64.

c) En un ejemplo, la división de BT vertical puede deshabilitarse cuando un bloque de codificación supera la anchura de imagen/subimagen en muestras de luma, pero su altura en muestras de luma es mayor que 64.

i. Como alternativa, la división de BT horizontal puede permitirse cuando un bloque de codificación supera la anchura de imagen/subimagen en muestras de luma.

d) En un ejemplo, la división de BT horizontal puede deshabilitarse cuando una anchura de bloque de codificación en muestras de luma es mayor que N (por ejemplo, N=64), pero su altura en muestras de luma es menor o igual que N (por ejemplo, N=64).

i. En un ejemplo, en caso de tamaño de transformada máximo 32x32, la división de BT vertical puede deshabilitarse para el bloque de codificación de 128x64.
ii. En un ejemplo, en caso de tamaño de transformada máximo 32x32, se puede permitir una división de BT horizontal para un bloque de codificación de 64x8/64x16/64x32.

e) En un ejemplo, la división de BT horizontal puede deshabilitarse cuando un bloque de codificación supera la altura de imagen/subimagen en muestras de luma, pero su anchura en muestras de luma es mayor que N (por ejemplo, N=64).

i. Como alternativa, la división de BT vertical puede permitirse cuando un bloque de codificación supera la altura de imagen/subimagen en muestras de luma.

Tamaño de transformada máximo configurable relacionado

11. Se propone que el tamaño máximo de TU pueda depender de las dimensiones de CTU (anchura y/o altura), o las dimensiones de CTU puedan depender del tamaño máximo de TU

a) En un ejemplo, puede usarse una restricción de flujo de bits de que el tamaño máximo de TU deberá ser menor o igual que las dimensiones de CTU.
b) En un ejemplo, la señalización del tamaño máximo de TU puede depender de las dimensiones de CTU.

i. Por ejemplo, cuando las dimensiones de CTU son menores que N (por ejemplo, N=64), el tamaño de TU máximo señalizado debe ser menor que N.
ii. Por ejemplo, cuando las dimensiones de CTU son menores que N (por ejemplo, N=64), la indicación de si el tamaño de transformada de luma máximo es 64 o 32 (por ejemplo, **sps_max_luma_transform_size_64_flag**) puede no señalizarse y el tamaño de transformada de luma máximo puede derivarse como 32 implícitamente.

12. Una cierta herramienta de codificación puede habilitarse para un bloque con anchura y/o altura mayor que el tamaño de bloque de transformada.

a) En un ejemplo, la cierta herramienta de codificación puede ser la predicción de intra subpartición (ISP), MIP, SBT u otras herramientas de codificación que pueden dividir una CU en múltiples TU o un CB en múltiples TB.
b) En un ejemplo, la cierta herramienta de codificación puede ser una herramienta de codificación que no aplica transformada (o únicamente se aplica transformada de identidad), tal como el modo de omisión de transformada, BDPCM/DPCM/PCM.
c) La cierta herramienta puede ser copia de intra bloque (IBC), paleta (PLT).
d) La cierta herramienta puede ser inter intra predicción combinada (CIIP).

13. Si una cierta herramienta de codificación está habilitada o no puede depender de las dimensiones del bloque de codificación.

a) En un ejemplo, la cierta herramienta de codificación puede ser la predicción de intra subpartición (ISP), intra predicción basada en matrices (MIP), transformada de subbloque (SBT), copia de intra bloque (IBC), paleta (PLT), etc.
b) En un ejemplo, la cierta herramienta de codificación (tal como ISP, MIP) puede permitirse cuando una anchura y/o altura de bloque de codificación en muestras de luma son menores o iguales que N (por ejemplo, N=64).

i. Como alternativa, la cierta herramienta de codificación puede deshabilitarse cuando una anchura y/o altura de bloque de codificación en muestras de luma es mayor que N (por ejemplo, N=64).

c) Si la cierta herramienta de codificación (tal como ISP, MIP) está habilitada o no puede depender de la relación entre el tamaño de bloque de codificación y el tamaño de VPDU.

i. En un ejemplo, la cierta herramienta de codificación puede permitirse cuando una anchura y/o altura de bloque de codificación en muestras de luma son menores o iguales que el tamaño de VPDU (tal como 32 o 64).

1. Como alternativa, la cierta herramienta de codificación puede deshabilitarse cuando una anchura y/o altura de bloque de codificación en muestras de luma es mayor que el tamaño de VPDU (tal como 32 o 64).

d) Si la predicción de intra subpartición (ISP) está habilitada o no y/o qué tipo o tipos de partición (por ejemplo, dirección de división) está(n) permitido(s) puede(n) depender de la relación entre las dimensiones de la subpartición y el tamaño de bloque de transformada máximo.

i. En un ejemplo, si la anchura y/o altura de subpartición no es mayor que el tamaño de bloque de transformada máximo para al menos un tipo de partición, ISP puede estar habilitada.

1. Como alternativa, además, de lo contrario, IPS puede estar deshabilitada.

ii. En un ejemplo, si la anchura y/o altura de subpartición no son mayores que el tamaño de bloque de transformada máximo para todos los tipos de partición permitidos, ISP puede estar habilitada.

1. Como alternativa, además, de lo contrario, ISP puede estar deshabilitada.

iii. En un ejemplo, la señalización del tipo de partición (por ejemplo, **intra_subpartitions_split_flag**) puede depender de la relación entre la anchura y/o altura de la subpartición correspondiente basándose en el tipo de partición y el tamaño de bloque de transformada máximo.

1. En un ejemplo, si únicamente un tipo de partición satisface la condición de que la anchura y/o altura de la subpartición correspondiente basándose en el tipo de partición no es mayor que el tamaño de bloque de transformada máximo, el tipo de partición puede no señalizarse e inferirse.

e) Si la cierta herramienta de codificación (tal como IBC, PLT) está habilitada o no puede depender de la relación entre el tamaño de bloque de codificación y el tamaño de transformada máximo (tal como 32 o 64).

i. En un ejemplo, si la cierta herramienta de codificación (tal como IBC, PLT) está habilitada o no puede NO estar condicionada a la relación entre la dimensión de bloque de codificación y un número fijo 64.
ii. En un ejemplo, la cierta herramienta de codificación (tal como IBC, PLT) puede permitirse cuando una anchura y altura de bloque de codificación en muestras de luma NO son mayores que el tamaño de transformada máximo.

1. En un ejemplo, la cierta herramienta de codificación (tal como IBC, PLT) puede deshabilitarse cuando la anchura y/o altura de bloque en muestras de luma es mayor que el tamaño de transformada máximo.
2. En un ejemplo, cuando el tamaño de transformada máximo es igual a 32, la cierta herramienta de codificación (tal como IBC, PLT) puede deshabilitarse cuando la anchura y/o altura de bloque en muestras de luma es igual a 64.

f) Si cierta herramienta de codificación está deshabilitada, los elementos de sintaxis relacionados (tales como **intra_subpartitions_mode_flag** e **intra_subpartitions_split_flag** para ISP, **intra_mip_flag** e **intra_mip_mode** para MIP, **pred_mode_ibc_flag** para IBC, **pred_mode_plt_flag** para PLT) pueden no señalizarse e inferirse que son 0.

g) Si cierta herramienta de codificación está deshabilitada, los elementos de sintaxis relacionados (tales como **intra_subpartitions_mode_flag** e **intra_subpartitions_split_flag** para ISP, **intra_mip_flag** e **intra_mip_mode** para MIP, **pred_mode_ibc_flag** para IBC, **pred_mode_plt_flag** para PLT) puede señalizarse, pero debe ser 0 en un flujo de bits de conformidad.

14. La división de QT implícita puede depender del tamaño de VPDU y/o tamaño de transformada máximo.

a) En un ejemplo, la división de QT implícita puede NO estar condicionada a la relación entre la dimensión de bloque de codificación y un número fijo 64.
b) En un ejemplo, un bloque de codificación puede dividirse implícitamente en particiones cuádruples, y cada subpartición puede dividirse implícitamente de manera recursiva hasta que tanto la anchura como la altura de la subpartición alcancen el tamaño de VPDU.
c) En un ejemplo, un bloque de codificación puede dividirse implícitamente en particiones cuádruples, y cada subpartición puede dividirse implícitamente de manera recursiva hasta que tanto la anchura como la altura de la subpartición alcancen el tamaño de transformada máximo.

15. La anchura y/o altura de bloque máximas usadas para la transformada de subbloque (SBT) puede depender del tamaño de transformada máximo.

a) En un ejemplo, el tamaño de SBT máximo puede establecerse igual al tamaño de transformada máximo.
b) En un ejemplo, el elemento de sintaxis (tal como **sps_sbt_max_size_64_flag**) relacionado con el tamaño de SBT máximo puede no señalizarse.

i. Por ejemplo, **sps_sbt_max_size_64_flag** no se señaliza y se infiere que es 0 cuando el tamaño de transformada máximo es menor que 64.
ii. Por ejemplo, **sps_sbt_max_size_64_flag** se señaliza cuando el tamaño de transformada máximo es menor que 64, pero debe ser igual a 0 en un flujo de bits de conformidad.

c) En un ejemplo, la señalización del elemento de sintaxis relacionado (tal como **cu_sbt_flag**) puede depender del tamaño de transformada máximo.
d) En un ejemplo, la señalización del elemento de sintaxis relacionado (tal como **cu_sbt_flag**) puede ser independiente del tamaño de SBT máximo.

16. El tamaño de bloque máximo usado para omisión de transformada y/o intra BDPCM puede depender del tamaño de transformada máximo.

a) En un ejemplo, el tamaño de omisión de transformada máximo puede establecerse igual al tamaño de transformada máximo.
b) En un ejemplo, el elemento de sintaxis (tal como **log2 transform skip_max size_minus2)** relacionado con el tamaño de omisión de transformada máximo puede no señalizarse.

17. El tamaño de bloque máximo usado para intra BDPCM puede señalizarse independientemente.

a) En un ejemplo, el tamaño de bloque máximo usado para intra BDPCM puede no depender del tamaño de bloque máximo usado para la omisión de transformada.
b) En un ejemplo, puede señalizarse una bandera de nivel de SPS/VPS/PPS/corte/VPDU/CTU/CU en el flujo de bits para especificar el tamaño de bloque máximo usado para intra BDPCM.

18. Si habilitar o deshabilitar inter intra predicción combinada (CIIP) para un bloque puede depender de la relación entre la anchura y/o altura de bloque y el tamaño de transformada máximo.

a) En un ejemplo, CIIP puede deshabilitarse para un bloque si una anchura y/o altura de bloque es mayor que el tamaño de transformada máximo.
b) En un ejemplo, el elemento de sintaxis para indicar CIIP (tal como un **ciip_flag**) puede no señalizarse si CIIP está deshabilitado.

19. Cuando tanto la anchura como la altura de una CU codificada con CIIP son menores que 128, no está permitido dividir una CU en varias subparticiones, en donde la intra predicción para una primera subpartición puede depender de la reconstrucción de una segunda subpartición, en la que se realiza la intra predicción antes que en la primera subpartición.

20. Se permite dividir una CU codificada con CIIP en varias subparticiones, en donde la intra predicción para una primera subpartición puede depender de la reconstrucción de una segunda subpartición, en la que se realiza la intra predicción antes que en la primera subpartición.

Codificación sin pérdidas relacionada

21. El tamaño máximo para bloques codificados de omisión de transformada (es decir, no se aplica transformada/únicamente se aplica transformada de identidad) se deriva del tamaño máximo para bloques de transformada aplicada (por ejemplo, MaxTbSizeY).

a) En un ejemplo, El tamaño máximo para bloques codificados de omisión de transformada se infiere a MaxTbSizeY.
b) En un ejemplo, se omite la señalización de tamaño máximo para bloques codificados de omisión de transformada.

22. Cuando se habilita la codificación sin pérdidas, el escalado de croma de mapeo de luma (LMCS) puede deshabilitarse para la unidad de vídeo actual en nivel de secuencia/imagen/subimagen/corte/grupo de mosaicos/mosaico/ladrillo/fila de CTU/CTU/CU/PU/TU/subbloque.

a) En un ejemplo, la bandera de LMCS habilitado (tal como **sps_lmcs_enabled_flag, slice_lmcs_enabled_flag, slice_chroma_residual_scale_flag, lmcs_data, y etc.**) puede señalizarse mediante el acondicionamiento en la bandera de omisión de cuantificación de transformada (tal como **sps_ transquant_bypass_flag, pps_transquant_bypass_flag, cu_ transquant_bypass_flag,**) en nivel de secuencia/imagen/subimagen/corte/grupo de mosaicos/mosaico/ladrillo/fila de CTU/CTU/CU/PU/TU/subbloque.

i. En un ejemplo, si la bandera de omisión de cuantificación de transformada es igual a 1, la bandera de LMCS habilitado puede no señalizarse e inferirse que es 0.

1. En un ejemplo, si la bandera de omisión de cuantificación de transformada de nivel de secuencia (tal como **sps_transquant_bypass_flag**) es igual a 1, la secuencia y la bandera de LMCS habilitado de nivel inferior (tal como **sps_lmcs_enabled_flag, slice_lmcs_enabled_flag, slice_chroma_residual_scale_flag**) puede no señalizarse e inferirse que es 0.
2. En un ejemplo, si el nivel de secuencia TransquantBypassEnabledFlag es igual a 1, el nivel de APS **lmcs_data** puede no estar señalizado.
3. En un ejemplo, si la bandera de omisión de cuantificación de transformada de nivel de PPS (tal como **pps_transquant_bypass_flag**) es igual a 1, la bandera de LMCS habilitado de nivel de corte (tal como **slice_lmcs_enabled_flag, slice_lmcs_aps_id, slice_chroma_residual_scale_flag**) puede no señalizarse e inferirse que es 0.

b) En un ejemplo, puede aplicarse una restricción de flujo de bits de que la bandera de LMCS habilitado debe ser igual a 0 cuando la bandera de omisión de cuantificación de transformada es igual a 1.

5. Realizaciones

Las partes recién añadidas están encerradas entre paréntesis dobles en negrita, por ejemplo, **{{a}}** indica que se ha añadido "a", mientras que, las partes eliminadas del borrador de trabajo de VVC están encerradas entre corchetes dobles en negrita, por ejemplo, **[[b]]** indica que se ha eliminado "b". Las modificaciones se basan en el último borrador de trabajo de VVC (JVET-O2001-v11).

5.1 Una realización de ejemplo n.° 1

La realización a continuación es para el método inventado que hace que el tamaño de TU máximo dependa del tamaño de CTU.

7.4.3.3. Semántica de RBSP de conjunto de parámetros de secuencia

sps_max_luma_transform_size_64_flag *igual a 1 especifica que el tamaño de transformada máximo en muestras de luma es igual a 64. sps_max_luma_transform_size_64_flag igual a 0 especifica que el tamaño de transformada máximo en muestras de luma es igual a 32.*

Cuando CtbSizeY es menor que 64, el valor de sps_max_luma_transform_size_64_flag deberá ser igual a 0.

Las variables MinTbLog2SizeY, MaxTbLog2SizeY, MinTbSizeY, y MaxTbSizeY se derivan como sigue:

$$MinTbLog2SizeY = 2 \quad (7\text{-}27)$$

$$MaxTbLog2SizeY = sps\_max\_luma\_transform\_size\_64\_flag\ ?\ 6\ :\ 5 \quad (7\text{-}28)$$

$$MinTbSizeY = 1\ <<\ MinTbLog2SizeY \quad (7\text{-}29)$$

$$MaxTbSizeY = \{\{min(CtbSizeY,\ 1\ <<\ MaxTbLog2SizeY)\}\} \quad (7\text{-}30)$$

5.2 Una realización de ejemplo n.° 2

La realización a continuación es para el método inventado que hace que el proceso de división de TT y BT dependa del tamaño de VPDU.

6.4.2 Proceso de división binaria permitido

La variable allowBtSplit se deriva como sigue:

- *De lo contrario, si todas las siguientes condiciones son verdaderas, allowBtSplit se establece igual a FALSO*
  - *btSplit es igual a SPLIT BT VER*
  - *cbHeight es mayor que* **[[***MaxTbSizeY***]] {{***VSize***}}**
  - *x0 + cb Width es mayor que pic_width_in_luma_samples*
- *De lo contrario, si todas las siguientes condiciones son verdaderas, allowBtSplit se establece igual a FALSO*
  - *btSplit es igual a SPLIT_BT_HOR*
  - *cbWidth es mayor que* **[[***MaxTbSizeY***]] {{***VSize***}}**
  - *y0 + cbHeight es mayor que pic_height_in_luma_samples*
- *De lo contrario, si todas las siguientes condiciones son verdaderas, allowBtSplit se establece igual a FALSO*
  - *btSplit es igual a SPLIT BT VER*
  - *cbWidth es menor o igual que* **[[***MaxTbSizeY***]] {{***VSize***}}**
  - *cbHeight es mayor que* **[[***MaxTbSizeY***]] {{***VSize*}}
- *De lo contrario, si todas las siguientes condiciones son verdaderas, allowBtSplit se establece igual a FALSO*
  - *btSplit es igual a SPLIT_BT_HOR*
  - *cbWidth es mayor que* **[[***MaxTbSizeY***]] {{***VSize***}}**
  - *cbHeight es menor o igual que* **[[***MaxTbSizeY***]] {{***VSize***}}**

6.4.3 Proceso de división ternaria permitido

La variable allowTtSplit se deriva como sigue:

- *Si una o más de las siguientes condiciones son verdaderas, allowTtSplit se establece igual a FALSO:*
  - *cbSize es menor o igual que 2 * MinTtSizeY*
  - *cbWidth es mayor que Min(* **[[***MaxTbSizeY***]] {{***VSize***}}**, *maxTtSize)*
  - *cbHeight es mayor que Min(* **[[***MaxTbSizeY***]] {{***VSize***}}**, *maxTtSize)*
  - *mttDepth es mayor o igual que maxMttDepth*
  - *x0 + cbWidth es mayor que pic_width_in_luma_samples*
  - *y0 + cbHeight es mayor que pic height_in_luma_samples*
  - *el tipo de árbol es igual a DUAL_TREE_CHROMA y (cbWidth / SubWidthC) * (cbHeight / SubHeightC) es menor o igual que 32*
  - *treeType es igual a DUAL_TREE_CHROMA y modeType es igual a INTRA*
- *De lo contrario, allowTtSplit se establece igual a VERDADERO.*

5.3 Una realización de ejemplo n.° 3

La realización a continuación es para el método inventado que hace que el cálculo de parámetros de modelo afín dependa del tamaño de CTU.

7.4.3.3. Semántica de RBSP de conjunto de parámetros de secuencia

log2_ctu_size_minus5 *plus 5 especifica el tamaño de bloque de árbol de codificación de luma de cada CTU. Es un requisito de conformidad de flujo de bits que el valor de log2_ctu_size_minus5 sea menor o igual que* [[*2*]] {{*3 (podría ser mayor según lo especificado)*}}.

CtbLog2SizeY = log2_ctu_size_minus5 + 5

{{CtbLog2SizeY se usa para indicar el tamaño de CTU en muestras de luma de la unidad de vídeo actual. Cuando se

usa un único tamaño de CTU para la unidad de vídeo actual, el CtbLog2SizeY se calcula mediante la ecuación anterior. De lo contrario, CtbLog2SizeY puede depender del tamaño de CTU real que puede señalizarse explícitamente o derivarse implícitamente para la unidad de vídeo actual. (un ejemplo) }}

8.5.5.5 Proceso de derivación para vectores de movimiento de punto de control afines de luma desde un bloque vecino

Las variables mvScaleHor, mvScaleVer, dHorX y dVerX se derivan como sigue:

- *Si isCTUboundary es igual a VERDADERO, se aplica lo siguiente:*

$$mvScaleHor = MvLX[\ xNb\ ][\ yNb + nNbH - 1\ ][\ 0\ ] << \{\{7\}\}\ \{\{CtbLog2SizeY\}\} \quad (8\text{-}533)$$

$$mvScaleVer = MvLX[\ xNb\ ][\ yNb + nNbH - 1\ ][\ 1\ ] << \{\{7\}\}\ \{\{CtbLog2SizeY\}\} \quad (8\text{-}534)$$

De lo contrario (isCTUboundary es igual a FALSO), se aplica lo siguiente:

$$mvScaleHor = CpMvLX[\ xNb\ ][\ yNb\ ][\ 0\ ][\ 0\ ] << \{\{7\}\}\ \{\{CtbLog2SizeY\}\} \quad (8\text{-}537)$$

$$mvScaleVer = CpMvLX[\ xNb\ ][\ yNb\ ][\ 0\ ][\ 1\ ] << \{\{7\}\}\ \{\{CtbLog2SizeY\}\} \quad (8\text{-}538)$$

8.5.5.6 Proceso de derivación para candidatos de fusión de vector de movimiento de punto de control afines construidos

Cuando availableFlagCorner[0] es igual a VERDADERO y availableFlagcorner[2] es igual a VERDADERO, se aplica lo siguiente:

- *Para X que se reemplaza por 0 o 1, se aplica lo siguiente:*
  - *La variable availableFlagLX se deriva como sigue:*
    - *Si todas las siguientes condiciones son VERDADERAS, availableFlagLX se establece igual a VERDADERO:*
      - *predFlagLXCorner[0] es igual a 1*
      - *predFlagLXCorner[2] es igual a 1*
      - *refIdxLXCorner[0] es igual a refIdxLXCorner[2]*
    - *De lo contrario, availableFlagLX se establece igual a FALSO.*
  - *Cuando availableFlagLX es igual a VERDADERO, se aplica lo siguiente:*
    - *El segundo vector de movimiento de punto de control cpMvLXCorner [1] se deriva como sigue:*

$$\begin{aligned} cpMvLXCorner[\ 1\ ][\ 0\ ] &= (\ cpMvLXCorner[\ 0\ ][\ 0\ ] << \{\{7\}\} \\ &\quad \{\{CtbLog2SizeY\}\}\ ) + \\ &\quad (\ (\ cpMvLXCorner[\ 2\ ][\ 1\ ] - cpMvLXCorner[\ 0\ ][\ 1\ ]\ ) \\ &\quad << (\ \{\{7\}\}\ \{\{CtbLog2SizeY\}\} + Log2(\ cbHeight / cbWidth\ )\ )\ ) \end{aligned} \quad (8\text{-}606)$$

$$\begin{aligned} cpMvLXCorner[\ 1\ ][\ 1\ ] &= (\ cpMvLXCorner[\ 0\ ][\ 1\ ] << \{\{7\}\} \\ &\quad \{\{CtbLog2SizeY\}\}\ ) + \\ &\quad (\ (\ cpMvLXCorner[\ 2\ ][\ 0\ ] - cpMvLXCorner[\ 0\ ][\ 0\ ]\ ) \\ &\quad << (\ \{\{7\}\}\ \{\{CtbLog2SizeY\}\} + Log2(\ cbHeight / cbWidth\ )\ )\ ) \end{aligned} \quad (8\text{-}607)$$

8.5.5.9 Proceso de derivación para matrices de vectores de movimiento a partir de vectores de movimiento de puntos de control afines

Las variables mvScaleHor, mvscale Ver, dHorX y dVerX se derivan como sigue:

$$mvScaleHor = cpMvLX[\,0\,][\,0\,] << [[7]]\ \{\{CtbLog2SizeY\}\} \quad (8\text{-}665)$$

$$mvScaleVer = cpMvLX[\,0\,][\,1\,] << [[7]]\ \{\{CtbLog2SizeY\}\} \quad (8\text{-}666)$$

5.4 Realización n.° 4 sobre permitir división de BT y TT dependiendo del tamaño de bloque

Como se muestra en la figura 1, puede permitirse división de TT para un bloque de codificación con un tamaño de bloque de 64x64, y puede permitirse una división de BT para tamaños de bloque de 32x64, 16x64, 8x64, 64x32, 64x16, 64x8, sin importar que el tamaño de transformada máximo sea 32x32 o 64x64.

La figura 1 es un ejemplo para permitir división de BT y TT dependiendo del tamaño de bloque.

5.5 Realización n.° 5 sobre la aplicación de ISP dependiente del tamaño de VPDU, o 64x64

Las modificaciones se basan en el último borrador de trabajo de VVC (JVET-O2001-v14)

Sintaxis de unidad de codificación

coding_unit(x0, y0, cb Width, cbHeight, cqtDepth, treeType, modeType) {	***Descriptor***
chType = treeType = = DUAL_TREE_CHROMA? 1 : 0	
...	
if(intra_mip_flag[x0][y0])	
intra_mip_mode*[x0][y0]*	*ae(v)*
else {	
if(sps_mrl_enabled_flag && ((y0 % CtbSizeY) > 0))	
intra_luma_ref_idx*[x0][y0]*	*ae(v)*
if (sps_isp_enabled_flag && intra_luma_ref_idx[x0][y0] = = 0 &&	
*(cbWidth < = **[[**MaxTbSizeY**]]** **{{**64 (u otra opción: Vsize)**}}**&& cbHeight <=*	
[[*MaxTbSizeY***]]** **{{**64 (u otra opción: Vsize)**}}**) &&	
*(cbWidth * cbHeight > MinTbSizeY * MinTbSizeY))*	
intra_subpartitions_mode_flag*[x0][y0]*	*ae(v)*
if(intra_subpartitions_mode_flag[x0][y0] = = 1)	
intra_subpartitions_split_flag*[x0][y0]*	*ae(v)*
if(intra_luma_ref_idx[x0][y0] = = 0)	
intra_luma_mpm_flag*[x0][y0]*	*ae(v)*
if(intra_luma_mpm_flag[x0][y0]) {	
if(intra_luma_ref_idx[x0][y0] = = 0)	
intra_luma_not_planar_flag*[x0][y0]*	*ae(v)*

(continuación)

if(intra_luma_not_planar_flag[x0][y0])	
intra_luma_mpm_idx*[x0][y0]*	*ae(v)*
} else	
intra_luma_mpm_remainder*[x0][y0]*	*ae(v)*
}	
}	
...	

Sintaxis de árbol de transformada

transform_tree(x0, y0, tbWidth, tbHeight, treeType, chType) {	***Descriptor***
InferTuCbfLuma = 1	
if(IntraSubPartitionsSplitType = = ISP_NO_SPLIT && !cu_sbt_flag) {	
if(tbWidth > MaxTbSizeY || tbHeight > MaxTbSizeY) {	
verSplitFirst = (tbWidth > MaxTbSizeY && tbWidth > tbHeight) ? 1 : 0	
trafoWidth = verSplitFirst ? (tbWidth /2) : tbWidth	
trafoHeight = IverSplitFirst ? (tbHeight / 2) : tbHeight	
transform_tree(x0, y0, trafoWidth, trafoHeight, chType)	
if(verSplitFirst)	
transform_tree(x0 + trafoWidth, y0, trafoWidth, trafoHeight, treeType, chType)	
else	
transform _tree(x0, y0 + trafoHeight, trafo Width, trafoHeight, treeType, chType)	
} else {	
transform_unit(x0, y0, tbWidth, tbHeight, treeType, 0, chType)	
}	
} else if(cu_sbt_flag) {	
if(!cu_sbt_horizontal_flag) {	
*trafoWidth = tbWidth * SbtNumFourthsTb0 / 4*	
transform_unit(x0, y0, trafoWidth, tbHeight, treeType, 0, 0)	
transform_unit(x0 + trafoWidth, y0, tbWidth - trafoWidth, tbHeight, treeType, 1, 0)	
} else {	
*trafoHeight = tbHeight * SbtNumFourthsTb0 / 4*	
transform_unit(x0, y0, tbWidth, trafoHeight, treeType, 0, 0)	
transform_unit(x0, y0 + trafoHeight, tbWidth, tbHeight - trafoHeight, treeType, 1, 0)	
}	
} else if(IntraSubPartitionsSplitType = = ISP_HOR_SPLIT) {	
trafoHeight = tbHeight / NumIntraSubPartitions	
for(partIdx = 0; partIdx < NumIntraSubPartitions; partIdx++) {	
{{ *if(tbWidth > MaxTbSizeY) {*	
*transform_unit(x0, y0 + trafoHeight * partIdx, tbWidthl2, trafoHeight, treeType, partIdx, 0)*	
transform_unit(x0+	
*tb Width/2, y0 + trafoHeight * partIdx, tbWidth/2, trafoHeight, treeType, partIdx, 0)*	
} else **{}}**	
*transform_unit(x0, y0 + trafoHeight * partIdx, tbWidth, trafoHeight, treeType, partIdx, 0)*	
{{{}}}	
{{{}}}	
} else if(IntraSubPartitionsSplitType = = ISP_VER_SPLIT) {	
trafoWidth = tbWidth / NumIntraSubPartitions	
for(partIdx = 0; partIdx < NumIntraSubPartitions; partIdx++) {{{	
if(Height > MaxTbSizeY) {	
*transform_unit(x0 + trafoWidth * partIdx, y0, trafoWidth, tbHeightl2, treeType, partIdx, 0)*	
*transform_unit(x0 + trafoWidth * partIdx, y0*	
+tbHeightl2, trafoWidth, tbHeightl2, treeType, partIdx, 0)	
} else **{}}**	
*transform_unit(x0 + trafoWidth * partIdx, y0, trafoWidth, tbHeight, treeType, partIdx, 0)*	
{{{}}}	
{{{}}}	
}	
}	

Semántica de unidad de codificación

intra_subpartitions_split_flag*[x0][y0] especifica si el tipo de división intra subparticiones es horizontal o vertical. Cuando intra_subpartitions_split_flag[x0][y0] no está presente, se infiere como sigue:*

- *Si cbHeight es mayor que* **[[***MaxTbSizeY***]] {{***64 (u otra opción: Vsize)***}}**, *intra_subpartitions_split_flag[x0][y0] se infiere que es igual a 0.*
-
- *De lo contrario (cbWidth es mayor que* **[[***MaxTbSizeY***]] {{***64 (u otra opción: Vsize)***}}**),
- *intra_subpartitions_split_flag[x0][y0] se infiere que es igual a 1.*
-

5.6 Realización n.° 6 sobre la aplicación de MIP dependiente del tamaño de VPDU, o 64x64

La realización a continuación es para el método inventado que hace que el ISP dependa del tamaño de VPDU. Las modificaciones se basan en el último borrador de trabajo de VVC (JVET-O2001-v14)

Sintaxis de unidad de codificación

coding_unit(x0, y0, cbWidth, cbHeight, cqtDepth, tree Type, mode Type) {	***Descriptor***
chType = treeType = = DUAL_TREE_CHROMA? 1 : 0	
...	
} else {	
if(sps_bdpcm_enabled_flag &&	
cb Width < = MaxTsSize && cbHeight <= MaxTsSize)	
intra_bdpcm_flag	*ae(v)*
if(intra_bdpcm_flag)	
intra_bdpcm_dir_flag	*ae(v)*
else {	
if(sps_mip_enabled_flag &&	
(Abs(Log2(cbWidth) - Log2(cbHeight)) <= 2) &&	
cbWidth <= **[[***MaxTbSizeY***]] {{***64 (u otra opción: Vsize)***}}** *&&*	
cbHeight <= **[[***MaxTbSizeY***]] {{***64 (u otra opción: Vsize)***}}**)	
intra_mip_flag*[x0][y0]*	*ae(v)*
if(intra_mip_flag[x0][y0])	
intra_mip_mode*[x0][y0]*	*ae(v)*
else {	

5.7 Realización n.° 7 sobre la aplicación de SBT dependiente del tamaño de transformada máximo

Sintaxis de RBSP de conjunto de parámetros de secuencia

sps_sbt_enabled_flag	*u(1)*
[[if(sps_sbt_enabled_flag)	
sps_sbt_max_size_64_flag	*u(1)***]]**
sps_affine_enabled_flag	*u(1)*
if(sps_affine_enabled_flag) {	
sps_affine_type_flag	*u(1)*
sps_affine_amvr_enabled_flag	*u(1)*
sps_affine_prof_enabled_flag	*u(1)*
}	

Sintaxis de unidad de codificación

coding_unit(x0, y0, cbWidth, cbHeight, cqtDepth, tree Type, mode Type) {	*Descriptor*
chType = treeType = = DUAL_TREE_CHROMA? 1 : 0	
...	
if(cu_cbf) {	
if(CuPredMode[chType][x0][y0] = = MODE_INTER && sps_sbt_enabled_flag && !ciip_flag[x0][y0] && !MergeTriangleFlag[x0][y0]) {	
if(cbWidth <= **[[***MaxSbtSize***]] {{***MaxTbSizeY***}}** *&& cbHeight <=* **[[***MaxSbtSize***]] {{***WaxTbSizeY***}}** *) {*	
allowSbtVerH = cbWidth >= 8	
allowSbtVerQ = cbWidth >= 16	

(continuación)

allowSbtHorH = cbHeight >= 8	
allowSbtHorQ = cbHeight >= 16	
if(allowSbtVerH || allowSbtHorH || allowSbtVerQ || allowSbtHorQ)	
cu_sbt_flag	*ae(v)*
}	
if(cu_sbt_flag) {	
if((allowSbtVerH || allowSbtHorH) && (allowSbtVerQ || allowSbtHorQ))	
cu_sbt_quad_flag	*ae(v)*
if((cu_sbt_quad_flag && allowSbtVerQ && allowSbtHorQ) || (!cu_sbt_quad_flag && allowSbtVerH && allowSbtHorH))	
cu_sbt_horizontal_flag	*ae(v)*
cu_sbt_pos_flag	*ae(v)*
}	
}	
...	

Semántica de RBSP de conjunto de parámetros de secuencia

[[*sps_sbt_max_size_64_flag* *igual a 0 especifica que la anchura y altura de CU máximas para permitir la transformada de subbloque es de 32 muestras de luma. sps_sbt_max_size_64_flag igual a 1 especifica que la anchura y altura de CU máximas para permitir la transformada de subbloque es 64 muestras de luma.*

$$MaxSbtSize = \text{Min}(\text{MaxTbSizeY}, \text{sps\_sbt\_max\_size\_64\_flag} \ ? \ 64 : 32) \quad (7\text{-}32)]]$$

5.8 Realización n.° 8 sobre la aplicación de omisión de transformada dependiente del tamaño de transformada máximo

Sintaxis de RBSP de conjunto de parámetros de imagen

pic_parameter_set_rbsp( ) {	***Descriptor***
...	
num_ref_idx_default_active_minus1 *[i]*	*ue(v)*
rpl1_idx_present_flag	*u(1)*
init_qp_minus26	*se(v)*
[[*if(sps_transform_skip_enabled_flag)*	
log2_transform_skip_max_size_minus2	*ue(v)***]]**
cu_qp_delta_enabled_flag	*u(1)*
if(cu_qp_delta_enabled_flag)	
cu_qp_delta_subdiv	*ue(v)*
pps_cb_qp_offset	*se(v)*
pps_cr_qp_offset	*se(v)*

Sintaxis de unidad de codificación

coding_unit(x0, y0, cbWidth, cbHeight, cqtDepth, treeType, modeType) {	***Descriptor***
...	
if(CuPredMode[chType][x0][y0] = = MODE_INTRA || CuPredMode[chType][x0][y0] = = MODE_PLT) {	
if(treeType = = SINGLE_TREE || treeType = = DUAL_TREE_LUMA) {	
if(pred_mode_plt_flag) {	
if(treeType = = DUAL_TREE_LUMA)	
palette_coding(x0, y0, cbWidth, cbHeight, 0, 1)	
else / SINGLE_TREE */*	
palette_coding(x0, y0, cbWidth, cbHeight, 0, 3)	
} else {	
if(sps_bdpcm_enabled_flag && cbWidth < = **[[***MaxTsSize***]] {{***MaxTbSizeY***}}** *&& cbHeight < =* **[[***MaxTsSize***]] {{***MaxTbSizeY***}}**)	
intra_bdpcm_flag	*ae (v)*
if(intra_bdpcm_flag)	
intra_bdpcm_dir_flag	*ae (v)*
else {	

Sintaxis de unidad de transformada

transform_unit(x0, y0, tbWidth, tbHeight, tree Type, subTuIndex, chType) {	***Descriptor***
...	
if(tu_cbf_luma[x0][y0] && treeType != DUAL_TREE_CHROMA && (tbWidth <= 32) && (tbHeight <= 32) && (IntraSubPartitionsSplit[x0][y0] = = ISP_NO_SPLIT) && (!cu_sbt_flag))	
{	
if(sps_transform_skip_enabled_flag && !BdpcmFlag[x0][y0] && tbWidth <= [[MaxTsSize]] {{MaxTbSizeY}} && tbHeight <= [[MaxTsSize]] {{MaxTbSizeY}})	
transform_skip_ flag*[x0][y0]*	*ae(v)*
if(((CuPredMode[chType][x0][y0] = = MODE_INTER && sps_explicit_mts_inter_enabled_flag) || (CuPredMode[chType][x0][y0] = = MODE_INTRA && sps_explicit_mts_intra_enabled_flag)) && (!transform_skip_flag[x0][y0]))	
tu_mts_idx*[x0][y0]*	*ae(v)*

Semántica de RBSP de conjunto de parámetros de imagen

[[***log2_transform_skip_max_size_minus2*** *especifica el tamaño de bloque máximo usado para omisión de transformada, y deberá estar en el intervalo de 0 a 3.*

Cuando no está presente, el valor de log2_transform_skip_max_size_minus2 se infiere que es igual a 0.

La variable MaxTsSize se establece igual a 1 << (log2_transform_skip_max_size_minus2 + 2).]]

5.9 Realización n.° 9 en ciip_flag dependiente del tamaño de transformada máximo

Sintaxis de datos de fusión

merge_data(x0, y0, cbWidth, cbHeight, chType) {	***Descriptor***
if (CuPredMode[chType][x0][y0] = = MODE_IBC) {	
if(MaxNumIbcMergeCand > 1)	
merge_idx*[x0][y0]*	*ae(v)*
} else {	
if(MaxNumSubblockMergeCand > 0 && cbWidth >= 8 && cbHeight > = 8)	
merge_subblock_flag*[x0][y0]*	*ae(v)*
if(merge_subblock_flag[x0][y0] = = 1) {	
if(MaxNumSubblockMergeCand > 1)	
merge_subblock_idx*[x0][y0]*	*ae(v)*
} else {	
*if((cbWidth * cbHeight) > = 64 && ((sps_ciip_enabled_flag && cu_skip_flag[x0][y0] = = 0 && cbWidth <= {{MaxTbSizeY}} && cbHeight <= {{MaxTbSizeY}}) || (sps_triangle_enabled_flag && MaxNumTriangleMergeCand > 1 && slice_type = = B)))*	
regular_merge_flag*[x0][y0]*	*ae(v)*
if (regular_merge_flag[x0][y0] = = 1){	
if(sps_mmvd_enabled_flag)	
mmvd_merge_flag*[x0][y0]*	*ae(v)*
if(mmvd_merge_flag[x0][y0] = = 1) {	
if(MaxNumMergeCand > 1)	
mmvd_cand_flag*[x0][y0]*	*ae(v)*
mmvd_distance_idx*[x0][y0]*	*ae(v)*
mmvd_direction_idx*[x0][y0]*	*ae(v)*
} else {	
if(MaxNumMergeCand > 1)	
merge_idx*[x0][y0]*	*ae(v)*
}	
} else {	
*if(sps_ciip_enabled_flag && sps_triangle_enabled_flag && MaxNumTriangleMergeCand > 1 && slice_type = = B && cu_skip_flag[x0][y0] = = 0 && (cbWidth * cbHeight) >= 64 && cbWidth <= {{MaxTbSizeY}} && cbHeight <= {{MaxTbSizeY}}) {*	
ciip_flag*[x0][y0]*	*ae(v)*
if(ciip_flag[x0][y0] && MaxNumMergeCand > 1)	
merge_idx*[x0][y0]*	*ae(v)*
if(!ciip_flag[x0][y0] && MaxNumTriangleMergeCand > 1) {	
merge_triangle_split_dir*[x0][y0]*	*ae(v)*

merge_triangle_idx0[x0][y0]	ae(v)

(continuación)

merge_triangle_idx1[x0][y0]	ae(v)
}	
}	
}	
}	
}	

ciip_flag[x0][y0] especifica si se aplica la fusión de inter-imagen y la predicción intra-imagen combinadas para la unidad de codificación actual. Los índices de matriz x0, y0 especifican la ubicación (x0, y0 J de la muestra de luma superior izquierda del bloque de codificación considerado en relación con la muestra de luma superior izquierda de la imagen.

Cuando ciip_flag[x0][y0] no está presente, se infiere como sigue:

- Si todas las siguientes condiciones son verdaderas, ciip_flag[x0][y0] se infiere que es igual a 1:
  - sps_ciip_enabled_flag es igual a 1.
  - general_merge_flag[x0][y0] es igual a 1.
  - merge_subblock_flag[x0][y0] es igual a 0.
  - regular_merge_flag[x0][y0] es igual a 0.
  - cbWidth es menor o igual que **{{**MaxTbSizeY**}}**.
  - cbHeight es menor o igual que **{{**MaxTbSizeY**}}**.
  - cbWidth * cbHeight es mayor o igual que 64.
- De lo contrario, ciip_flag[x0][y0] se infiere que es igual a 0.

5.10 Realización n.° 10 sobre sps_max_luma_transform_size_64_flag dependiente de CtbSizeY

7.3.2.3 Sintaxis de RBSP de conjunto de parámetros de secuencia

seq_parameter_set_rbsp( ) {	**Descriptor**
...	
if(qtbtt_dual_tree_intra_flag) {	
sps_log2_diff_min_qt_min_cb_intra_slice_chroma	ue(v)
sps_max_mtt_hierarchy_depth_intra_slice_chroma	ue(v)
if (sps_max_mtt_hierarchy_depth_intra_slice_chroma != 0) {	
sps_log2_diff_max_bt_min_qt_intra_slice_chroma	ue(v)
sps_log2_diff_max_tt_min_qt_intra_slice_chroma	ue(v)
}	
}	
{{if(log2_ctu_size_minus5 != 0)**}}**	
sps_max_luma_transform_size_64_flag	u(1)
if(ChromaArrayType != 0) {	
same_qp_table_for_chroma	u(1)
for(i = 0; i < same_qp_table_for_chroma ? 1 : 3; i++) {	
num_points_in_qp_table_minus1[i]	ue(v)
for(j = 0; j <= num_points_in_qp_table_minus1[i]; j++) {	
delta_qp_in_val_minus1[i][j]	ue(v)
delta_qp_out_val[i][j]	ue(v)
}	
}	
}	
...	

7.4.3.3. Semántica de RBSP de conjunto de parámetros de secuencia

... ***sps_max_luma_transform_size_64 flag*** igual a 1 especifica que el tamaño de transformada máximo en muestras de luma es igual a 64. sps_max_luma_transform_size_64_flag igual a 0 especifica que el tamaño de transformada máximo en muestras de luma es igual a 32.

Cuando **[[**CtbSizeY es menor que 64,**]]** **{{**sps_max_luma_transform_size_64_flag no está presente,**}}** el valor de sps_max_luma_transform_size_64_flag **[[**deberá**]]** **{{**se infiere para**}}** ser igual a 0.

Las variables MinTbLog2SizeY, MaxTbLog2SizeY, MinTbSizeY, y MaxTbSizeY se derivan como sigue:

$$MinTbLog2SizeY = 2 \quad (7\text{-}27)$$

$$MaxTbLog2SizeY = sps\_max\_luma\_transform\_size\_64\_flag\ ?\ 6\ :\ 5 \quad (7\text{-}28)$$

$$MinTbSizeY = 1\ <<\ MinTbLog2SizeY \quad (7\text{-}29)$$

$$MaxTbSizeY = 1\ <<\ MaxTbLog2SizeY \quad (7\text{-}30)$$

La figura 2 es un diagrama de bloques de un aparato de procesamiento de vídeo 200. El aparato 200 puede usarse para implementar uno o más de los métodos descritos en el presente documento. El aparato 200 puede incorporarse en un teléfono inteligente, tableta, ordenador, receptor de Internet de las Cosas (IoT), y así sucesivamente. El aparato 200 puede incluir uno o más procesadores 202, una o más memorias 204 y hardware de procesamiento de vídeo 206. El procesador o procesadores 202 pueden configurarse para implementar uno o más métodos descritos en el presente documento. La memoria (memorias) 204 puede usarse para almacenar datos y código usado para implementar los métodos y técnicas descritos en el presente documento. El hardware de procesamiento de vídeo 206 puede usarse para implementar, en circuitería de hardware, algunas técnicas descritas en el presente documento. En algunas realizaciones, el hardware de procesamiento de vídeo 206 puede estar al menos parcialmente dentro de los procesadores 202 (por ejemplo, un coprocesador de gráficos).

En algunas realizaciones, los métodos de codificación de vídeo pueden implementarse usando un aparato que se implementa en una plataforma de hardware como se describe con respecto a la figura 2.

Algunas realizaciones de la tecnología divulgada incluyen tomar una decisión o determinación para habilitar una herramienta o modo de procesamiento de vídeo. En un ejemplo, cuando la herramienta o modo de procesamiento de vídeo está habilitado, el codificador usará o implementará la herramienta o modo en el procesamiento de un bloque de vídeo, pero no necesariamente puede modificar el flujo de bits resultante basándose en el uso de la herramienta o modo. Es decir, una conversión del bloque de vídeo a la representación de flujo de bits del vídeo usará la herramienta o modo de procesamiento de vídeo cuando se habilita basándose en la decisión o determinación. En otro ejemplo, cuando la herramienta o modo de procesamiento de vídeo está habilitado, el descodificador procesará el flujo de bits con el conocimiento de que el flujo de bits se ha modificado basándose en la herramienta o modo de procesamiento de vídeo. Es decir, se realizará una conversión de la representación de flujo de bits del vídeo al bloque de vídeo usando la herramienta o modo de procesamiento de vídeo que se habilitó basándose en la decisión o determinación.

Algunas realizaciones de la tecnología divulgada incluyen tomar una decisión o determinación para deshabilitar una herramienta o modo de procesamiento de vídeo. En un ejemplo, cuando la herramienta o modo de procesamiento de vídeo está deshabilitado, el codificador no usará la herramienta o modo en la conversión del bloque de vídeo a la representación de flujo de bits del vídeo. En otro ejemplo, cuando la herramienta o modo de procesamiento de vídeo está deshabilitado, el descodificador procesará el flujo de bits con el conocimiento de que el flujo de bits no se ha modificado usando la herramienta o modo de procesamiento de vídeo que se habilitó basándose en la decisión o determinación.

La figura 3 es un diagrama de bloques que muestra un sistema de procesamiento de vídeo 300 de ejemplo en el que pueden implementarse diversas técnicas divulgadas en el presente documento. Diversas implementaciones pueden incluir algunos o todos los componentes del sistema 300. El sistema 300 puede incluir la entrada 302 para recibir contenido de vídeo. El contenido de vídeo puede recibirse en un formato sin procesar o sin comprimir, por ejemplo, valores de píxel de múltiples componentes de 8 o 10 bits, o puede estar en un formato comprimido o codificado. La entrada 302 puede representar una interfaz de red, una interfaz de bus periférico o una interfaz de almacenamiento. Ejemplos de interfaz de red incluyen interfaces cableadas tales como Ethernet, red óptica pasiva (PON), etc., e interfaces inalámbricas tales como Wi-Fi o interfaces celulares.

El sistema 300 puede incluir un componente de codificación 304 que puede implementar los diversos métodos de codificación descritos en el presente documento. El componente de codificación 304 puede reducir la tasa de bits promedio de vídeo desde la entrada 302 a la salida del componente de codificación 304 para producir una representación codificada del vídeo. Por lo tanto, las técnicas de codificación a veces se denominan técnicas de compresión de vídeo o transcodificación de vídeo. La salida del componente de codificación 304 puede almacenarse o transmitirse a través de una comunicación conectada, como se representa por el componente 306. La representación de flujo de bits almacenada o comunicada (o codificada) del vídeo recibido en la entrada 302 puede usarse por el componente 308 para generar valores de píxel o vídeo visualizable que se envía a una interfaz de visualización 310. El proceso de generación de vídeo visible por el usuario a partir de la representación de flujo de bits se denomina en

ocasiones descompresión de vídeo. Además, mientras que ciertas operaciones de procesamiento de vídeo se denominan operaciones o herramientas de "codificación", se apreciará que las herramientas u operaciones de codificación se usan en un codificador y las correspondientes herramientas u operaciones de descodificación que invierten los resultados de la codificación se realizarán por un descodificador.

Ejemplos de una interfaz de bus periférico o una interfaz de visualización pueden incluir bus serie universal (USB) o interfaz multimedia de alta definición (HDMI) o Displayport, y así sucesivamente. Los ejemplos de interfaces de almacenamiento incluyen interfaz SATA (conexión de tecnología avanzada en serie), PCI, IDE y similares. Las técnicas descritas en el presente documento pueden realizarse en diversos dispositivos electrónicos tales como teléfonos móviles, portátiles, teléfonos inteligentes u otros dispositivos que son capaces de realizar procesamiento de datos digitales y/o visualización de vídeo.

La Figura 4 es un diagrama de flujo para un método 400 de procesamiento de vídeo. El método 400 incluye, en la operación 410, usar una dimensión de una unidad de datos de canalización virtual (VPDU) usada para una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo y una representación de flujo de bits del vídeo para realizar una determinación de si se habilita una partición de árbol ternario (TT) o de árbol binario (BT) de un bloque de vídeo de los uno o más bloques de vídeo, siendo la dimensión igual a VSize en muestras de luma.

El método 400 incluye, en la operación 420, realizar, basándose en la determinación, la conversión.

La Figura 5 es un diagrama de flujo para un método 500 de procesamiento de vídeo. El método 500 incluye, en la operación 510, usar, para una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo y una representación de flujo de bits del vídeo, una dimensión de un bloque de vídeo de los uno o más bloques de vídeo para realizar una determinación de si se habilita una partición de árbol ternario (TT) o de árbol binario (BT) del bloque de vídeo.

El método 500 incluye, en la operación 520, realizar, basándose en la determinación, la conversión.

La Figura 6 es un diagrama de flujo para un método 600 de procesamiento de vídeo. El método 600 incluye, en la operación 610, usar una altura o una anchura de un bloque de vídeo para realizar una determinación de si se habilita una herramienta de codificación para una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo que comprenden el bloque de vídeo y una representación de flujo de bits del vídeo, basándose la determinación en una comparación entre la altura o la anchura con un valor N, y siendo N un número entero positivo.

El método 600 incluye, en la operación 620, realizar, basándose en la determinación, la conversión.

La Figura 7 es un diagrama de flujo para un método 700 de procesamiento de vídeo. El método 700 incluye, en la operación 710, usar la comparación entre una altura o una anchura de un bloque de vídeo y un tamaño de un bloque de transformada para realizar una determinación de si se habilita una herramienta de codificación para una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo que comprenden el bloque de vídeo y una representación de flujo de bits del vídeo.

El método 700 incluye, en la operación 720, realizar, basándose en la determinación, la conversión.

La Figura 8 es un diagrama de flujo para un método 800 de procesamiento de vídeo. El método 800 incluye, en la operación 810, usar una altura o una anchura de un bloque de vídeo para realizar una determinación de si se habilita una herramienta de codificación para una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo que comprenden el bloque de vídeo y una representación de flujo de bits del vídeo.

El método 800 incluye, en la operación 820, realizar, basándose en la determinación, la conversión.

La Figura 9 es un diagrama de flujo para un método 900 de procesamiento de vídeo. El método 900 incluye, en la operación 910, usar una comparación entre una dimensión de una subpartición de un bloque de vídeo y un tamaño de transformada máximo para realizar (a) una determinación de si se habilita un modo de predicción intra subpartición (ISP) para una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo que comprenden el bloque de vídeo, y (b) una selección de uno o más tipos de partición permitidos para la conversión.

El método 900 incluye, en la operación 920, realizar, basándose en la determinación y la selección, la conversión.

La Figura 10 es un diagrama de flujo para un método 1000 de procesamiento de vídeo. El método 1000 incluye, en la operación 1010, realizar una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo y una representación de flujo de bits del vídeo, comprendiendo la conversión una

herramienta de codificación que se ha deshabilitado, y elementos de sintaxis relacionados con la herramienta de codificación que se excluyen de la representación de flujo de bits y se infiere que es un valor predeterminado que especifica que la herramienta de codificación está deshabilitada.

La Figura 11 es un diagrama de flujo para un método 1100 de procesamiento de vídeo. El método 1100 incluye, en la operación 1110, realizar una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo y una representación de flujo de bits del vídeo, comprendiendo la conversión una herramienta de codificación que se ha deshabilitado, y comprendiendo la representación de flujo de bits elementos de sintaxis relacionados con la herramienta de codificación que se infiere que es un valor predeterminado basándose en la herramienta de codificación que está deshabilitada.

La Figura 12 es un diagrama de flujo para un método 1200 de procesamiento de vídeo. El método 1200 incluye, en la operación 1210, usar una dimensión de una unidad de datos de canalización virtual (VPDU) y/o un tamaño de transformada máximo usados para una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo y una representación de flujo de bits del vídeo para realizar una determinación de si se habilita una partición implícita (QT) de un bloque de vídeo de los uno o más bloques de vídeo.

El método 1200 incluye, en la operación 1220, realizar, basándose en la determinación, la conversión.

La Figura 13 es un diagrama de flujo para un método 1300 de procesamiento de vídeo. El método 1300 incluye, en la operación 1310, realizar una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo y una representación de flujo de bits del vídeo, comprendiendo la conversión una transformada de subbloque (SBT), y una altura máxima o una anchura máxima de la SBT basándose en un tamaño de transformada máximo.

La Figura 14 es un diagrama de flujo para un método 1400 de procesamiento de vídeo. El método 1400 incluye, en la operación 1410, realizar una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo y una representación de flujo de bits del vídeo, comprendiendo la conversión un modo de omisión de transformada y/o un modo de modulación de código de pulsos diferencial basada en intra bloque (BDPCM), y un tamaño de bloque máximo usado para el modo de omisión de transformada basándose en un tamaño de transformada máximo.

La Figura 15 es un diagrama de flujo para un método 1500 de procesamiento de vídeo. El método 1500 incluye, en la operación 1510, usar una comparación entre una altura o una anchura de un bloque de vídeo y un tamaño de transformada máximo para realizar una determinación de si se habilita un modo de inter intra predicción combinado (CIIP) para una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo que comprenden el bloque de vídeo y una representación de flujo de bits del vídeo.

El método 1500 incluye, en la operación 1520, realizar, basándose en la determinación, la conversión.

La Figura 16 es un diagrama de flujo para un método 1600 de procesamiento de vídeo. El método 1600 incluye, en la operación 1610, tomar una determinación, para una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo y una representación de flujo de bits del vídeo, con respecto a la partición de un bloque de vídeo de los uno o más bloques de vídeo codificados con inter intra predicción combinada (CIIP).

El método 1600 incluye, en la operación 1620, realizar, basándose en la determinación, la conversión.

La Figura 17 es un diagrama de flujo para un método 1700 de procesamiento de vídeo. El método 1700 incluye, en la operación 1710, realizar una conversión entre un vídeo que comprende una región de vídeo que comprende múltiples bloques de vídeo y una representación de flujo de bits del vídeo de acuerdo con una regla, especificando la regla que un tamaño de bloque máximo de los múltiples bloques de vídeo en la región de vídeo que se codifican en la representación de flujo de bits usando una codificación de transformada determina un tamaño de bloque máximo de los múltiples bloques de vídeo en la región de vídeo que se codifican en la representación de flujo de bits sin usar codificación de transformada.

La Figura 18 es un diagrama de flujo para un método 1800 de procesamiento de vídeo. El método 1800 incluye, en la operación 1810, realizar una conversión entre un vídeo que comprende una región de vídeo que comprende múltiples bloques de vídeo y una representación de flujo de bits del vídeo de acuerdo con una regla, especificando la regla que un proceso de mapeo de luma con escalado de croma (LMCS) está deshabilitado para la región de vídeo cuando la codificación sin pérdidas está habilitada para la región de vídeo, y siendo la región de vídeo una secuencia, una imagen, una subimagen, un corte, un grupo de mosaicos, un mosaico, un ladrillo, una fila de unidad de árbol de codificación (CTU), una CTU, una unidad de codificación (CU), una unidad de predicción (PU), una unidad de transformada (TU) o un subbloque.

En los métodos 400-1800, en el modo de ISP, un bloque de vídeo de los uno o más bloques de vídeo se particiona en

múltiples subparticiones antes de la aplicación de una intra-predicción y transformada.

En los métodos 400-1800, la SBT comprende una o más transformadas que se aplican por separado a una o más particiones de un bloque de vídeo de los uno o más bloques de vídeo.

En los métodos 400-1800, el modo de omisión de transformada comprende omitir procesos de transformada y transformada inversa para una herramienta de codificación correspondiente, y en el modo de BDPCM, un residual de una intra predicción del bloque de vídeo actual se codifica de manera predictiva usando una operación de modulación de codificación de pulso diferencial.

En los métodos 400-1800, en el modo de CIIP, una predicción final del bloque de vídeo se basa en una suma ponderada de una inter predicción del bloque de vídeo y una intra predicción del bloque de vídeo.

En los métodos 400-1800, el proceso de LMCS comprende muestras de luma de la región de vídeo que se vuelven a conformar entre un primer dominio y un segundo dominio y un residual de croma que se escala de una manera dependiente de luma.

Las soluciones divulgadas y otras, ejemplos, realizaciones, módulos y las operaciones funcionales descritas en este documento pueden implementarse en circuitería electrónica digital, o en software informático, firmware o hardware, incluyendo las estructuras divulgadas en este documento y sus equivalentes estructurales, o en combinaciones de una o más de las mismas. Las realizaciones divulgadas y otras pueden implementarse como uno o más productos de programa informático, es decir, uno o más módulos de instrucciones de programa informático codificadas en un medio legible por ordenador para su ejecución por, o para controlar la operación de, el aparato de procesamiento de datos. El medio legible por ordenador puede ser un dispositivo de almacenamiento legible por máquina, un sustrato de almacenamiento legible por máquina, un dispositivo de memoria, una composición de materia que efectúa una señal propagada legible por máquina, o una combinación de uno o más de ellos. La expresión "aparato de procesamiento de datos" abarca todos los aparatos, dispositivos y máquinas para procesar datos, incluyendo, a modo de ejemplo, un procesador programable, un ordenador o múltiples procesadores u ordenadores. El aparato puede incluir, además de hardware, código que crea un entorno de ejecución para el programa informático en cuestión, por ejemplo, código que constituye firmware de procesador, una pila de protocolos, un sistema de gestión de base de datos, un sistema operativo o una combinación de uno o más de los mismos. Una señal propagada es una señal generada artificialmente, por ejemplo, una señal eléctrica, óptica o electromagnética generada por máquina, que se genera para codificar información para su transmisión a un aparato receptor adecuado.

Un programa informático (también conocido como programa, software, aplicación de software, secuencia de comandos o código) puede escribirse en cualquier forma de lenguaje de programación, incluyendo lenguajes compilados o interpretados, y puede desplegarse en cualquier forma, incluyendo como un programa independiente o como un módulo, componente, subrutina u otra unidad adecuada para su uso en un entorno informático. Un programa informático no corresponde necesariamente a un archivo en un sistema de archivos. Un programa puede almacenarse en una porción de un archivo que contiene otros programas o datos (por ejemplo, una o más secuencias de comandos almacenadas en un documento de lenguaje de marcado), en un único archivo dedicado al programa en cuestión, o en múltiples archivos coordinados (por ejemplo, archivos que almacenan uno o más módulos, subprogramas o porciones de código). Un programa informático puede desplegarse para ejecutarse en un ordenador o en múltiples ordenadores que están ubicados en un sitio o distribuidos a través de múltiples sitios e interconectados por una red de comunicación.

Los procesos y flujos lógicos descritos en este documento pueden realizarse por uno o más procesadores programables que ejecutan uno o más programas informáticos para realizar funciones operando sobre datos de entrada y generando la salida. Los procesos y flujos lógicos también pueden realizarse y el aparato también puede implementarse como, circuitería lógica de propósito especial, por ejemplo, una FPGA (matriz de puertas programables en campo) o un ASIC (circuito integrado de aplicación específica).

Los procesadores adecuados para la ejecución de un programa informático incluyen, a modo de ejemplo, tanto microprocesadores de propósito general como especial, y uno o más procesadores cualesquiera de cualquier tipo de ordenador digital. Generalmente, un procesador recibirá instrucciones y datos de una memoria de solo lectura o de una memoria de acceso aleatorio o de ambas. Los elementos esenciales de un ordenador son un procesador para realizar instrucciones y uno o más dispositivos de memoria para almacenar instrucciones y datos. Generalmente, un ordenador también incluirá, o estará acoplado operativamente para recibir datos desde o transferir datos a, o ambos, uno o más dispositivos de almacenamiento masivo para almacenar datos, por ejemplo, discos magnéticos, magneto-ópticos o discos ópticos. Sin embargo, no es necesario que un ordenador tenga tales dispositivos. Los medios legibles por ordenador adecuados para almacenar instrucciones y datos de programa informático incluyen todas las formas de memoria no volátil, medios y dispositivos de memoria, incluyendo, a modo de ejemplo, dispositivos de memoria semiconductores, por ejemplo, EPROM, EEPROM y dispositivos de memoria flash; discos magnéticos, por ejemplo, discos duros internos o discos extraíbles; discos magneto-ópticos; y discos CD ROM y DVD-ROM. El procesador y la memoria pueden complementarse con, o incorporarse en, circuitería lógica de propósito especial.

Este documento de patente contiene muchos detalles específicos que deberían interpretarse como descripciones de características que pueden ser específicas para realizaciones particulares de técnicas particulares. Ciertas características que se describen en este documento de patente en el contexto de realizaciones separadas también pueden implementarse en combinación con una única realización. A la inversa, diversas características que se describen en el contexto de una única realización también pueden implementarse en múltiples realizaciones de manera separada o en cualquier subcombinación adecuada. Además, aunque anteriormente pueda haberse descrito que las características actúan en determinadas combinaciones e incluso se haya reivindicado inicialmente como tal, en algunos casos pueden eliminarse de la combinación una o más características de una combinación reivindicada, y la combinación reivindicada puede referirse a una subcombinación o variación de una subcombinación.

De manera similar, aunque las operaciones se representan en los dibujos en un orden particular, esto no debe entenderse como que se requiere que tales operaciones se realicen en el orden particular mostrado ni en orden secuencial, o que todas las operaciones ilustradas se realicen, para lograr los resultados deseables. Además, la separación de diversos componentes del sistema en las realizaciones descritas en este documento de patente no debe entenderse como que requiere tal separación en todas las realizaciones.

Únicamente se describen unas pocas implementaciones y ejemplos y otras implementaciones, se pueden realizar mejoras y variaciones basándose en lo que se describe e ilustra en este documento de patente.

REIVINDICACIONES

1. Un método de procesamiento de datos de vídeo, que comprende:

determinar, para una conversión entre una región de vídeo de un vídeo y un flujo de bits del vídeo, si se permite o no una primera herramienta de codificación para un primer bloque de la región de vídeo, basándose en una altura y una anchura del primer bloque y un tamaño de transformada máximo de la región de vídeo, en donde en un caso en el que está permitida la primera herramienta de codificación, el primer bloque se divide en múltiples bloques de transformada de acuerdo con la primera herramienta de codificación; y
realizar la conversión basándose en la determinación,
en donde, en respuesta a que al menos una de la altura o la anchura del primer bloque sea mayor que el tamaño de transformada máximo, la primera herramienta de codificación no está permitida para el primer bloque;
en donde la primera herramienta de codificación es una herramienta de transformada de subbloque (SBT);
en donde para un segundo bloque de la región de vídeo, una segunda herramienta de codificación se deshabilita en respuesta a que al menos una de una altura o una anchura del segundo bloque sea mayor que 64; y en donde la segunda herramienta de codificación es un modo de copia de intra bloque (IBC).

2. El método de la reivindicación 1, en donde si se incluye o no en el flujo de bits un primer elemento de sintaxis que indica si la primera herramienta de codificación está permitida o no para el primer bloque, se basa en el tamaño de transformada máximo.

3. El método de las reivindicaciones 1 o 2, en donde determinar que la primera herramienta de codificación está permitida para el primer bloque se basa además en que el primer bloque se codifica con un modo de inter predicción y que no se aplica una fusión de inter-imagen y una predicción de intra-imagen combinadas.

4. El método de una cualquiera de las reivindicaciones 1-3, en donde en un caso en el que está permitida la primera herramienta de codificación, el primer bloque se divide en dos bloques de transformada, y en donde un bloque de transformada tiene datos residuales, el otro bloque de transformada no tiene datos residuales.

5. El método de la reivindicación 1, en donde, en respuesta a que al menos una de la altura o la anchura del segundo bloque sea mayor que N, un segundo elemento de sintaxis que indica si la segunda herramienta de codificación está permitida o no para el segundo bloque se excluye del flujo de bits, y un valor del segundo elemento de sintaxis inferido para indicar que la segunda herramienta de codificación no está permitida para el segundo bloque.

6. El método de una cualquiera de las reivindicaciones 1-4, en donde para un tercer bloque de la región de vídeo, una tercera herramienta de codificación se deshabilita en respuesta a que al menos una de una altura o una anchura del tercer bloque sea mayor que M, donde M es un número entero positivo;

en donde la tercera herramienta de codificación es un modo de predicción de paleta;
y preferiblemente, en donde M = 64,
en donde, en respuesta a que al menos una de la altura o la anchura del tercer bloque sea mayor que M, un tercer elemento de sintaxis que indica si la tercera herramienta de codificación está permitida o no para el tercer bloque se excluye del flujo de bits, y un valor del tercer elemento de sintaxis inferido para indicar que la tercera herramienta de codificación no está permitida para el tercer bloque.

7. El método de una cualquiera de las reivindicaciones 1-4, en donde para un cuarto bloque que es un bloque de intra codificación, una cuarta herramienta de codificación se deshabilita en respuesta a que al menos una de una altura o una anchura del cuarto bloque sea mayor que S, donde S es un número entero positivo;

en donde la cuarta herramienta de codificación es un modo de predicción de intra subpartición;
y preferiblemente, en donde S = 64,
en donde en respuesta a que al menos una de la altura o la anchura del cuarto bloque sea mayor que S, un cuarto elemento de sintaxis que indica si la cuarta herramienta de codificación está permitida o no para el cuarto bloque y un quinto elemento de sintaxis que indica si un tipo de división del modo de predicción de intra subpartición es horizontal o vertical se excluyen del flujo de bits, y un valor del cuarto elemento de sintaxis inferido para indicar que la cuarta herramienta de codificación no está permitida para el cuarto bloque.

8. El método de una cualquiera de las reivindicaciones 1-4, en donde para un quinto bloque de la región de vídeo, una quinta herramienta de codificación se deshabilita en respuesta a que al menos una de una altura o una anchura del quinto bloque sea mayor que 64;
en donde la quinta herramienta de codificación es un modo de intra-inter predicción combinada (CIIP).

9. El método de una cualquiera de las reivindicaciones 1-8, en donde la conversión comprende codificar la región de vídeo en el flujo de bits.

10. El método de una cualquiera de las reivindicaciones 1-8, en donde la conversión comprende descodificar la región

de vídeo del flujo de bits.

11. Un aparato para procesar datos de vídeo que comprende un procesador y una memoria no transitoria con instrucciones en la misma, en donde las instrucciones, tras su ejecución por el procesador, hacen que el procesador:

determine, para una conversión entre una región de vídeo de un vídeo y un flujo de bits del vídeo, si se permite o no una primera herramienta de codificación para un primer bloque de la región de vídeo, basándose en una altura y una anchura del primer bloque y un tamaño de transformada máximo de la región de vídeo, en donde en un caso en el que está permitida la primera herramienta de codificación, el primer bloque se divide en múltiples bloques de transformada de acuerdo con la primera herramienta de codificación; y
realice la conversión basándose en la determinación,
en donde, en respuesta a que al menos una de la altura o la anchura del primer bloque sea mayor que el tamaño de transformada máximo, la primera herramienta de codificación no está permitida para el primer bloque;
en donde la primera herramienta de codificación es una herramienta de transformada de subbloque (SBT);
en donde para un segundo bloque de la región de vídeo, una segunda herramienta de codificación se deshabilita en respuesta a que al menos una de una altura o una anchura del segundo bloque sea mayor que 64; y en donde la segunda herramienta de codificación es un modo de copia de intra bloque (IBC).

12. Un medio de almacenamiento legible por ordenador no transitorio que almacena instrucciones que hacen que un procesador:

determine, para una conversión entre una región de vídeo de un vídeo y un flujo de bits del vídeo, si se permite o no una primera herramienta de codificación para un primer bloque de la región de vídeo, basándose en una altura y una anchura del primer bloque y un tamaño de transformada máximo de la región de vídeo, en donde en un caso en el que está permitida la primera herramienta de codificación, el primer bloque se divide en múltiples bloques de transformada de acuerdo con la primera herramienta de codificación; y
realice la conversión basándose en la determinación,

en donde, en respuesta a que al menos una de la altura o la anchura del primer bloque sea mayor que el tamaño de transformada máximo, la primera herramienta de codificación no está permitida para el primer bloque;

en donde la primera herramienta de codificación es una herramienta de transformada de subbloque (SBT);
en donde para un segundo bloque de la región de vídeo, una segunda herramienta de codificación se deshabilita en respuesta a que al menos una de una altura o una anchura del segundo bloque sea mayor que 64; y en donde la segunda herramienta de codificación es un modo de copia de intra bloque (IBC).

13. Un medio de grabación legible por ordenador no transitorio que almacena un flujo de bits de un vídeo que se genera mediante un método realizado por un aparato de procesamiento de vídeo, en donde el método comprende:

determinar, para una región de vídeo del vídeo, si se permite o no una primera herramienta de codificación para un primer bloque de la región de vídeo, basándose en una altura y una anchura del primer bloque y un tamaño de transformada máximo de la región de vídeo, en donde en un caso en el que está permitida la primera herramienta de codificación, el primer bloque se divide en múltiples bloques de transformada de acuerdo con la primera herramienta de codificación; y
generar el flujo de bits basándose en la determinación,

en donde, en respuesta a que al menos una de la altura o la anchura del primer bloque sea mayor que el tamaño de transformada máximo, la primera herramienta de codificación no está permitida para el primer bloque;
en donde la primera herramienta de codificación es una herramienta de transformada de subbloque (SBT);
en donde para un segundo bloque de la región de vídeo, una segunda herramienta de codificación se deshabilita en respuesta a que al menos una de una altura o una anchura del segundo bloque sea mayor que 64; y en donde la segunda herramienta de codificación es un modo de copia de intra bloque (IBC).

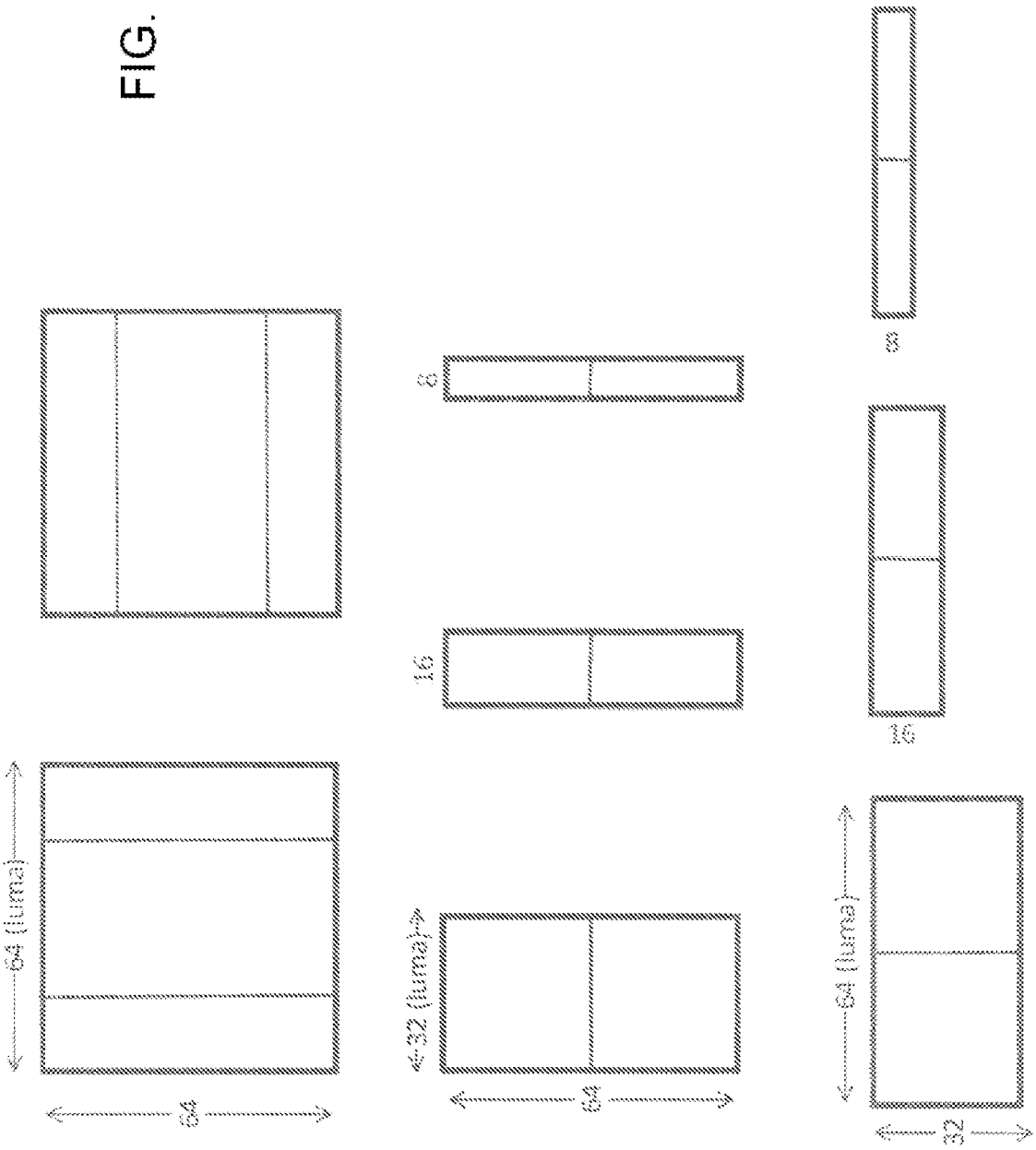
14. Un método para almacenar el flujo de bits de un vídeo, que comprende:

determinar, para una región de vídeo del vídeo, si se permite o no una primera herramienta de codificación para un primer bloque de la región de vídeo, basándose en una altura y una anchura del primer bloque y un tamaño de transformada máximo de la región de vídeo, en donde en un caso en el que está permitida la primera herramienta de codificación, el primer bloque se divide en múltiples bloques de transformada de acuerdo con la primera herramienta de codificación;
generar el flujo de bits basándose en la determinación; y
almacenar el flujo de bits en un medio de grabación legible por ordenador no transitorio,

en donde, en respuesta a que al menos una de la altura o la anchura del primer bloque sea mayor que el tamaño de transformada máximo, la primera herramienta de codificación no está permitida para el primer

bloque;
en donde la primera herramienta de codificación es una herramienta de transformada de subbloque (SBT);
en donde para un segundo bloque de la región de vídeo, una segunda herramienta de codificación se deshabilita en respuesta a que al menos una de una altura o una anchura del segundo bloque sea mayor que 64; y en donde la segunda herramienta de codificación es un modo de copia de intra bloque (IBC).

FIG. 1



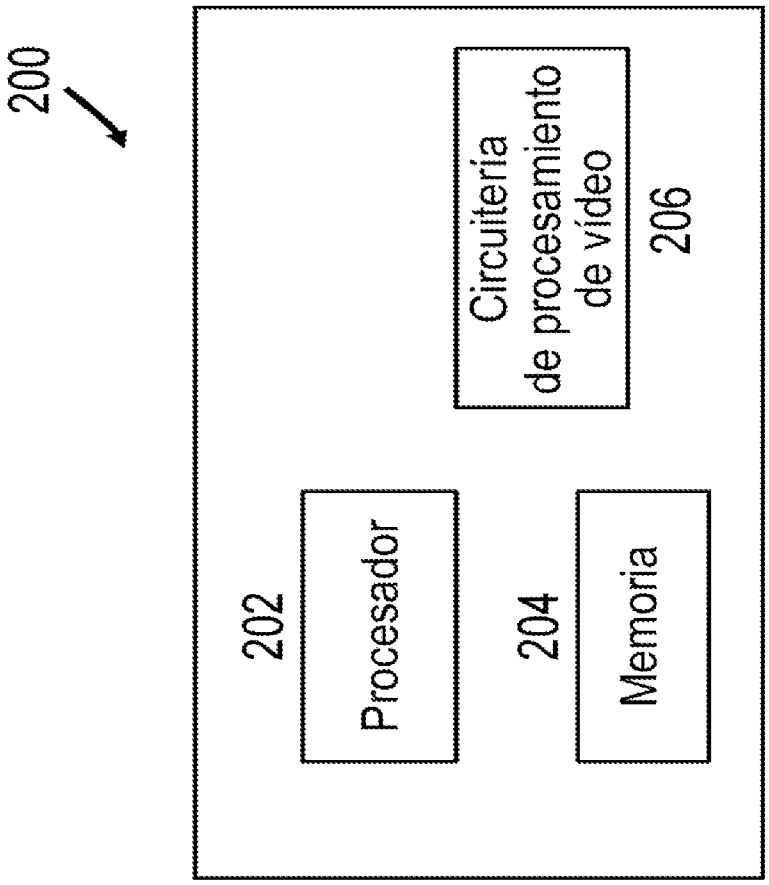

200
202
Procesador
204
Memoria
Circuiteria de procesamiento de vídeo
206

FIG. 2

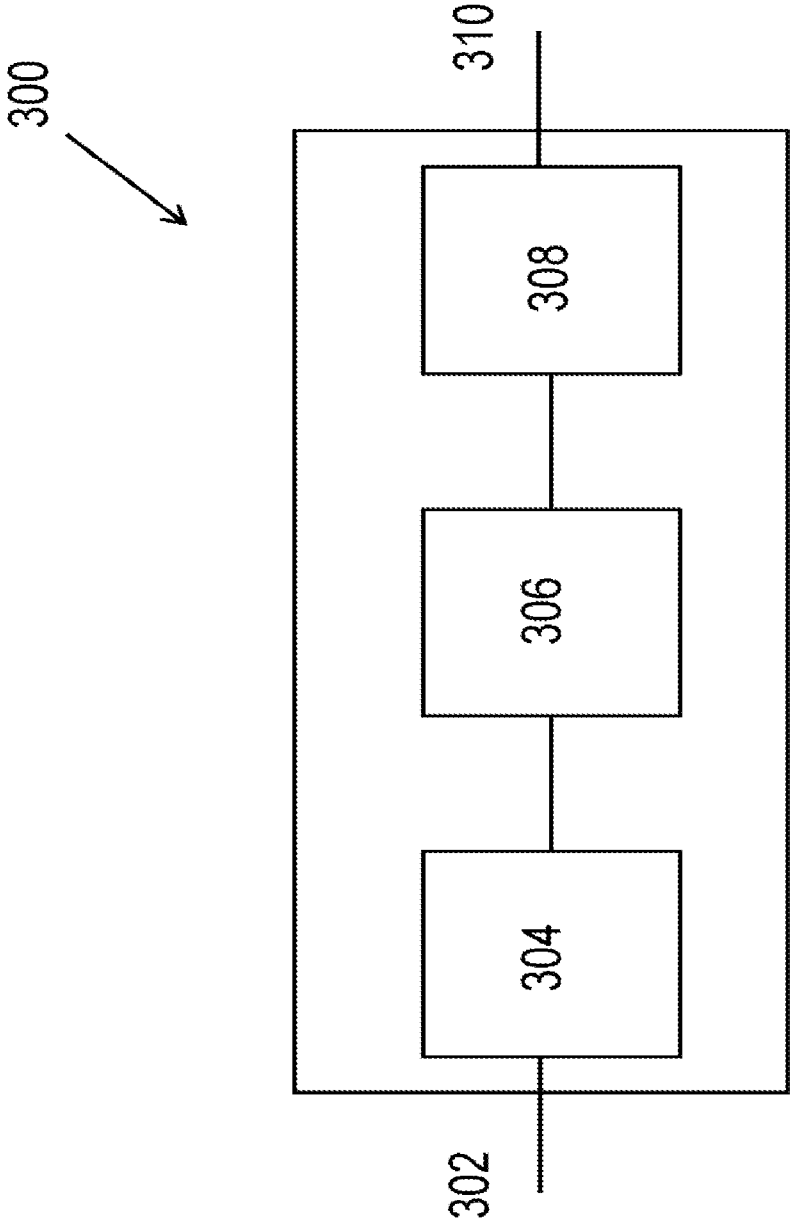


FIG. 3

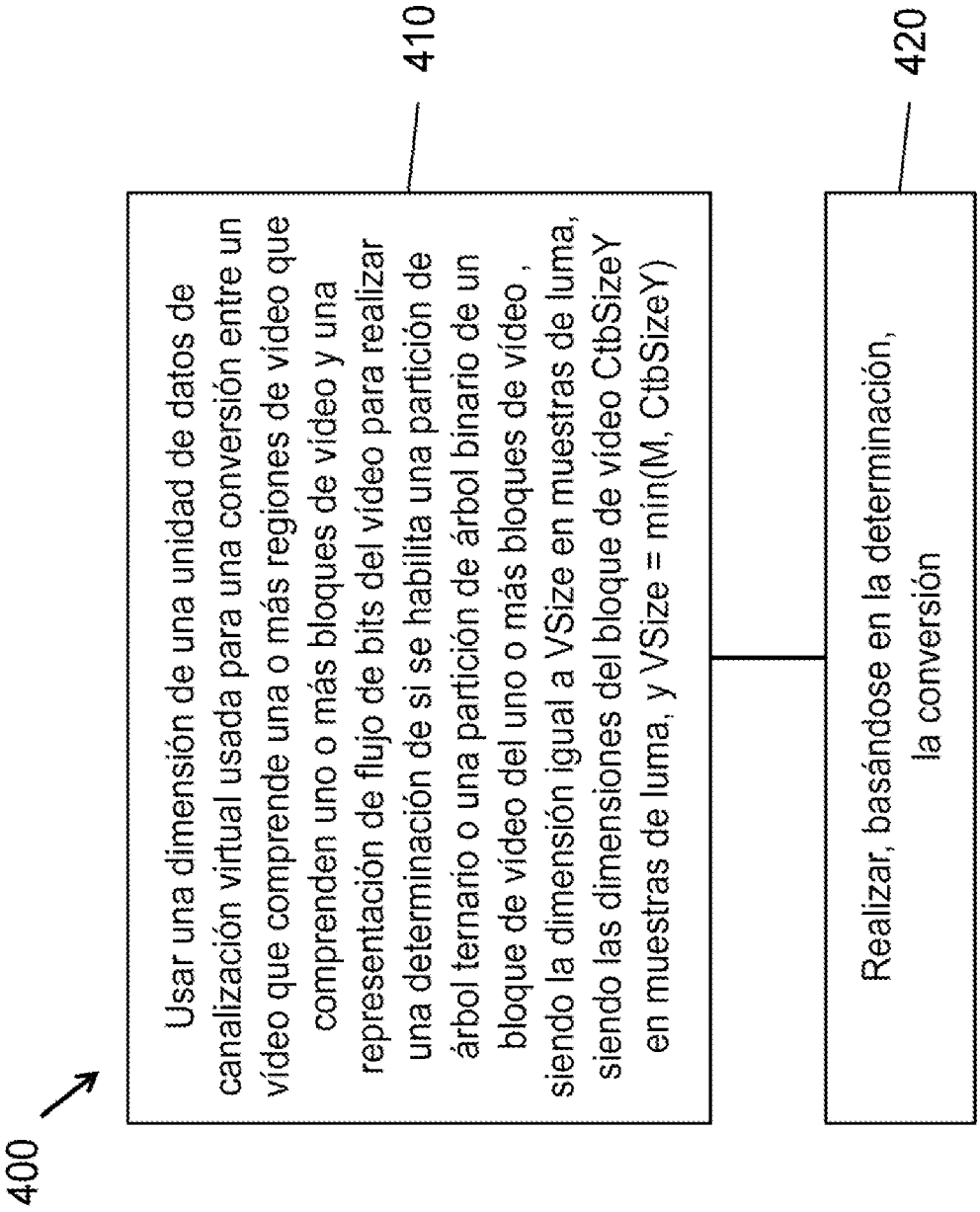

400
Usar una dimensión de una unidad de datos de canalización virtual usada para una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo y una representación de flujo de bits del vídeo para realizar una determinación de si se habilita una partición de árbol ternario o una partición de árbol binario de un bloque de vídeo del uno o más bloques de vídeo , siendo la dimensión igual a VSize en muestras de luma, siendo las dimensiones del bloque de vídeo CtbSizeY en muestras de luma, y VSize = min(M, CtbSizeY)
410
Realizar, basándose en la determinación, la conversión
420

FIG. 4

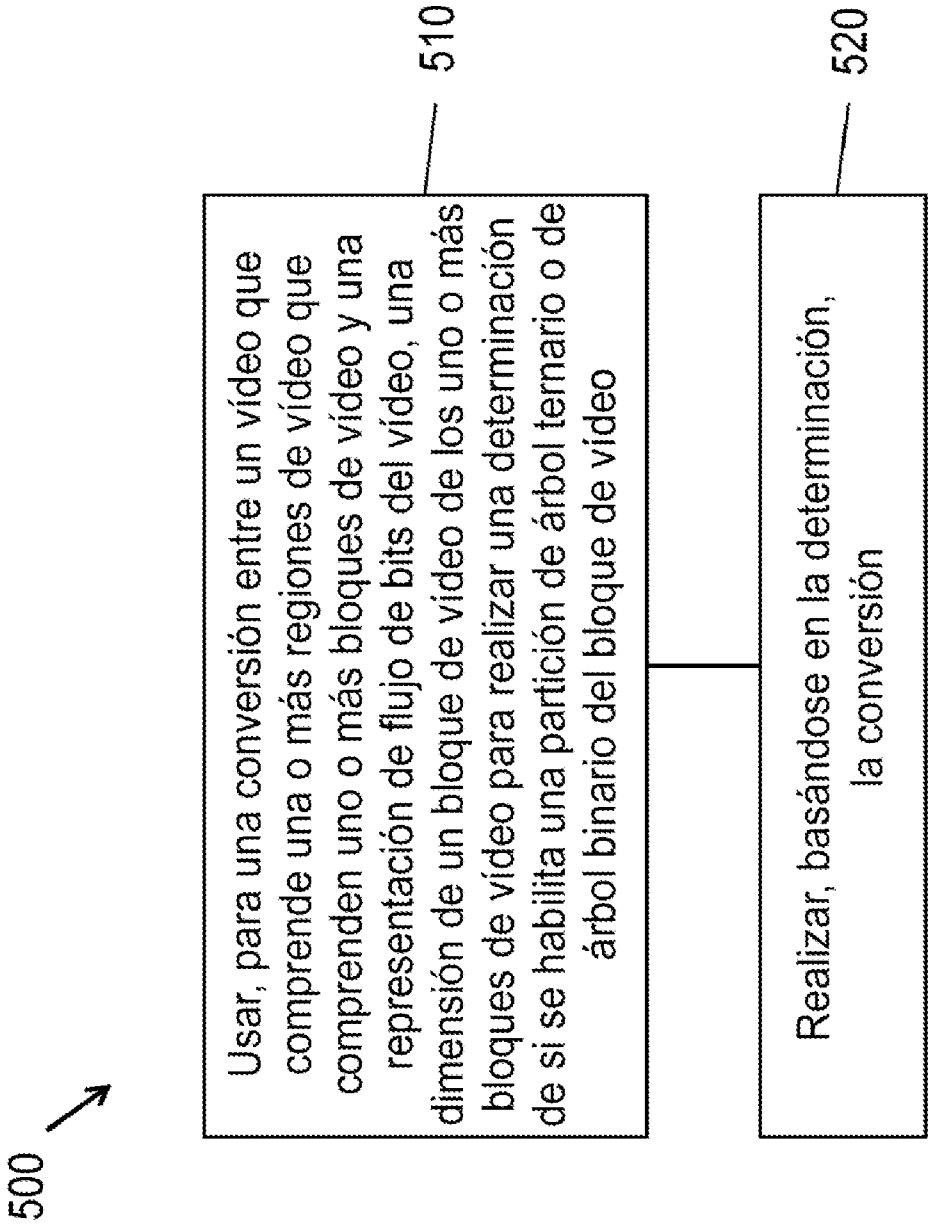

500
Usar, para una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo y una representación de flujo de bits del vídeo, una dimensión de un bloque de vídeo de los uno o más bloques de vídeo para realizar una determinación de si se habilita una partición de árbol ternario o de árbol binario del bloque de vídeo
510
Realizar, basándose en la determinación, la conversión
520

FIG. 5

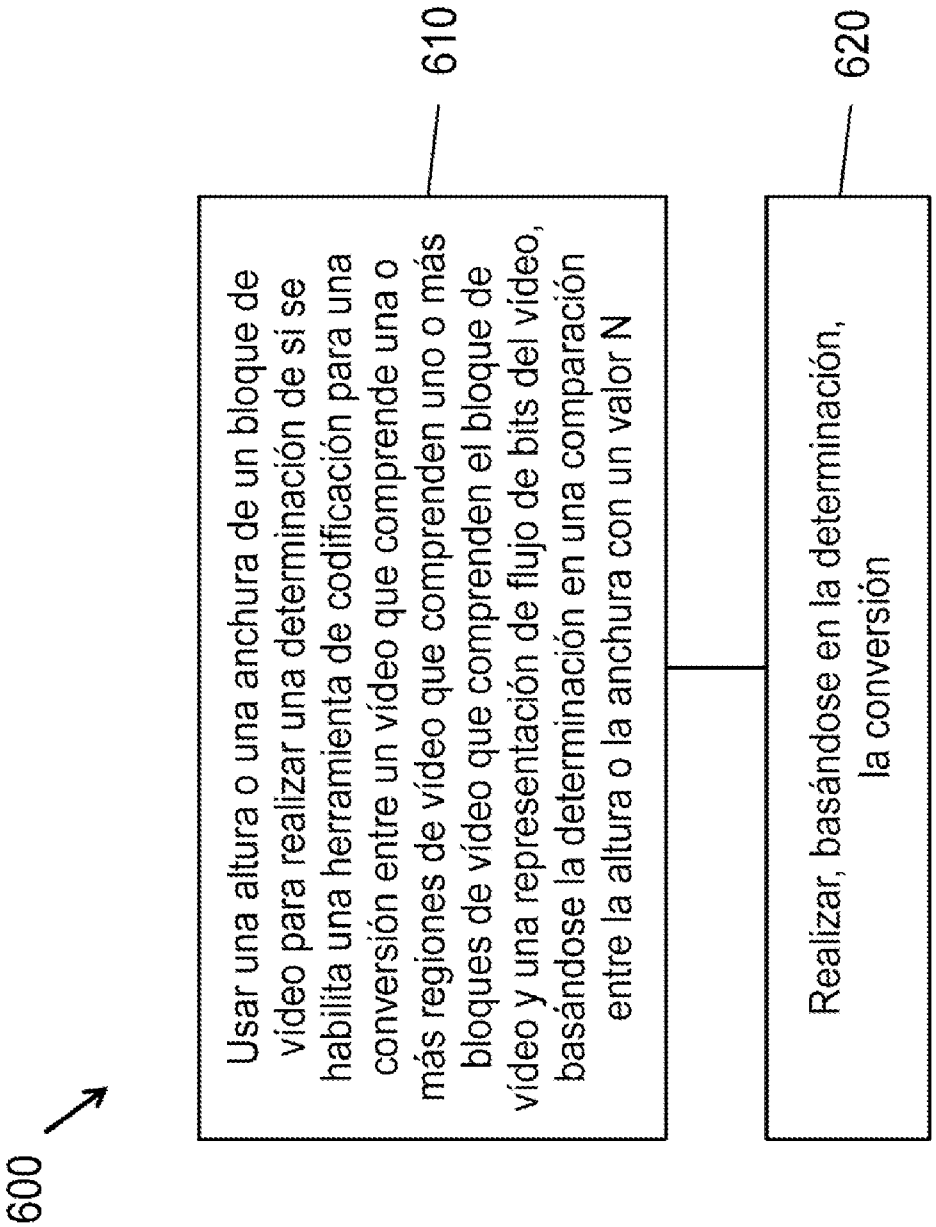

600
610
Usar una altura o una anchura de un bloque de vídeo para realizar una determinación de si se habilita una herramienta de codificación para una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo que comprenden el bloque de vídeo y una representación de flujo de bits del vídeo, basándose la determinación en una comparación entre la altura o la anchura con un valor N
620
Realizar, basándose en la determinación, la conversión

FIG. 6

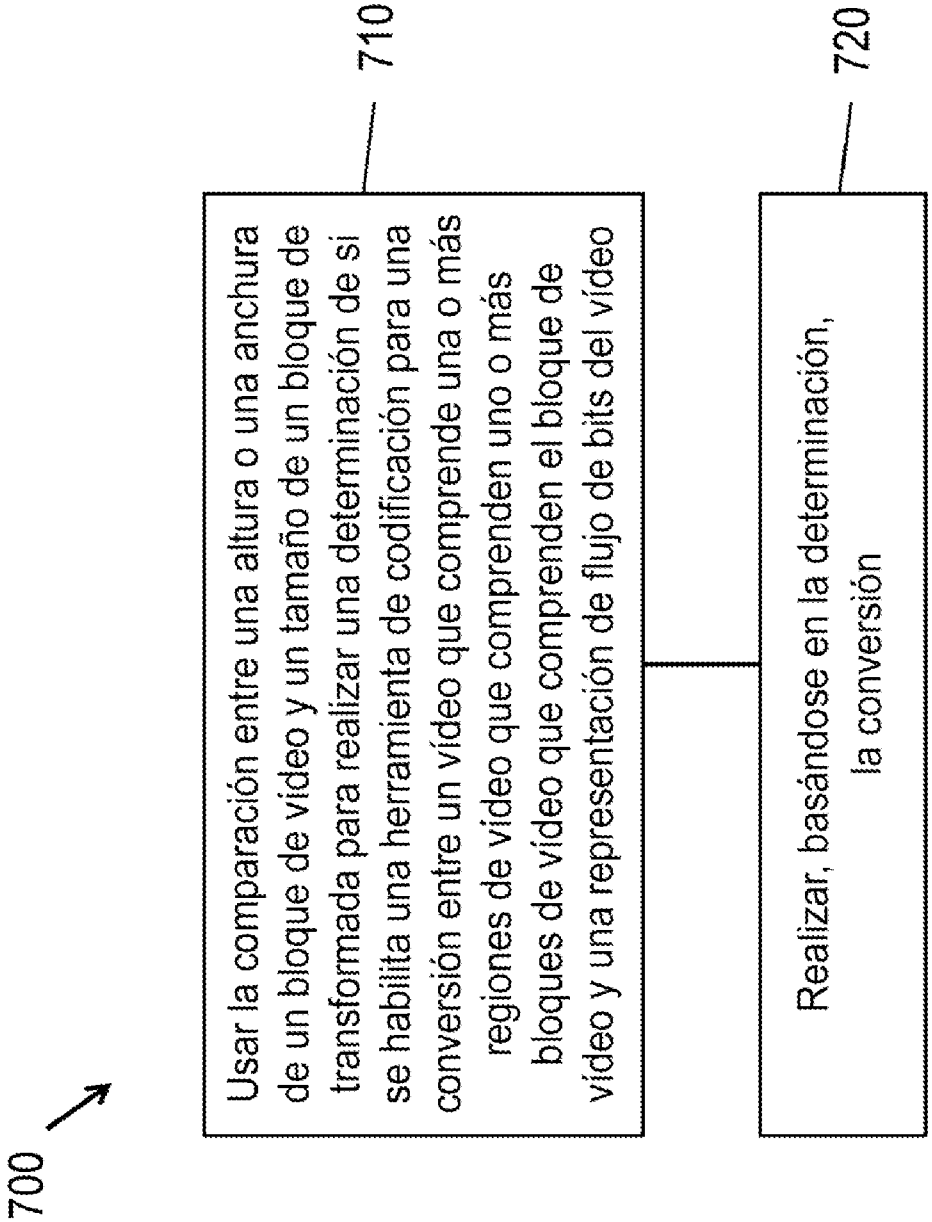

700
710
Usar la comparación entre una altura o una anchura de un bloque de vídeo y un tamaño de un bloque de transformada para realizar una determinación de si se habilita una herramienta de codificación para una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo que comprenden el bloque de vídeo y una representación de flujo de bits del vídeo
720
Realizar, basándose en la determinación, la conversión

FIG. 7

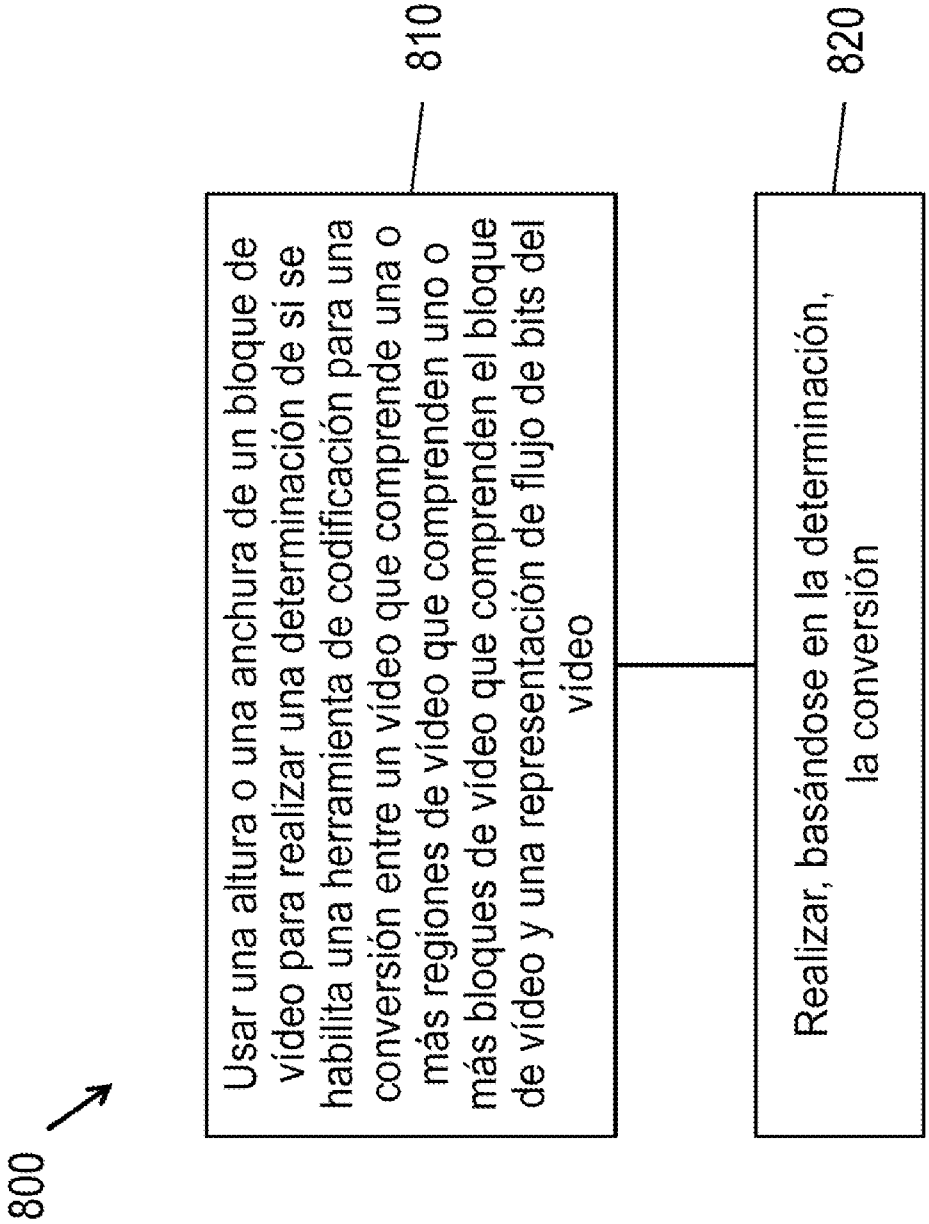

800
Usar una altura o una anchura de un bloque de vídeo para realizar una determinación de si se habilita una herramienta de codificación para una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo que comprenden el bloque de vídeo y una representación de flujo de bits del vídeo
810
Realizar, basándose en la determinación, la conversión
820

FIG. 8

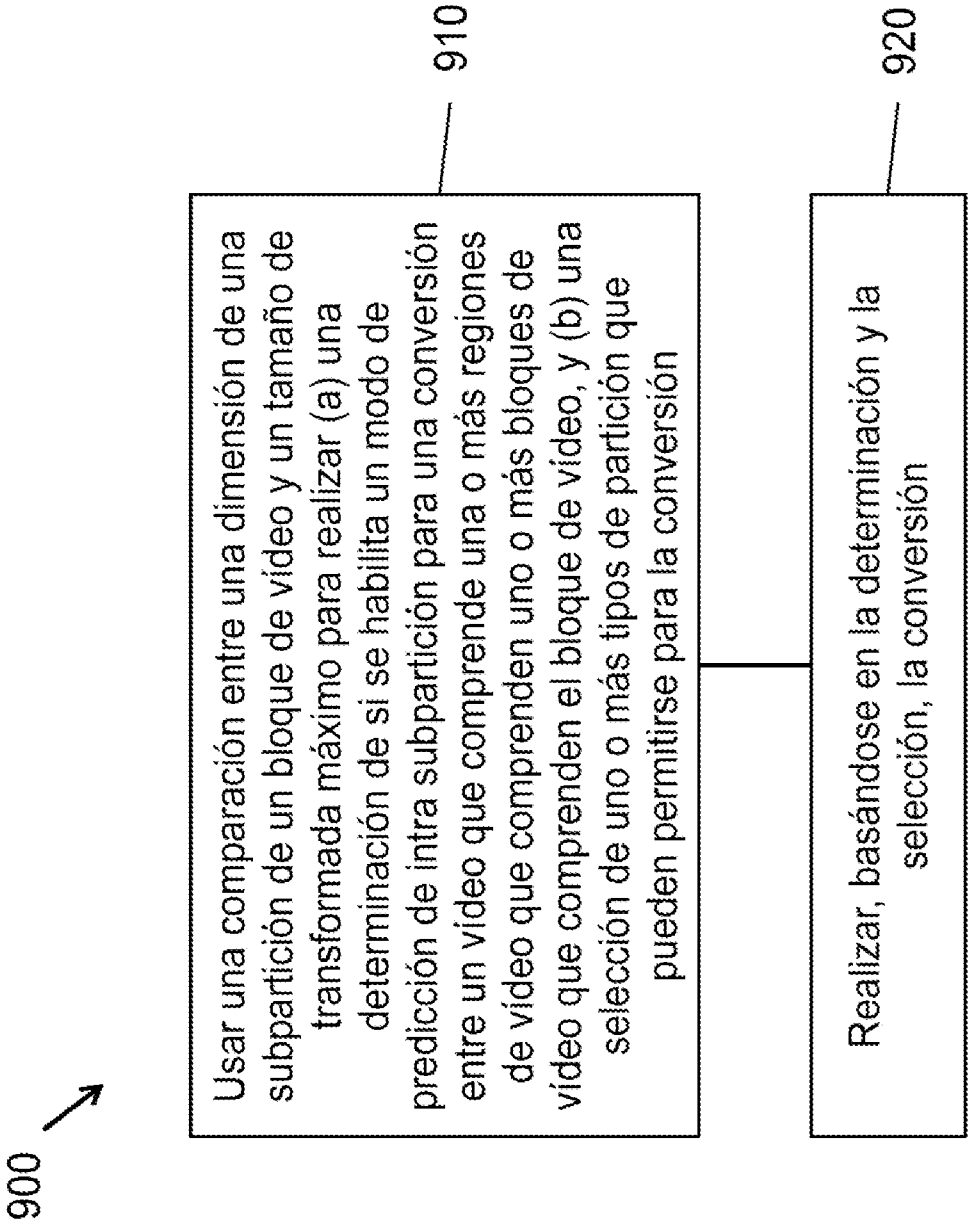

900
Usar una comparación entre una dimensión de una subpartición de un bloque de vídeo y un tamaño de transformada máximo para realizar (a) una determinación de si se habilita un modo de predicción de intra subpartición para una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo que comprenden el bloque de vídeo, y (b) una selección de uno o más tipos de partición que pueden permitirse para la conversión
910
Realizar, basándose en la determinación y la selección, la conversión
920

FIG. 9

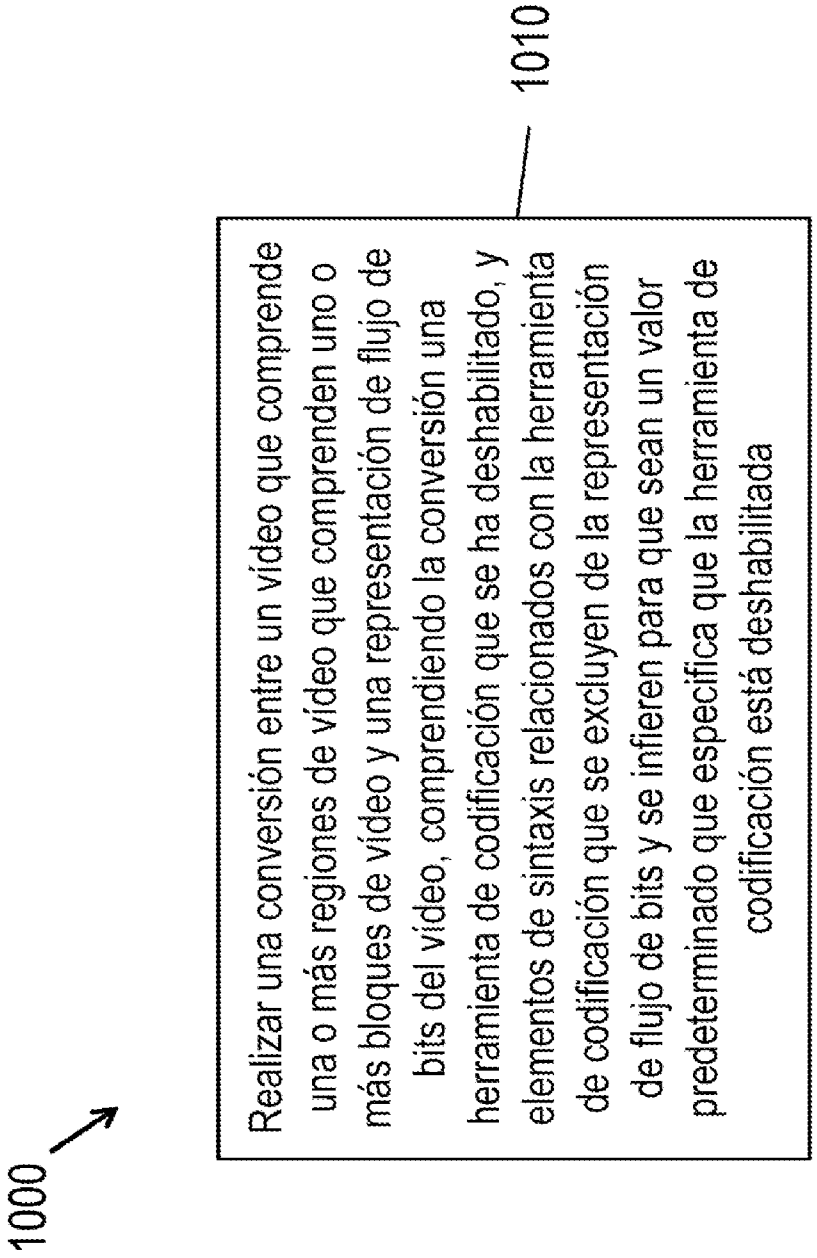

1000
1010
Realizar una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo y una representación de flujo de bits del vídeo, comprendiendo la conversión una herramienta de codificación que se ha deshabilitado, y elementos de sintaxis relacionados con la herramienta de codificación que se excluyen de la representación de flujo de bits y se infieren para que sean un valor predeterminado que especifica que la herramienta de codificación está deshabilitada

FIG. 10

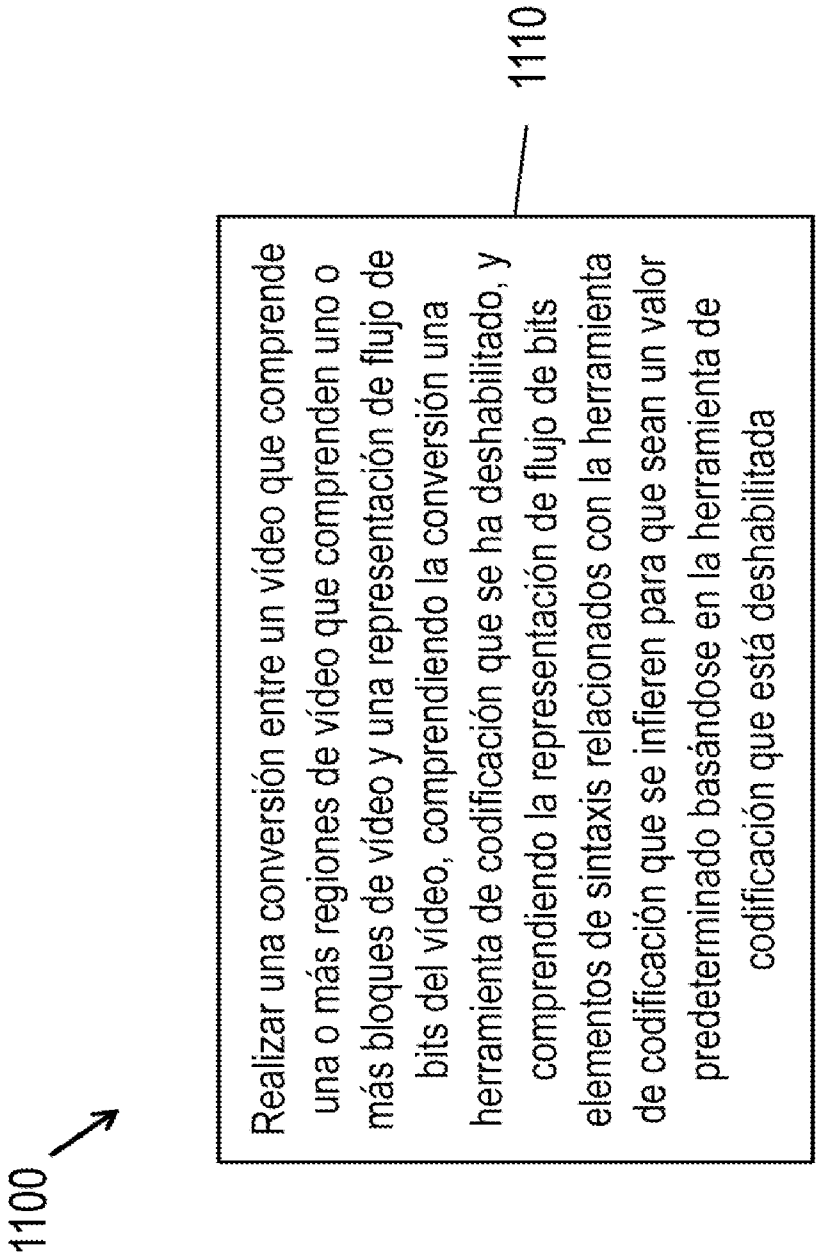

1100
1110
Realizar una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo y una representación de flujo de bits del vídeo, comprendiendo la conversión una herramienta de codificación que se ha deshabilitado, y comprendiendo la representación de flujo de bits elementos de sintaxis relacionados con la herramienta de codificación que se infieren para que sean un valor predeterminado basándose en la herramienta de codificación que está deshabilitada

FIG. 11

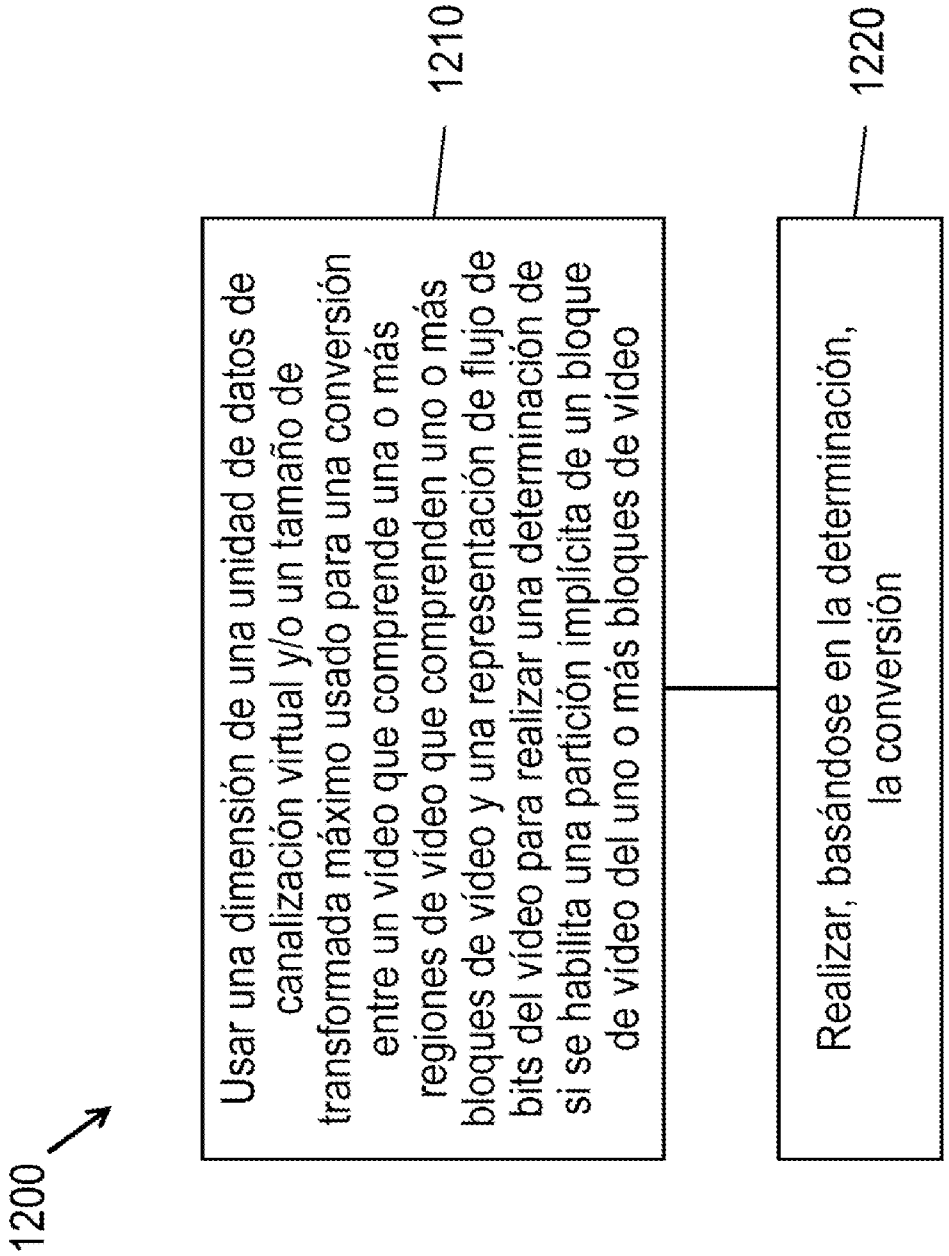

1200
Usar una dimensión de una unidad de datos de canalización virtual y/o un tamaño de transformada máximo usado para una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo y una representación de flujo de bits del vídeo para realizar una determinación de si se habilita una partición implícita de un bloque de vídeo del uno o más bloques de vídeo
1210
Realizar, basándose en la determinación, la conversión
1220

FIG. 12

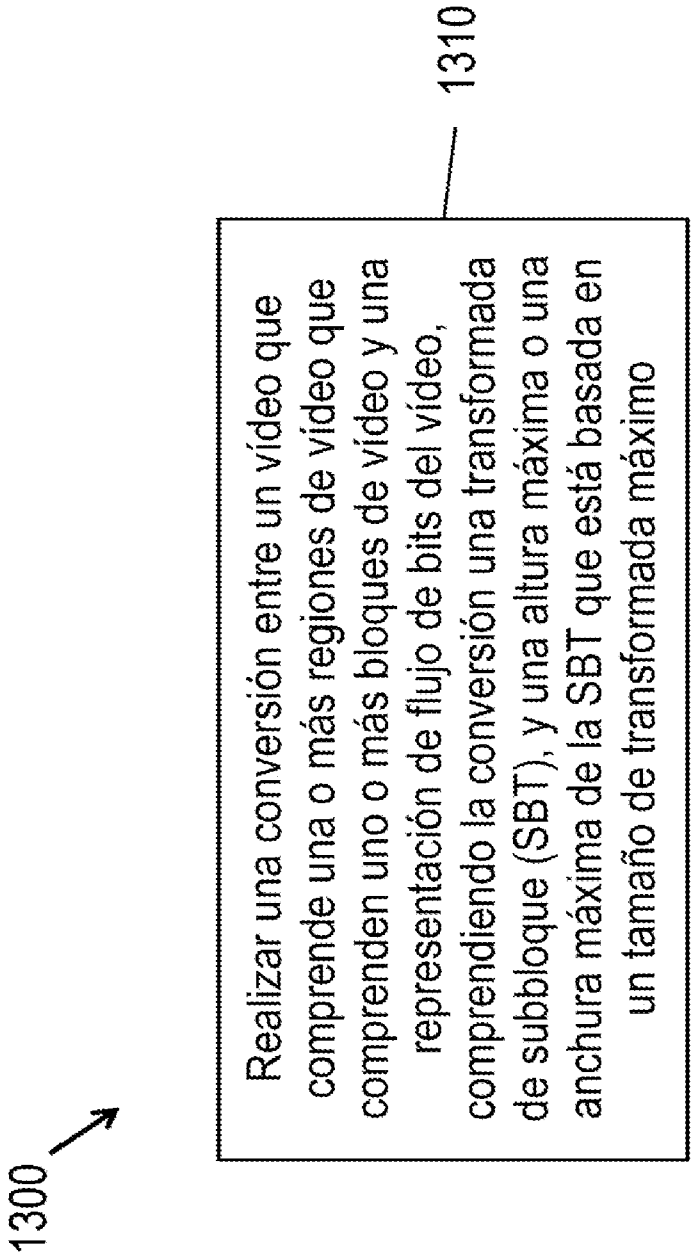

1300
1310
Realizar una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo y una representación de flujo de bits del vídeo, comprendiendo la conversión una transformada de subbloque (SBT), y una altura máxima o una anchura máxima de la SBT que está basada en un tamaño de transformada máximo

FIG. 13

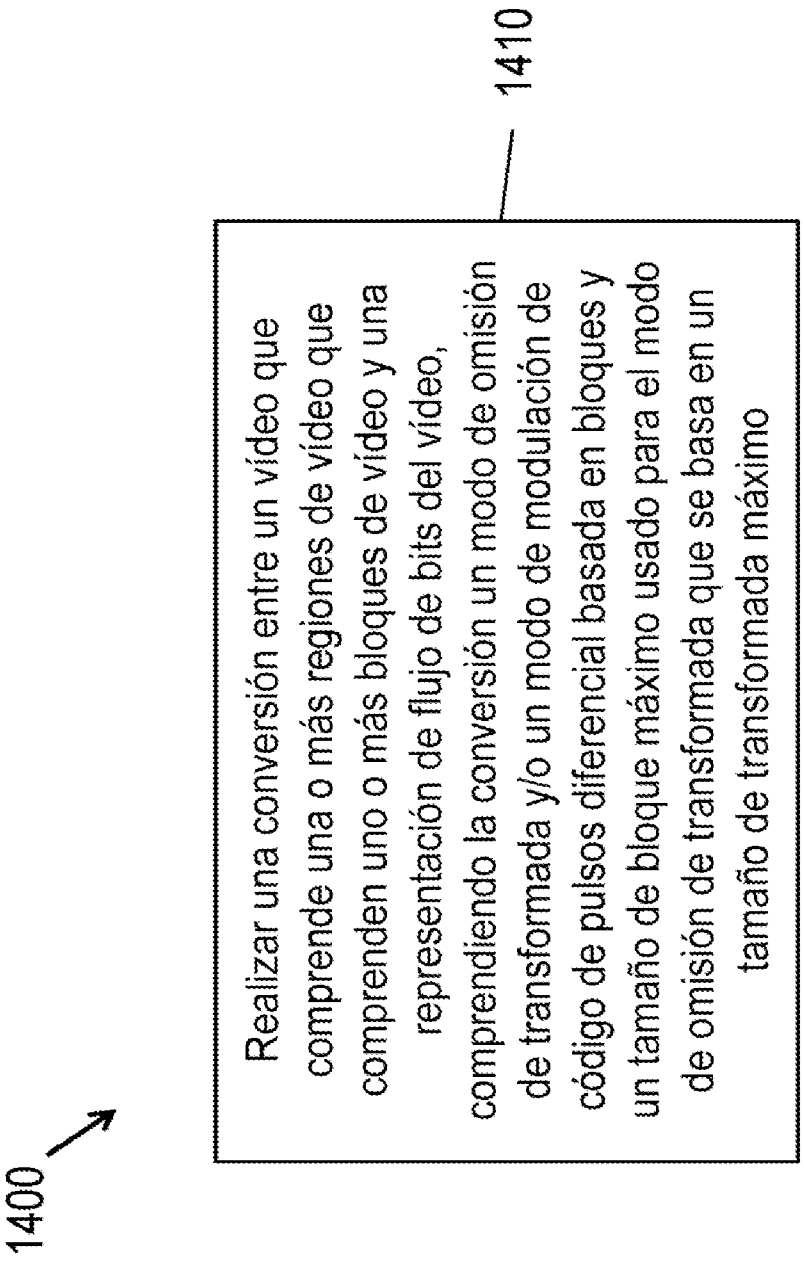

1400
1410
Realizar una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo y una representación de flujo de bits del vídeo, comprendiendo la conversión un modo de omisión de transformada y/o un modo de modulación de código de pulsos diferencial basada en bloques y un tamaño de bloque máximo usado para el modo de omisión de transformada que se basa en un tamaño de transformada máximo

FIG. 14

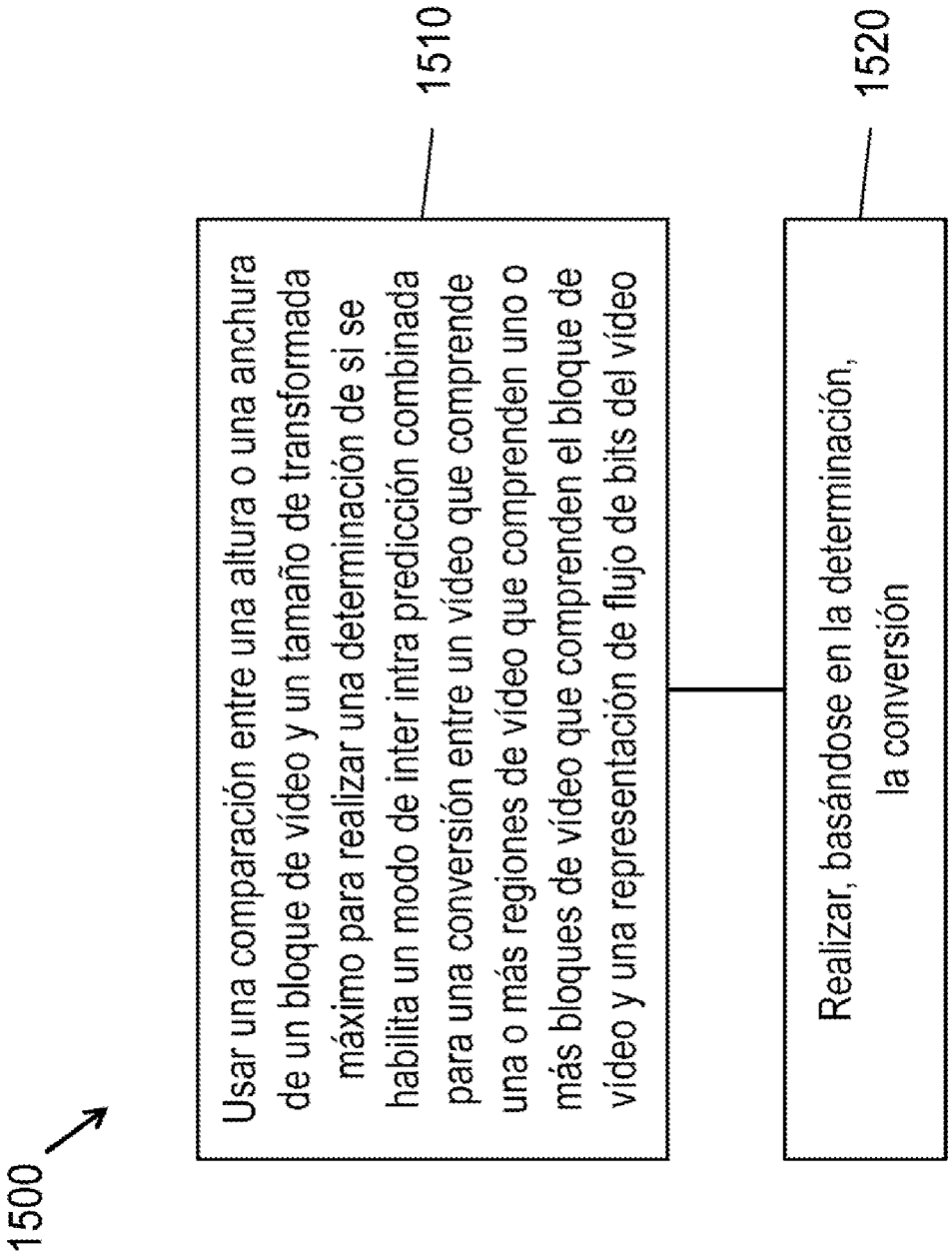

1500
1510
Usar una comparación entre una altura o una anchura de un bloque de vídeo y un tamaño de transformada máximo para realizar una determinación de si se habilita un modo de inter intra predicción combinada para una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo que comprenden el bloque de vídeo y una representación de flujo de bits del vídeo
1520
Realizar, basándose en la determinación, la conversión

FIG. 15

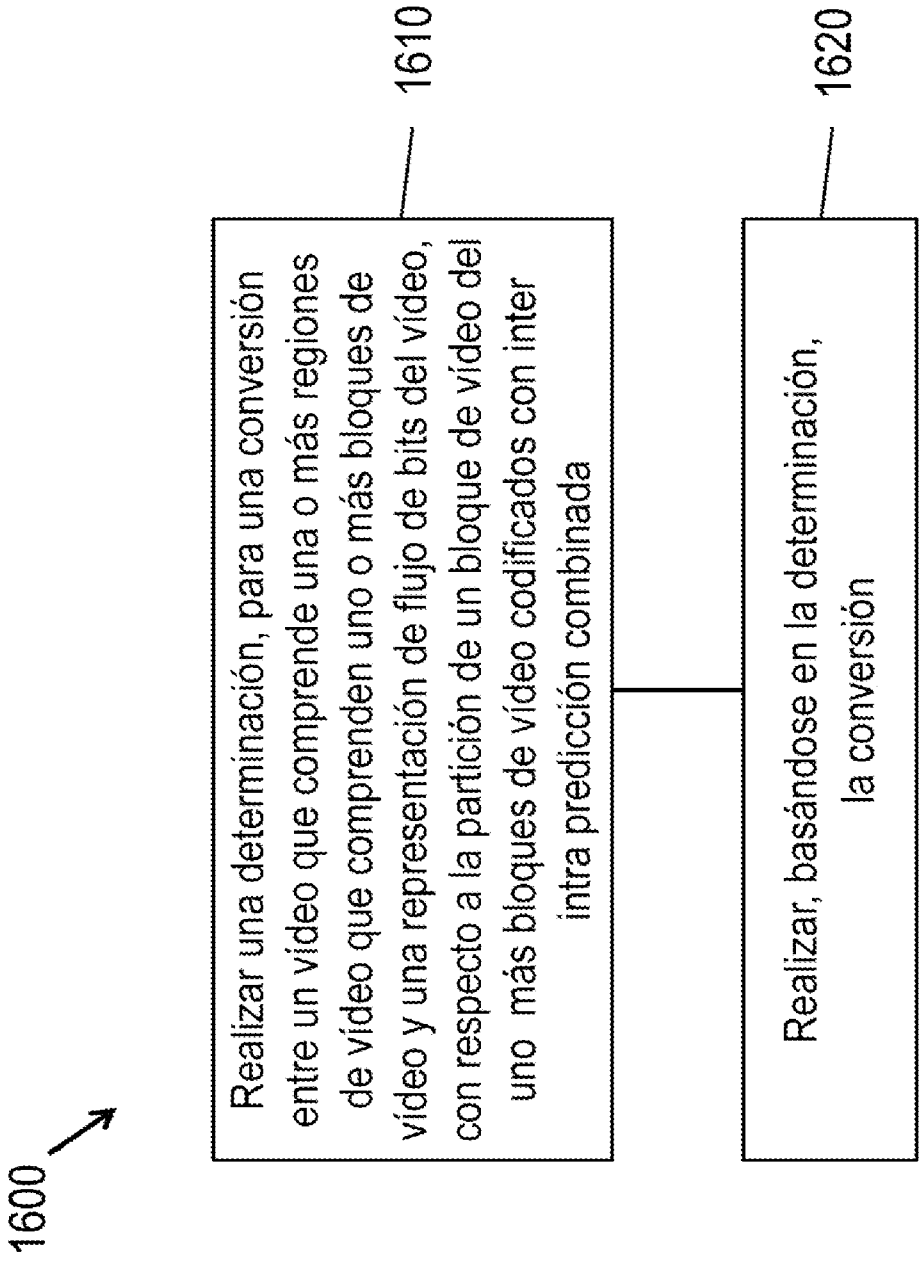

1600
Realizar una determinación, para una conversión entre un vídeo que comprende una o más regiones de vídeo que comprenden uno o más bloques de vídeo y una representación de flujo de bits del vídeo, con respecto a la partición de un bloque de vídeo del uno más bloques de vídeo codificados con inter intra predicción combinada
1610
Realizar, basándose en la determinación, la conversión
1620

FIG. 16

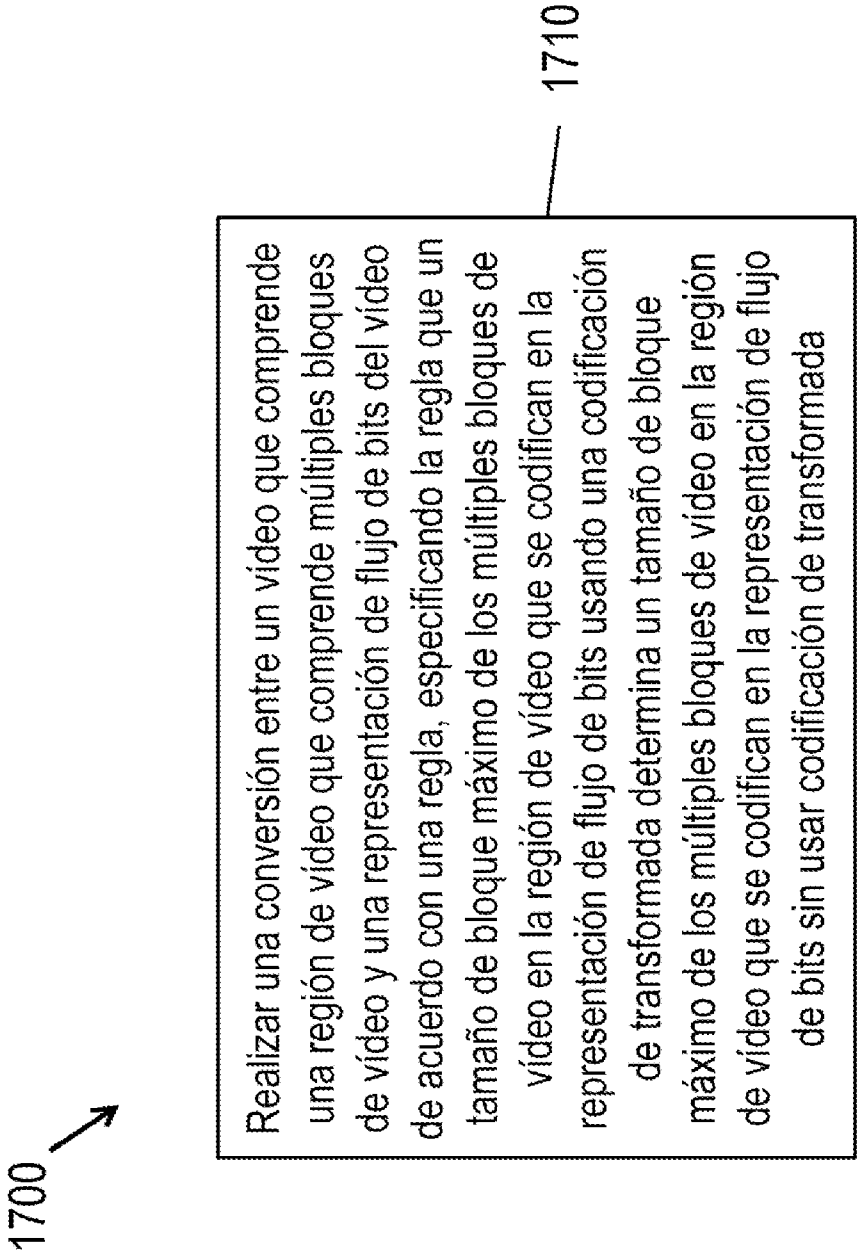

1700
1710
Realizar una conversión entre un vídeo que comprende una región de vídeo que comprende múltiples bloques de vídeo y una representación de flujo de bits del vídeo de acuerdo con una regla, especificando la regla que un tamaño de bloque máximo de los múltiples bloques de vídeo en la región de vídeo que se codifican en la representación de flujo de bits usando una codificación de transformada determina un tamaño de bloque máximo de los múltiples bloques de vídeo en la región de vídeo que se codifican en la representación de flujo de bits sin usar codificación de transformada

FIG. 17

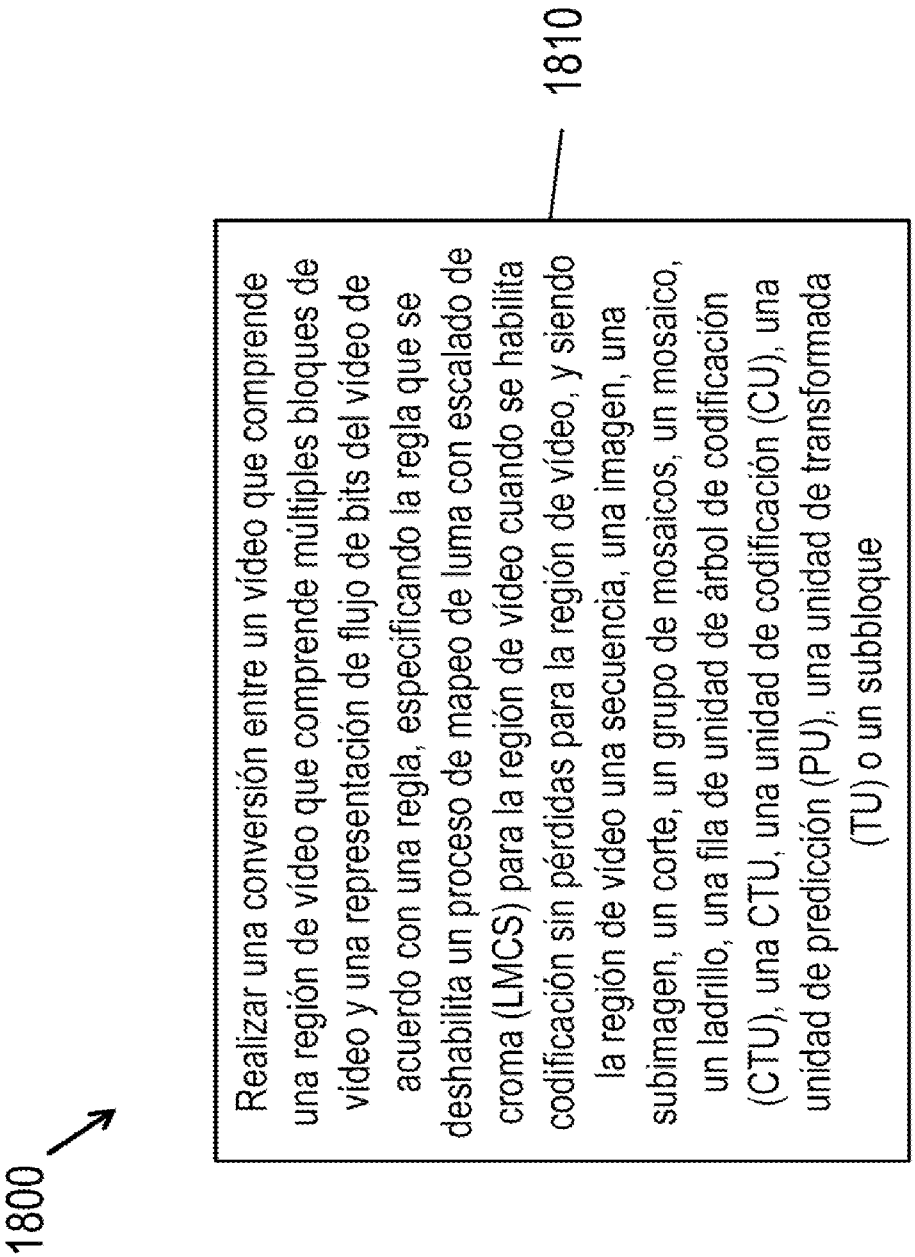

1800
1810
Realizar una conversión entre un vídeo que comprende una región de vídeo que comprende múltiples bloques de vídeo y una representación de flujo de bits del vídeo de acuerdo con una regla, especificando la regla que se deshabilita un proceso de mapeo de luma con escalado de croma (LMCS) para la región de vídeo cuando se habilita codificación sin pérdidas para la región de vídeo, y siendo la región de vídeo una secuencia, una imagen, una subimagen, un corte, un grupo de mosaicos, un mosaico, un ladrillo, una fila de unidad de árbol de codificación (CTU), una CTU, una unidad de codificación (CU), una unidad de predicción (PU), una unidad de transformada (TU) o un subbloque

FIG. 18