(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 789 024**

(51) Int. CI.:

H04N 19/70 (2014.01)
H04N 19/174 (2014.01)
H04N 19/13 (2014.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(96) Fecha de presentación y número de la solicitud europea: **21.12.2012 E 16161035 (7)**

(97) Fecha y número de publicación de la concesión europea: **05.02.2020 EP 3051824**

(54) Título: **Gestión de datos de extensión**

(30) Prioridad:

12.04.2012 US 201261623290 P

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
23.10.2020

(73) Titular/es:

VELOS MEDIA INTERNATIONAL LIMITED (100.0%)
Unit 32, the Hyde Building The Park, Carrickmines
Dublin 18, IE

(72) Inventor/es:

SAMUELSSON, JONATAN y
SJÖBERG, RICKARD

(74) Agente/Representante:

MILTENYI , Peter

DESCRIPCIÓN

Gestión de datos de extensión

Campo técnico

Las realizaciones se refieren en general a la codificación y la decodificación de segmentos (*slices*) y, en particular, sobre la gestión de datos de extensión en conexión con la codificación y la decodificación de segmentos.

Antecedentes

La codificación de vídeo de alta eficiencia (*High Efficiency Video Coding*, HEVC) es un nuevo estándar de codificación de vídeo que en la actualidad está siendo desarrollado en el *Joint Collaborative Team on Video Coding* (JCT-VC). JCT-VC es un proyecto de colaboración entre el Grupo de Expertos de Imágenes en Movimiento (MPEG) y el sector de estandarización de telecomunicaciones de la Unión Internacional de Telecomunicaciones (ITU-T). En la actualidad hay definido un borrador de comité (*Commitee Draft*, CD) que incluye varias nuevas herramientas y es considerablemente más eficiente que la Codificación Avanzada de Vídeo (*Advanced Video Coding*, AVC) H.264.

Es probable que en el futuro se definan extensiones de HEVC, tales como una extensión de vistas múltiples y una extensión de redimensionamiento. Estas extensiones podrían entonces contener datos relativos, por ejemplo, a vistas adicionales dependientes de una vista base, o capas adicionales dependientes de una capa base.

Puede asumirse que la retrocompatibilidad es un requisito para algunas partes de una secuencia codificada, por ejemplo, la vista base o la capa base. Sin embargo, es importante que un decodificador preexistente *(legacy decoder)*, es decir, compatible con la especificación base, sea capaz de gestionar correctamente el flujo de bits incluso cuando se usen tales extensiones.

La HEVC define algunos tipos de unidades de la Capa de Abstracción de Red (*Network Abstraction Layer*, NAL) que han de reservarse, entre otros, para extensiones. También los conjuntos de parámetros, por ejemplo el Conjunto de Parámetros de Secuencia (*Sequence Parameter Set*, SPS), el Conjunto de Parámetros de Imagen (*Picture Parameter Set*, PPS) y el Conjunto de Parámetros de Adaptación (*Adaptation Parameter Set*, APS), contienen campos de extensión al final de sus tablas de sintaxis. Se especifica que un decodificador que se conforme a la especificación base de la HEVC ignorará los datos del campo de extensión al final de un conjunto de parámetros. La sintaxis para los datos de extensión es mostrada a continuación en un ejemplo ilustrativo en forma de SPS como conjunto de parámetros. La sintaxis sería similar si se usaran otros conjuntos de parámetros distintos de SPS. La longitud total de un conjunto de parámetros es conocida por el decodificador por medio del elemento de sintaxis *NumBytesInRBSP*, normalmente proporcionado por la capa de sistema, para que el decodificador no tenga ningún problema en detectar dónde acaba el campo de extensión.

seq_parameter_set_rbsp( ) {	**Descriptor**
...	
sps_extension_flag	u(1)
if(*sps_extension_flag*)	
while(*more_rbsp_data( )*)	
sps_extension_data_flag	u(1)
rbsp_trailing_bits( )	
}	

Una unidad NAL que contiene un segmento (*slice*) codificado de una imagen consiste en dos partes: en primer lugar, la cabecera de segmento y luego los datos de segmento. En la actualidad no hay ningún campo de extensión para segmentos codificados.

La cabecera de segmento contiene información sobre el segmento actual, por ejemplo el tipo de segmento, qué imágenes de referencia usar, etc. Existe la necesidad de añadir extensiones y campos

de extensión en HEVC. Sin embargo, en tal caso es importante que tanto un decodificador preexistente que se conforme a la especificación base —es decir, que no use extensiones— como un decodificador compatible con las extensiones puedan gestionar correctamente un flujo de bits (*bitstream*) que comprenda tales extensiones.

El actual diseño de una cabecera de segmento para una representación codificada de un segmento hace esta necesidad difícil o incluso imposible de implementar cuando se tiene decodificadores tanto preexistentes como compatibles con extensiones.

Boyce et al., Extensible High Layer Syntax for Scalability, JCT-VC of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, 5º Encuentro: Ginebra, CH, 16-23 de marzo de 2011, Documento: JCTVC-E279, propone la introducción de un nuevo conjunto de parámetros de dependencia y de dos nuevos mensajes SEI, así como cambios en la cabecera de la unidad NAL, SPS, PPS y la cabecera de segmento para un mejor soporte del redimensionamiento y las extensiones futuras.

Sumario

Es un objetivo general habilitar el uso de extensiones de cabecera de segmento.

Es un objetivo particular gestionar extensiones de cabecera de segmento en conexión con la codificación y la decodificación de segmentos.

Estos y otros objetivos son satisfechos por las realizaciones dadas a conocer en la presente memoria.

Según un primer aspecto se proporciona un receptor. El receptor comprende una unidad de entrada configurada para recibir una representación codificada de un segmento de una imagen y un decodificador para decodificar la representación codificada. Dicha representación codificada comprende una cabecera de segmento y datos de segmento. Dicho decodificador comprende una unidad de identificación de conjuntos configurada para identificar un conjunto de parámetros asociado con dicha representación codificada en base a un identificador del conjunto de parámetros obtenible en función de dicha cabecera de segmento; una unidad de análisis configurada para analizar una bandera de presencia de extensión en la cabecera de segmento presente en dicho conjunto de parámetros y analizar, si dicha bandera de presencia de extensión en la cabecera de segmento tiene un valor predefinido, un indicador de longitud en un código de longitud variable asociado con dicha representación codificada e indicativo de una longitud en bits o bytes de un campo de extensión en dicha cabecera de segmento; y una unidad de procesamiento configurada para hacer que, si dicha bandera de presencia de extensión en la cabecera de segmento tiene dicho valor predefinido, dicho decodificador ignore, durante la decodificación de dicha representación codificada, cualquier valor de dicho campo de extensión identificado en función de dicho indicador de longitud.

Según un segundo aspecto se proporciona un transmisor. El transmisor comprende un codificador para codificar un segmento de una imagen. Dicho codificador comprende una unidad de configuración que está configurada para poner una bandera de presencia de extensión en la cabecera de segmento en un valor predefinido para indicar la presencia de un campo de extensión en una cabecera de segmento de dicho segmento; una unidad de generación configurada para generar un indicador de longitud en un código de longitud variable indicativo de una longitud en bits o bytes de dicho campo de extensión presente en dicha cabecera de segmento de dicho segmento; una unidad de codificación configurada para codificar dicho segmento creando una representación codificada de dicho segmento que comprende dicha cabecera de segmento y datos de segmento; una unidad de asociación configurada para asociar dicho indicador de longitud con dicha representación codificada, insertar dicha bandera de presencia de extensión en la cabecera de segmento en un conjunto de parámetros y asociar un identificador del conjunto de parámetros que identifica dicho conjunto de parámetros con dicha representación codificada, y una unidad de salida para dar salida a la imagen codificada.

Las presentes realizaciones habilitan la introducción de campos de extensión en cabeceras de segmento de representaciones codificadas de segmentos y seguir produciendo tales representaciones codificadas que podrían ser gestionadas correctamente incluso por decodificadores preexistentes que no sean compatibles con campos de extensión.

Breve descripción de los dibujos

La invención es definida por las reivindicaciones adjuntas. En las realizaciones de las Figuras 16 y 19 se encuentra una divulgación habilitante para la invención. El resto de realizaciones se deben entender como ejemplos que no caen dentro del alcance de la presente invención.

La invención, junto con objetos y ventajas adicionales de la misma, puede ser entendida de manera óptima haciendo referencia a la siguiente descripción tomada junto con los dibujos adjuntos, en los que:

La Fig. 1 es una ilustración esquemática de un flujo de vídeo de imágenes que comprenden uno o más segmentos;

La Fig. 2 es una ilustración de un paquete de datos que comprende una unidad NAL;

Las Figuras 3A-3C son ilustraciones de representaciones codificadas de un segmento;

La Fig. 4 es un diagrama de flujo de un método de decodificación de una representación codificada de un segmento;

La Fig. 5 es un diagrama de flujo que ilustra etapas adicionales opcionales del método de la Fig. 4;

La Fig. 6 es un diagrama de flujo que ilustra etapas adicionales opcionales del método de la Fig. 4;

La Fig. 7 es un diagrama de flujo que ilustra etapas adicionales opcionales del método de la Fig. 4;

La Fig. 8 es un diagrama de flujo de un método de decodificación de una representación codificada de un segmento;

La Fig. 9 es un diagrama de flujo de un método de codificación de un segmento;

La Fig. 10 es un diagrama de flujo que ilustra etapas adicionales opcionales del método de la Fig. 9;

La Fig. 11 es un diagrama de flujo que ilustra etapas adicionales opcionales del método de la Fig. 9;

La Fig. 12 es un diagrama de flujo que ilustra un ejemplo de la etapa de asociación de la Fig. 11;

La Fig. 13 es un diagrama esquemático de bloques de un decodificador;

La Fig. 14 es un diagrama esquemático de bloques de un decodificador;

La Fig. 15 es un diagrama esquemático de bloques de un decodificador;

La Fig. 16 es un diagrama esquemático de bloques de un receptor según una realización;

La Fig. 17 es un diagrama esquemático de bloques de un codificador;

La Fig. 18 es un diagrama esquemático de bloques de un codificador;

La Fig. 19 es un diagrama esquemático de bloques de un transmisor según una realización; y

La Fig. 20 es un diagrama esquemático de bloques de un ordenador que comprende un medio legible en ordenador y un programa informático.

Descripción detallada

En todos los dibujos se usan los mismos números de referencia para elementos similares o correspondientes.

Las presentes realizaciones están dirigidas hacia la gestión de datos de extensión en conexión con la codificación y la decodificación de segmentos de imágenes. Por ende, las realizaciones habilitan la introducción y el uso de lo que ha dado en llamarse campos de extensión, también denominados elementos de sintaxis de extensión, en las cabeceras de segmento para los segmentos. Tales campos de extensión pueden ser añadidos, según las realizaciones, a las cabeceras de segmento y seguir presentando una representación codificada que podría ser gestionada correctamente tanto por decodificadores compatibles con extensiones como con los decodificadores que no pueden gestionar datos de extensión, es decir, que se atienen a un perfil que no usa campos de extensión. Estos decodificadores son generalmente denominados decodificadores preexistentes o decodificadores compatibles con la especificación base.

La presencia de decodificadores tanto compatibles con las extensiones como preexistentes impone problemas en conexión con el uso de campos de extensión en las cabeceras de segmento. Por ende, es preciso que un decodificador preexistente sea capaz de decodificar e interpretar correctamente los datos de un representación codificada de un segmento aunque la representación codificada transporte datos de extensión en su cabecera de segmento.

No es posible añadir el mismo tipo de campo de extensión que en los conjuntos de parámetros expuestos en la sección de antecedentes, en los que todos los datos adicionales al fin de la estructura de sintaxis son ignorados por el decodificador. El razón es que el decodificador no tiene ninguna forma de identificar dónde acaba la cabecera de segmento y comienzan los datos de segmento. Así, no se conoce la longitud de las estructuras individuales de datos —es decir, la cabecera de segmento y los datos de segmento—; únicamente la longitud total por medio del elemento de sintaxis *NumBytesInRBSP.* Así, generalmente la sintaxis de la carga útil de secuencia de bytes en bruto (*Raw Byte Sequence Payload*, RBSP) de la capa de segmentos tiene este aspecto:

```
slice_layer_rbsp( ) {
        slice_header( )

    slice_data( )
    rbsp_slice_trailing_bits( )
}
```

Así, no es posible añadir un simple campo de extensión en el que todos los datos adicionales del final de la cabecera de segmento sea ignorados por un decodificador preexistente, dado que no habría ninguna forma de que el decodificador preexistente identificara dónde acaba la cabecera de segmento y comienzan los datos de segmento.

Se propone asociar un indicador de longitud con la representación codificada del segmento y que este indicador de longitud sea indicativo de una longitud de un campo de extensión presente en la cabecera de segmento del segmento. Por ende, el indicador de longitud permite que un decodificador identifique la porción de la representación codificada del segmento y la cabecera de segmento que constituye el campo de extensión. Así, por lo tanto, el decodificador preexistente podría determinar qué porción de la representación codificada ignorar durante la decodificación, dado que contiene datos de extensión que el decodificador preexistente no puede tratar.

Sin embargo, mientras que un decodificador preexistente debería ignorar los datos de extensión durante la decodificación, un decodificador compatible con extensiones es capaz de decodificar y usar estos datos de extensión durante la decodificación.

La Fig. 1 es una ilustración esquemática de una secuencia o flujo de vídeo 1 de imágenes 2. Como es bien sabido en la técnica, cada imagen 2 puede comprender una o más de lo que se denomina segmentos 3. Así, cada imagen 2 del flujo de vídeo 1 se divide en uno o más segmentos 3, siendo cada segmento 3 un trozo de la imagen 2 decodificable independientemente. En otras palabras, si se pierde un segmento 3, los otros segmentos 3 de esa imagen 2 siguen siendo decodificables.

La codificación de un segmento 3 genera una representación codificada del segmento 3 que comprende una cabecera de segmento y datos de segmento. La presentación codificada sale del procedimiento de codificación como lo que se denomina unidad 11 de la Capa de Adaptación de Red (NAL), según se muestra en la Fig. 2. La primera parte de la unidad NAL 11 es una cabecera que contiene una indicación del tipo de datos de la unidad NAL. La parte restante de la unidad NAL 11 contiene datos de carga útil en forma de cabecera de segmento y datos de segmento.

Puede entonces añadirse la unidad NAL 11 con cabeceras 12 para formar un paquete de datos 10 que puede ser transmitido como parte de un flujo de bits del codificador al decodificador. A la unidad NAL 11 se le podrían añadir, por ejemplo, cabeceras 12 del Protocolo de Transporte de Tiempo Real (*Real-time Transport Protocol*, RTP), del Protocolo de Datagramas de Usuario (*User Datagram Protocol*, UDP) y del Protocolo de Internet (*Internet Protocol*, IP). Esta forma de paquetización de las unidades NAL 11 constituye meramente un ejemplo en conexión con el transporte de vídeo. Son posibles otros enfoques de gestión de las unidades NAL 11, tales como el formato de ficheros, flujos de transporte MPEG-2, flujos de programa MPEG-2, etc.

Las realizaciones dadas a conocer en la presente memoria son en particular aplicables y están dirigidas hacia la decodificación y la codificación de segmentos 3 de imágenes 2 en un flujo o secuencia de vídeo 1, comprendiendo el flujo o secuencia de vídeo 1 múltiples imágenes 2, según se muestra en la Fig. 1. Por lo tanto, las realizaciones serán expuestas en conexión con tal decodificación y codificación de vídeo.

Sin embargo, las realizaciones también podrían ser aplicadas a la decodificación y la codificación de segmentos en una única imagen, por ejemplo en conexión con tal decodificación y codificación de fotografías o imágenes fijas. Por ejemplo, podrían usarse datos de extensión en conexión con imágenes fijas con redimensionamiento espacial.

La Fig. 4 es un diagrama de flujo que ilustra un método de decodificación de una representación codificada de un segmento de una imagen, opcional pero preferentemente en un flujo de vídeo. Preferentemente, el método de la Fig. 4 es un método realizado por un decodificador durante la decodificación de una representación codificada de un segmento de una imagen o un método de decodificación para una representación codificada de un segmento de una imagen.

Las Figuras 3A-3C ilustran ejemplos no limitantes de tal representación codificada 20 de un segmento. La representación codificada 20 comprende una cabecera de segmento 21 y datos de segmento 25.

El método de la Fig. 4 comprende una etapa S1 en la que se analiza un indicador de longitud 23 (*Length Indicator*, LI) asociado con la representación codificada 20. Este indicador de longitud 23 es indicativo de una longitud de un campo de extensión 24 (*Extension Field*, EXT F) de la cabecera de segmento 21. En una etapa S2 siguiente se determina ignorar, durante la decodificación de la representación codificada 20, cualquier valor del campo de extensión 24 identificado en función del indicador de longitud 23.

Esto quiere decir que el indicador de longitud 23 asociado con la representación codificada 20 puede ser usado para identificar la porción de la representación codificada 20 ocupada por el campo de extensión 24. Por ende, un decodificador puede usar, por ello, el indicador de longitud 23 para determinar o decidir ignorar y, por ello, no usar el o los valores o los datos del campo de extensión 24, estando identificado este campo de extensión 24 en la cabecera de segmento 21 en función del indicador de longitud 23. Así, el decodificador usa, por ello, el indicador de longitud 23 para saltarse una cantidad o porción de datos en la cabecera de segmento 21, estando indicada la cantidad o porción de datos saltada por medio del indicador de longitud 23.

En una realización particular, el método de decodificación de una representación codificada 20 de un segmento de una imagen, tal como en un flujo de vídeo, comprende analizar un indicador de longitud 23 asociado con la representación codificada 20 e indicativo de una longitud de un campo de extensión 24 en la cabecera de segmento 21.

En esta realización particular, el método comprende, además, ignorar, durante la decodificación de la representación codificada 20, cualquier valor del campo de extensión 24 identificado en función del indicador de longitud 23.

El indicador de longitud 23 asociado con la representación codificada 20 también permite que los decodificadores preexistentes traten y decodifiquen la representación codificada 20. Estos decodificadores preexistentes simplemente determinan ignorar el o los valores o los datos obtenidos de la porción de la cabecera de segmento 21 que está ocupada por el campo de extensión 24, identificándose esta porción de la cabecera de segmento 21 en función del indicador analizado de longitud 23.

Según se la usa en la presente memoria, la asociación implica que es posible identificar y recuperar el indicador relevante de longitud 23 en función de la representación codificada 20. Según se divulga adicionalmente en la presente memoria, podría lograrse esto, por ejemplo, incluyendo el indicador de longitud 23 en la representación codificada 20, normalmente en la cabecera de segmento 21 (véanse las Figuras 3A y 3B), o incluyendo datos, tales como un identificador 26, en la representación codificada 20, pudiendo usarse estos datos o identificador 26 para identificar el indicador relevante de longitud 23 (véase la Fig. 3C). En este caso, podría proporcionarse el indicador de longitud 23 en otra estructura de datos distinta de la representación codificada 20. Por lo tanto, los datos o identificador 26 apuntan a esta otra estructura de datos y permiten la identificación de la misma.

La Fig. 5 es un diagrama de flujo que ilustra etapas adicionales del método según una realización particular. Estas etapas adicionales, pero opcionales, comienzan con la etapa S10 que continúa desde la etapa S1 de la Fig. 4. En la etapa S10 se determina la longitud del campo de extensión en función del indicador de longitud 23 analizado en la etapa S1. En una etapa S11 siguiente se identifica el fin del campo de extensión 24 en la cabecera de segmento 21 en función de la longitud determinada en la etapa S10. Por lo tanto, en la etapa S2 siguiente el decodificador puede decidir o determinar ignorar el valor o los valores del campo de extensión 24 hasta el fin del campo de extensión 24 identificado en la etapa S11.

Opcionalmente, también se identifica el comienzo del campo de extensión 24 en la cabecera de segmento 21. En una realización, el comienzo podría estar en una posición fija o predefinida en la cabecera de segmento 21, tal como una posición definida en bits o bytes en la cabecera de segmento 21. Alternativamente. El comienzo del campo de extensión 24 podría ser determinado en función del indicador de longitud 23, tal como siguiendo directamente el indicador de longitud 23 en la cabecera de segmento 21 (véanse las Figuras 3A y 3B). Generalmente, una posición de un elemento de sintaxis en la cabecera de segmento 21 está dada por la longitud de los elementos de sintaxis precedentes. Esto es así en particular si se aplica codificación de longitud variable a uno o más elementos de sintaxis precedentes en la cabecera de segmento 21. En tal caso, el decodificador normalmente empieza a

analizar y decodificar la cabecera de segmento 21 desde la posición del primer bit y pasando por los diversos elementos de sintaxis hasta que alcanza el comienzo del campo de extensión 24.

En una realización, se introduce un indicador de longitud 23 que indica la longitud de datos adicionales de segmento, es decir, el campo de extensión 24. El indicador de longitud 23 está ejemplificado por un elemento de sintaxis en, por ejemplo, la cabecera de segmento 21 u otra estructura de datos, que está asociada con la cabecera de segmento 21, o la longitud puede ser deducida mediante un patrón de bits. En consecuencia, un codificador está configurado para usar el elemento definido 23 de sintaxis para señalar la longitud de los datos adicionales. Un decodificador está configurado para determinar la longitud de los datos adicionales. Los datos adicionales serán ignorados por los decodificadores que se atienen a algunos perfiles, mientras que los decodificadores que se atienen a otros perfiles lo analizarán y usarán los valores del elemento de sintaxis en los datos adicionales. Los decodificadores que se atienen a perfiles que lo ignoran usarán la información de longitud para determinar el fin de la cabecera de segmento 21, de modo que la unidad NAL pueda ser decodificada.

Por lo tanto, las realizaciones habilitan la extensión de una cabecera de segmento 21 de manera simple y eficiente en la tasa de transferencia de bits.

En una realización particular aplicable al método de la Fig. 4 y a la realización mostrada en la Fig. 5, el indicador de longitud 23 es una palabra de código o elemento de sintaxis presente en la cabecera de segmento 21, según se muestra en las Figuras 3A y 3B. Esto significa, entonces, que la etapa S1 comprende analizar el indicador de longitud 23 presente en la cabecera de segmento 21.

En una realización particular, el indicador de longitud 23 está presente en la cabecera de segmento 21 que precede inmediatamente al campo de extensión 24. Por ende, el campo de extensión 24, sigue directamente el indicador de longitud 23 en la cabecera de segmento 21 en esta realización particular. En tal caso, la etapa S2 de la Fig. 4 podría comprender, en una realización, determinar ignorar, durante la decodificación de la representación codificada 20, cualquier valor o cualesquiera datos del campo de extensión 24 que sigue directamente el indicador de longitud 23 en la cabecera de segmento 21 hasta el fin del campo de extensión 24. Este fin del campo de extensión 24 es identificado en función del indicador de longitud 23, tal como se ha divulgado anteriormente en conexión con la etapa S11 de la Fig. 5.

En otra realización particular, el indicador de longitud 23 no está necesariamente presente en la cabecera de segmento 21, pero está asociado de otra forma con la representación codificada 20. Por ejemplo, normalmente una cabecera de segmento 21 comprende un identificador 26 de un conjunto de parámetros asociado con la representación codificada 20 y que comprende datos aplicables al segmento actual. La Fig. 3C ilustra este caso con un identificador (ID) 26 de un conjunto de parámetros (PS) en la cabecera de segmento 21.

Por ejemplo, un conjunto de parámetros de adaptación (APS) comprende información de control válida para más de un segmento. La información de control puede diferir entre los segmentos. Un conjunto de parámetros de imagen (PPS) comprende información de control válida para varias imágenes, y que puede ser igual para múltiples imágenes de la misma secuencia o flujo de vídeo. Un conjunto de parámetros de secuencia (SPS) comprende información de control válida para toda una secuencia o flujo de vídeo. La HEVC también usa lo que se denomina Conjunto de Parámetros de Vídeo (VPS). En tal caso, el indicador de longitud 23 podría estar presente en cualquiera de estos conjuntos de parámetros, tal como en el APS, VPS, PPS o SPS aplicable al segmento actual. En tal caso, la cabecera de segmento 21 comprende, preferentemente, un identificador del conjunto de parámetros 26 que permite la identificación del conjunto relevante de parámetros que lleva el indicador de longitud 23. El decodificador podría entonces usar este identificador del conjunto de parámetros 26 para identificar el conjunto correcto de parámetros y luego analizar el indicador de longitud 23 del conjunto de parámetros identificado. El identificador del conjunto de parámetros 26 podría identificar directamente el conjunto de parámetros, tal como un identificador de APS o un identificador de PPS. En un enfoque alternativo, el identificador del conjunto de parámetros 26 identifica un conjunto de parámetros que, a su vez, comprende otro identificador del conjunto de parámetros para otro conjunto de parámetros que lleva el indicador de longitud 23. Por ejemplo, la cabecera de segmento 21 podría comprender un identificador de PPS para un PPS que sea aplicable al segmento actual. A su vez, este PPS puede comprender un identificador de SPS para un SPS que sea aplicable al PPS y al segmento actual, y comprendiendo este SPS el indicador de longitud 23.

Si es probable que el tamaño o longitud del campo de extensión 24 en términos del número de bytes o de bits sea similar para varios segmentos en el flujo de vídeo, podría preferirse añadir el indicador de longitud 23 en un conjunto de parámetros, en lugar de señalarlo directamente en cada representación codificada 20. En tal caso, podría reducirse algo el tamaño o longitud total de la representación codificada 20 al no tener que incluir los bits del indicador de longitud 23 en la cabecera de segmento

21. Por lo tanto, en este enfoque, múltiples segmentos compartirán un mismo indicador de longitud 23 al referirse al mismo conjunto de parámetros que comprende el indicador de longitud 23.

Sin embargo, hay generalmente mucha mayor flexibilidad al incluir el indicador de longitud 23 directamente en las cabeceras de segmento 21 en cada representación codificada 20. Por lo tanto, en consecuencia, se puede permitir que el tamaño o longitud del campo de extensión 24 difiera entre diferentes segmentos. Además, no se requiere ninguna referencia a un conjunto de parámetros para analizar el indicador de longitud 23.

El análisis del indicador de longitud, según se muestra en la etapa S1 de la Fig. 4, no tiene que ser realizado necesariamente para cada representación codificada 20. En particular, si el indicador de longitud 23 está presente en un conjunto de parámetros identificado en función de un identificador del conjunto de parámetros 26 presente en la cabecera de segmento, podría identificarse el conjunto relevante de parámetros para la primera representación codificada 20 de la secuencia de vídeo que comprenda el identificador del conjunto de parámetros 26. A continuación, los datos del conjunto identificado de parámetros son entonces analizados y los valores decodificados y los datos presentes en el mismo, incluyendo el indicador de longitud 23, son normalmente almacenados en una memoria del decodificador. Entonces el decodificador hace un seguimiento del conjunto activado de parámetros, es decir, un conjunto de parámetros que ya ha sido analizado y decodificado. Esto quiere decir que cuando una representación codificada 20 de la secuencia de vídeo subsiguiente comprenda un identificador del conjunto de parámetros 26 que identifique a este conjunto de parámetros, los datos, incluyendo el indicador de longitud 23, podrían ser recuperados de los valore almacenados en la memoria.

Por ende, el análisis del indicador de longitud 23 en la etapa S1 no tiene que realizarse necesariamente para cada representación codificada 20, en particular si el indicador de longitud 23 está presente en un conjunto de parámetros. Sin embargo, esta etapa S1 se realiza preferentemente al menos una vez para una representación codificada 20 que tenga un identificador del conjunto de parámetros 26 apuntando al conjunto relevante de parámetros en su cabecera de segmento 21. La etapa S1 podría comprender, entonces, para cualquier representación codificada 20 de la secuencia de vídeo que siga que comprenda un identificador del conjunto de parámetros 26 para este conjunto relevante de parámetros, leer o recuperar, de una memoria, el indicador de longitud asociado con la representación codificada e indicativo de la longitud del campo de extensión en la cabecera de segmento.

El indicador de longitud 23 (véanse las Figuras Fig. 3A y 3B) o el identificador del conjunto de parámetros 26 (véase la Fig. 3C) no tienen necesariamente que preceder directamente el campo de extensión 24 en la cabecera de segmento 21, según se muestra en las Figuras 3A-3C. Generalmente se prefiere hacer que el campo de extensión 24 siga inmediatamente al indicador de longitud 23 o al identificador del conjunto de parámetros 26. Sin embargo, esto no es necesario. En realizaciones alternativas, podría haber presentes otros datos de cabecera de segmento, es decir, al menos un elemento de sintaxis o palabra de código, entre la posición del indicador de longitud 23 o el identificador del conjunto de parámetros 26 y el campo de extensión 24 en la cabecera de segmento 21.

La Fig. 6 es un diagrama de flujo de etapas adicionales opcionales del método de la Fig. 4. En la etapa S21 se analiza una bandera de presencia de extensión en la cabecera de segmento asociada con la representación codificada. Esta bandera de presencia de extensión en la cabecera de segmento podría estar presente en la cabecera de segmento 21 o estar asociada de otro modo con la representación codificada 20. Por ejemplo, la bandera de presencia de extensión en la cabecera de segmento podría estar presente en un conjunto de parámetros asociado con el segmento, tal como en un PPS. Preferentemente, el método comprende entonces la etapa opcional S20, en la que un conjunto de parámetros, tal como un PPS, asociado con la representación codificada 20 es identificado en función de un identificador del conjunto de parámetros, tal como un identificador de PPS, obtenible en función de la cabecera de segmento 21. El identificador del conjunto de parámetros podría ser recuperado directamente de la cabecera de segmento 21 o podría ser recuperado de otro conjunto de parámetros que esté identificado por otro identificador del conjunto de parámetros directamente recuperado de la cabecera de segmento 21. La etapa S21 comprende entonces, preferentemente, el análisis de la bandera de la extensión de cabecera de segmento presente en el conjunto de parámetros, tal como en el PPS.

El análisis de la bandera de presencia de extensión en la cabecera de segmento en la etapa S21 no tiene que ser realizado necesariamente para cada representación codificada 20 de una secuencia de vídeo, en particular si la bandera de presencia de extensión en la cabecera de segmento está presente en un conjunto de parámetros identificado en la etapa S20. En tal caso, se podría identificar, analizar y decodificar el conjunto relevante de parámetros para la primera representación codificada 20 que tenga un identificador del conjunto de parámetros 26 apuntando a este conjunto de parámetros. Los datos decodificados del conjunto de parámetros, incluyendo la bandera de presencia de extensión en la cabecera de segmento, podrían ser almacenados entonces en una memoria del decodificador. En tal caso, el valor de la bandera de presencia de extensión en la cabecera de segmento puede ser simplemente leído o recuperado de memoria por cualquier representación codificada 20 subsiguiente que

tenga un identificador del conjunto de parámetros 26 apuntando al conjunto de parámetros analizado y decodificado anteriormente.

La bandera de presencia de extensión en la cabecera de segmento indica si hay o no presente algún campo de extensión 24 en la cabecera de segmento 21. Así, si la bandera de presencia de extensión en la cabecera de segmento tiene un primer valor predefinido, tal como 1_{bin}, hay un campo de extensión 24 presente en la cabecera de segmento 21 y la representación codificada 20 está asociada con un indicador de longitud 23. Sin embargo, si la bandera de presencia de extensión en la cabecera de segmento tiene un segundo valor predefinido, tal como 0_{bin}, no hay ningún campo de extensión 24 presente en la cabecera de segmento 21 y, por lo tanto, ningún indicador de longitud 23 se asocia con la representación codificada 20.

En una etapa S22 siguiente, la bandera analizada de presencia de extensión en la cabecera de segmento es analizada y se determina si tiene el primer valor predefinido o el segundo valor predefinido. Por ejemplo, si la bandera de presencia de extensión en la cabecera de segmento está puesta —es decir, es igual a 1_{bin} —, el método prosigue a la etapa S1 de la Fig. 4, en la que se analiza el indicador de longitud 23. Así, en este caso hay un campo de extensión 24 presente y un decodificador preexistente precisa identificar la porción de la cabecera de segmento 21 que corresponde al campo de extensión 24.

Si la bandera de presencia de extensión en la cabecera de segmento no está puesta —es decir, es igual a 0_{bin} —, el método prosigue, en vez de ello, a la etapa S23, en la que el decodificador puede simplemente empezar a analizar y decodificar los datos de segmento 25 que siguen a la cabecera de segmento 21. Así, en este caso, no hay ningún campo de extensión presente en la cabecera de segmento 21 que precise ser identificado por un decodificador preexistente.

En una realización particular, la sintaxis de RBSP del conjunto de parámetros de imagen podría tener este aspecto:

pic_parameter_set_rbsp( ) {	**Descriptor**
…	
slice_header_extension_present_flag	u(1)
…	
}	

Que ***slice_header_extension_present_flag*** sea igual a 0 especifica que no hay ningún elemento de sintaxis de la extensión de cabecera de segmento presente en la cabecera de segmento para imágenes codificadas que se refieran al conjunto de parámetros de imagen.

En esta realización particular, una sintaxis general de cabecera de segmento podría entonces tener este aspecto:

slice_header( ) {	**Descriptor**
…	
if(*slice_header_extension_present_flag*) {	
slice_header_extension_length	ue(v)
for(i=0; i<*slice_header_extension_length*; i++)	
slice_header_extension_data_byte[i]	u(8)
}	
byte_alignment( )	
}	

El elemento de sintaxis ***slice_header_extension_length*** especifica la longitud de los datos de la extensión de cabecera de segmento en bytes, sin incluir los bits usados para señalizar el propio elemento

de sintaxis *slice_header_extension_length*. El valor de *slice_header_extension_length* podría estar en el intervalo de 0 a 256, inclusive. El tamaño de *slice_header_extension_length* no tiene necesariamente que estar en Código Universal de Longitud Variable (*Universal Variable Length Code*, UVLC), sino que podría ser de longitud fija, es decir, u(q) para algún entero positivo q.

El elemento de sintaxis ***slice_header_extension_data_byte*** puede tener cualquier valor. Los decodificadores preexistentes deberían ignorar el valor de *slice_header_extension_data_byte. de* forma alternativa, *slice_header_extension_data_byte* podría usar bits, es decir, u(1), en vez de bytes, es decir, u(8).

La bandera de presencia de extensión en la cabecera de segmento implica que el decodificador puede determinar, leyendo simplemente esta única bandera, preferentemente en forma de bandera de un solo bit, si una representación codificada 20 comprende algún campo de extensión 24 que precise ser procesado —es decir, analizado y usado— por un decodificador compatible con extensiones o ignorado por un decodificador preexistente. Por ende, el uso de la bandera de presencia de extensión en la cabecera de segmento simplifica la decodificación de las representaciones codificadas 20 en situaciones en las que algunas representaciones codificadas 20 podrían comprender campos de extensión 24, mientras que en otras no. En tal caso, no es preciso que el decodificador analice ni use ningún indicador de longitud 23, sino que puede simplemente investigar el valor de la bandera de presencia de extensión en la cabecera de segmento.

La bandera de presencia de extensión en la cabecera de segmento es insertada ventajosamente en un conjunto de parámetros, dado que es probable que múltiples segmentos del flujo de vídeo no tengan ningún campo de extensión 24 o que tengan campos de extensión 24. Entonces, es más eficiente en el uso de bits señalizar la bandera de presencia de extensión en la cabecera de segmento en el conjunto de parámetros que incluir tal bandera de presencia de extensión en la cabecera de segmento en cada cabecera de segmento 21.

En una realización particular, los datos de segmento 25 comienzan inmediatamente después del fin del campo de extensión 24 en la representación codificada 20. Por ende, en este enfoque la posición del bit siguiendo el fin del campo de extensión 24 identificado en función del indicador de longitud 23 constituye el primer bit de los datos de segmento 25.

En otra realización particular, los datos de alineamiento de bytes 22 podrían ser insertados entre el campo de extensión 24 y los datos de segmento 25, según se indica en la Fig. 3A y se menciona además en la sintaxis general de cabecera de segmento más arriba. Tales datos de alineamiento de bytes pueden ser empleados al final de la cabecera de segmento 21 para conseguir que el inicio de los datos de segmento 25 esté alineado en bytes. En tal caso, el indicador de longitud 23 puede ser usado para identificar la porción de la cabecera de segmento 21 que está ocupada por el campo de extensión 24 y podría indicar, por ejemplo, la longitud del campo de extensión 24 y, por ello, señalizar el fin del campo de extensión 24. Posteriormente siguen los datos de alineamiento de bytes 22 y luego los datos de segmento 25. Esto quiere decir que, aunque se usen los datos de alineamiento de bytes 22, el decodificador podrá identificar el inicio de los datos de segmento 25 y la porción de la cabecera de segmento 21 que constituye el campo de extensión 24 en función del indicador de longitud 23.

Preferentemente, los datos de alineamiento de bytes 22 tienen una sintaxis bien definida, por ejemplo:

byte_alignment( ) {	**Descriptor**
bit_equal_to_one /* *igual a* 1 */	f(1)
while(!*byte_aligned*( ))	
bit_equal_to_zero /* *igual a* 0 */	f(1)

En este ejemplo, los datos de alineamiento de bytes siempre comprenden al menos un bit. Si el fin del campo de extensión 24 debe estar alineado en bytes, se usa una palabra de código de 8 bits: $1000\ 0000_{bin}$.

La Fig. 3B ilustra una realización de la representación codificada 20 de un segmento que muestra que el indicador de longitud 23 y el campo de extensión 24 no tienen necesariamente la forma de los últimos elementos de sintaxis (excluyendo los datos opcionales 22 de alineamiento de bytes) de la cabecera de segmento 21. En clara contraposición, el indicador de longitud 23 y el campo de extensión 24 (Fig. 3B), o, ciertamente, el identificador del conjunto de parámetros 26 y el campo de extensión (Fig. 3C), podrían estar presentes en cualquier lugar de la cabecera de segmento 21.

La Fig. 7 es un diagrama de flujo que ilustra etapas adicionales opcionales del método de la Fig. 4. El método prosigue de la etapa S2 de la Fig. 4. En una etapa S siguiente se identifica el inicio de los datos de segmento 25 en la representación codificada 20 en función del fin del campo de extensión 24 identificado en función del indicador de longitud 23, tal como se divulga en la etapa S11 de la Fig. 5. Según se ha mencionado anteriormente, el inicio de los datos de segmento 25 podría seguir directamente el fin del campo de extensión 24. En un enfoque alternativo, los datos de alineamiento de bytes 22 se interponen entre el campo de extensión 24 y el inicio de los datos de segmento 25. En tal caso, el indicador de longitud 23 puede ser usado para identificar el fin del campo de extensión 24. El bit o los bits siguientes en la representación codificada 20 constituye(n) los datos de alineamiento de bytes 22. Esta porción de la representación codificada 20 es interpretada por el decodificador como datos de alineamiento de bytes 22, dado que tiene una sintaxis bien definida, tal como se ha ejemplificado más arriba. Una vez que se alcanza el fin de los datos de alineamiento de bytes 22, se encuentra el inicio de los datos de segmento 25. También es posible, lo que es descrito adicionalmente más abajo, que otros datos de cabecera de segmento pudieran seguir al campo de extensión 24.

Cuando el decodificador haya identificado el inicio de los datos de segmento 25, puede analizar y decodificar los datos de segmento 25 en la etapa S31 a partir del inicio identificado de los datos de segmento 25.

Preferentemente, un decodificador preexistente analiza y decodifica los datos de segmento en la etapa S31 ignorando —es decir, no usando— cualquier valor o cualesquiera datos del campo de extensión 24 en la cabecera de segmento 21 durante el procedimiento de análisis y decodificación de la etapa S31. Por ende, el decodificador preexistente no puede usar el campo de extensión 24 en la cabecera de segmento 21, dado que este campo de extensión 24, según el perfil base o la especificación base a los que se atiene el decodificador preexistente, es indeterminado.

Sin embargo, un decodificador compatible con extensiones debería hacer uso de los datos en el campo de extensión 24 en la cabecera de segmento durante el procedimiento de análisis y decodificación, incluyendo opcionalmente un procesamiento posterior a la decodificación de los datos de segmento 25.

Las presentes realizaciones también pueden ser aplicadas al caso en el que la cabecera de segmento 21 comprende múltiples —es decir, al menos dos— campos de extensión 24. En tal caso, cada campo de extensión 24 comprende, preferentemente, un indicador respectivo 23 de longitud. Los campos de extensión 24 podrían estar dispuestos todos en conexión con el fin de la cabecera de segmento 21, posiblemente seguidos por datos de alineamiento de bytes opcionales 22. Sin embargo, también es posible contar con una organización distribuida de los campos de extensión 24 en la cabecera de segmento 21. En este caso, los campos de extensión 24 podrían ser proporcionados en cualquier lugar de la cabecera de segmento 21. Por ejemplo, podría haber un primer campo de extensión 24 presente en la parte primera o central de la cabecera de segmento 21, mientras que un segundo campo de extensión 24 podría estar presente en la parte final de la cabecera de segmento 21.

Si puede haber múltiples campos de extensión 24 presentes en la cabecera de segmento 21, los respectivos indicadores de longitud 23 podrían estar presentes en uno o más conjuntos de parámetros, en la cabecera de segmento 21, o estar distribuidos entre la cabecera de segmento 21 y uno o más conjuntos de parámetros. Por ejemplo, un primer indicador de longitud 23 de un primer campo de extensión 24 podría estar presente en un conjunto de parámetros, tal como un PPS. Un segundo indicador de longitud 23 de un segundo campo de extensión 24 podría entonces estar presente en la cabecera de segmento 21. La presencia de cualquiera de los campos de extensión 24 e indicadores de longitud 23 podría ser señalizada por una bandera, por ejemplo en un conjunto de parámetros. Por ejemplo, un PPS podría comprender el primer indicador de longitud 23 y una bandera que indique la presencia o la ausencia del segundo campo de extensión 24. Entonces, el segundo indicador de longitud 23 y el segundo campo de extensión 24 están presentes en la cabecera de segmento 21 si la bandera ha sido puesta a un valor definido.

En el caso de múltiples campos de extensión 24, la etapa S1 de la Fig. 4 se realiza preferentemente una vez para cada campo de extensión 24 e indicador de longitud 23. Así, cada indicador de longitud 23 asociado con la representación codificada 20 es identificado y analizado para permitir la identificación del respectivo campo de extensión 24 en la cabecera de segmento 21 para que un decodificador preexistente pueda determinar ignorar el valor o los valores respectivos de los campos de extensión en la etapa S2.

La Fig. 8 es un diagrama de flujo que ilustra otra realización de un método de decodificación de una representación codificada de un segmento de una imagen. La representación codificada comprende una cabecera de segmento y datos de segmento. Esta realización es llevada a cabo en particular por un decodificador compatible con extensiones, es decir, un decodificador que comprende y puede hacer uso del valor o los valores de un campo de extensión presente en la cabecera de segmento.

El método se inicia en la etapa S80, en la que se analiza un indicador de longitud asociado con la representación codificada e indicativo de una longitud de un campo de extensión en la cabecera de segmento. Esta etapa S1 corresponde básicamente a la etapa S1 de la Fig. 4 y no es expuesta adicionalmente en la presente memoria.

Una etapa S81 siguiente identifica el campo de extensión en la cabecera de segmento en función del indicador de longitud analizado en la etapa S80. Así, en la etapa S81 se usa el indicador de longitud para identificar la porción de la cabecera de segmento que está ocupada por el campo de extensión. Esta etapa S81 puede ser efectuada como se ha divulgado previamente en la presente memoria, por ejemplo en conexión con las Figuras 4 y 5.

Si la cabecera de segmento comprende múltiples campos de extensión, las etapas S80 y S81 se llevan a cabo, preferentemente, una vez para cada campo de extensión y el indicador de longitud asociado. En la etapa S82 siguiente, se decodifica la representación codificada del segmento en función de al menos un valor, preferentemente todos los valores, del campo de extensión en la cabecera de segmento identificado en la etapa S81 en función del indicador de longitud analizado en la etapa S80.

Así, en esta realización, el decodificador es compatible con extensiones y, por lo tanto, usará el valor o los valores del campo de extensión o al menos una porción de los valores cuando decodifique la representación codificada del segmento y, en particular, durante la decodificación de los datos de segmento en la representación codificada.

El campo de extensión, por ejemplo, podría incluir una o más banderas y/u otros elementos de sintaxis que dictan y proporcionan instrucciones al decodificador de cómo llevar a cabo la decodificación de los datos de segmento. Por ende, el decodificador compatible con extensiones hace uso de los datos en el campo de extensión en la cabecera de segmento durante el procedimiento de análisis y decodificación, y que incluye opcionalmente un procesamiento posterior a la decodificación de los datos de segmento.

La decodificación de la representación codificada del segmento en función de al menos un valor del campo de extensión en la etapa S82 está relacionada con cualquier procesamiento de la representación codificada del segmento llevada a cabo por un decodificador. Por ende, la decodificación definida en la etapa S82 no está meramente limitada a la generación en sí de datos de píxeles del segmento a partir de los datos de segmento presentes en la representación codificada del segmento. Esto quiere decir que el al menos un valor del campo de extensión podría ser usado por el decodificador, por ejemplo, para gestionar una memoria intermedia de imágenes de referencia que comprenda imágenes previamente decodificadas de una secuencia de vídeo que estén almacenadas temporalmente para darles salida, por ejemplo, para su visualización, y/o que estén almacenadas temporalmente para ser usadas como referencia de decodificación para las imágenes siguientes, según un orden de decodificación, de la secuencia de vídeo. Alternativamente, el al menos un valor del campo de extensión podría ser usado por el decodificador, por ejemplo, para eliminar o descartar del flujo de bits unidades NAL específicas. Por ejemplo, el campo de extensión podría indicar que puede descartarse una unidad NAL específica, y, por lo tanto, no ser introducida o eliminada de la memoria intermedia de imágenes de referencia si solo se está decodificando una representación codificada de capa superior.

Así, la decodificación definida en la etapa S82 abarca diversas operaciones del decodificador que tienen lugar en conexión con la generación en sí —es decir, precediéndola, simultáneamente o siguiéndola— de valores de píxel para un segmento actual.

Una realización particular se refiere a un método llevado a cabo durante la decodificación de una representación codificada de un segmento de una imagen en una secuencia de vídeo que comprende múltiples imágenes. Cada imagen de la secuencia de vídeo pertenece a una capa de múltiples capas. La representación codificada comprende una cabecera de segmento y datos de segmento. El método comprende analizar un indicador de longitud asociado con la representación codificada e indicativo de una longitud de un campo de extensión en la cabecera de segmento. El campo de extensión en la cabecera de segmento es identificado en función del indicador de longitud. El método también comprende procesar una estructura estratificada de dichas múltiples imágenes en función de al menos un valor del campo de extensión en la cabecera de segmento.

En esta realización particular, las imágenes de la secuencia de vídeo son organizadas en una estructura estratificada con las múltiples capas. Estas capas podrían entonces corresponder a diferentes vistas en el caso de un visionado de vistas múltiples, a diferentes capas temporales en el caso de vídeo de redimensionamiento, etc. El procesamiento de la estructura estratificada, incluyendo, por ejemplo, la eliminación en el decodificador de una o más imágenes o unidades NAL que contengan imágenes, es llevado a cabo preferentemente, al menos en parte, en función de al menos un valor del campo de extensión.

Siguen en la presente memoria varias realizaciones ejemplares que dan a conocer variantes y alternativas de implementación de las presentes realizaciones.

En una primera realización ejemplar hay en la cabecera de segmento 21 una palabra de código (elemento de sintaxis), es decir, el indicador de longitud 23, que representa la longitud de un campo de extensión 24 en la cabecera de segmento 21. El codificador está configurado para insertar el indicador de longitud 23 como un elemento de sintaxis según esta primera realización ejemplar. Un decodificador analizará el valor de la palabra de código, es decir, el indicador de longitud 23, para localizar el inicio de los datos de segmento 25 y, con ello, saltarse el campo de extensión 24.

El nombre de la palabra de código que representa la longitud del campo de extensión 24 podría ser, por ejemplo, *slice_header_extension_length.*

Según esta realización ejemplar, el decodificador está configurado para llevar a cabo las siguientes etapas.

1. Se analiza el elemento de sintaxis 23 *slice_header_extension_length*.

2. El decodificador analiza e ignora los valores con número de bits igual a *slice_header_extension_length* que sigan inmediatamente después del elemento de sintaxis 23 *slice_header_extension_length*.

3. El decodificador analiza y decodifica los datos de segmento 25 empezando con el primer bit que sigue después del análisis del número de bits igual a *slice_header_extension_length* (en la posición de bit en la que acabó el análisis en la etapa 2).

Por lo tanto, en esta realización, el decodificador analiza y lee los datos o valores del campo de extensión 24 en la cabecera de segmento 21 durante la decodificación de la cabecera. Sin embargo, aunque los valores son analizados y posiblemente decodificados, el decodificador ignora los valores, es decir, no los usa ulteriormente durante la decodificación y el procesamiento de los datos de segmento 25.

Alternativamente, puede configurarse un decodificador para que lleve a cabo las siguientes etapas:

1. Se analiza el elemento de sintaxis 23 23 *slice_header_extension_length*.

2. El decodificador se salta hacia delante en el flujo 20 de bits un número de bits igual a *slice_header_extension_length*.

3. El decodificador analiza y decodifica los datos de segmento 25 empezando con el primer bit que sigue después del salto del número de bits igual a *slice_header_extension_length* (en la posición de bit en la que acabó el análisis en la etapa 2).

En esta realización, el decodificador no analiza ni lee los datos o valores del campo de extensión 24, sino que, en vez de ello, se salta la porción de la cabecera de segmento 21 que está ocupada por el campo de extensión 24 definido en función del indicador de longitud 23.

En una segunda realización ejemplar, la presencia del elemento de sintaxis 23 descrito en la primera realización ejemplar está condicionada a otro elemento de sintaxis denominado, por ejemplo, *slice_header_extension_present_flag.* Si *slice_header_extension_present_flag* es 0, no están presentes el elemento de sintaxis 23 *slice_header_extension_length* ni los datos 24 de extensión. Si *slice_header_extension_present_flag* es 1, están presentes el elemento de sintaxis 23 *slice_header_extension_length* y los datos 24 de extensión. Preferentemente, el elemento de sintaxis *slice_header_extension_present_flag* es señalizado en el SPS, pero puede ser señalizado, alternativamente, en el PPS, el VPS, el APS o en la cabecera de segmento 21.

En consecuencia, el codificador está configurado para condicionar la presencia del indicador de longitud 23 de la primera realización ejemplar usando una bandera.

Según esta realización, el decodificador está configurado para llevar a cabo las siguientes etapas ordenadas:

1. Se analiza el elemento de sintaxis *slice_header_extension_present_flag* del SPS activo.

2. Si *slice_header_extension_present_flag* es igual a 1:

    a. Se analiza el elemento de sintaxis 23 *slice_header_extension_length*.

b. El decodificador analiza e ignora los valores con número de bits igual a *slice_header_extension_length* que sigan inmediatamente después del elemento de sintaxis 23 *slice_header_extension_length*.

c. El decodificador analiza y decodifica los datos de segmento 25 empezando con el primer bit que sigue después del análisis del número de bits igual a *slice_header_extension_length* (en la posición de bit en la que acabó el análisis en la etapa b).

3. Si no (*slice_header_extension_present_flag* es igual a 0), el decodificador analiza y decodifica los datos de segmento 25 sin analizar un elemento de sintaxis 23 *slice_header_extension_length* y sin analizar ningún dato de extensión 24. No estarán presentes en el segmento ni el elemento de sintaxis 23 *slice_header_extension_length* ni ningún dato de extensión 24.

Alternativamente, un decodificador puede estar configurado para llevar a cabo las siguientes etapas:

1. Se usa el elemento de sintaxis *slice_header_extension_present_flag* del SPS activo.

2. Si *slice_header_extension_present_flag* es igual a 1:

a. Se analiza el elemento de sintaxis 23 *slice_header_extension_length*.

b. El decodificador se salta hacia delante en el flujo 20 de bits un número de bits igual a *slice_header_extension_length*.

c. El decodificador analiza y decodifica los datos de segmento empezando con el primer bit que sigue después del salto del número de bits igual a *slice_header_extension_length* (en la posición de bit en la que acabó el análisis en la etapa b).

Si no (*slice_header_extension_present_flag* es igual a 0), el decodificador analiza y decodifica los datos de segmento 25 sin analizar un elemento de sintaxis 23 *slice_header_extension_length* y sin efectuar un salto en el flujo 20 de bits. No estarán presentes en el segmento ni el elemento de sintaxis 23 *slice_header_extension_length* ni ningún dato de extensión 24.

La tabla de sintaxis podría tener, por ejemplo, el siguiente aspecto:

slice_header( )	**Descriptor**
...	
if(*slice_header_extension_present_flag*) {	
slice_header_extension_length	ue(v)
for(i = 0; i < *slice_header_extension_length*; i++)	
slice_header_extension_data_flag	u(1)
}	
}	

En una tercera realización ejemplar, la longitud de toda la cabecera de segmento 21 está dada por un elemento de sintaxis 23 y se añade el campo de extensión 24 al final de la cabecera de segmento 21 de forma similar a los campos de extensión de los conjuntos de parámetros.

En consecuencia, el codificador está configurado para indicar la longitud de toda la cabecera de segmento 21 por medio de un elemento de sintaxis, es decir, el indicador de longitud 23, y se añade el campo de extensión 24 al final de la cabecera de segmento 21 de forma similar a los campos de extensión de los conjuntos de parámetros.

Según esta realización, el decodificador está configurado para llevar a cabo las siguientes etapas:

1. Si está presente, se usa el elemento de sintaxis *slice_header_extension_present_flag* del SPS activo. Si no, se omiten las etapas 1, 2 y 3 y solo se llevan a cabo las etapas a, b y c.

2. Si *slice_header_extension_present_flag* es igual a 1:

a. Se calcula el elemento de sintaxis *slice_header_extension_length* como la longitud de la cabecera de segmento 21 menos la posición de bit actual.

b. El decodificador analiza e ignora los valores con número de bits igual a *slice_header_extension_length* que sigan inmediatamente después del elemento de sintaxis *slice_header_extension_length*, es decir, indicando el elemento de sintaxis la longitud de toda la cabecera de segmento 21.

c. El decodificador analiza y decodifica los datos de segmento 25 empezando con el primer bit que sigue después del análisis del número de bits igual a *slice_header_extension_length* (en la posición de bit en la que acabó el análisis en la etapa b).

3. Si no (*slice_header_extension_present_flag* es igual a 0), el decodificador analiza y decodifica los datos de segmento 25 sin analizar un elemento de sintaxis 23 *slice_header_extension_length* y sin analizar ningún dato de extensión 24. No estarán presentes en el segmento ni el elemento de sintaxis 23 *slice_header_extension_length* ni ningún dato de extensión 24.

Así, en esta realización, la etapa S10 de la Fig. 5 relativa a la determinación de la longitud del campo de extensión 24 es efectuada, preferentemente, calculando la longitud del campo de extensión 24 restando una posición de bit o byte del fin del indicador de longitud 23 en la cabecera de segmento 21 de un valor representado por el indicador de longitud 23. Este valor representado por el indicador de longitud 23 es, entonces, preferentemente, la longitud de la cabecera de segmento 21.

En una variante de la etapa 3 de las realizaciones ejemplares primera y segunda y de la etapa c de la tercera realización ejemplar, se pueden interponer datos de alineamiento de bytes 22 entre el campo de extensión 24 y los datos de segmento 25. En tal caso, el decodificador analiza los datos de alineamiento de bytes 22 empezando con el primer bit que sigue después del análisis del número de bits igual a *slice_header_extension_length* (realizaciones ejemplares primera y tercera) o con el primer bit que sigue después del salto del número de bits igual a *slice_header_extension_length* (segunda realización ejemplar). Después, el decodificador, una vez se ha alcanzado el fin de los datos de alineamiento de bytes 22, analiza y decodifica los datos de segmento 25.

En una cuarta realización ejemplar, se usa un patrón específico de bits (“código de inicio”) para indicar el fin del campo de datos de extensión 24. Por ende, el codificador está configurado para insertar tal patrón específico de bits. El decodificador analiza todos los datos hasta que se encuentra el patrón de bits. Se puede requerir o no que el patrón de bits esté alineado en bytes. A continuación, el decodificador decodifica los datos de segmento 25 empezando con el primer bit posterior al patrón específico de bits.

Un decodificador puede llevar a cabo las etapas:

1. Se analiza la cabecera de segmento 21 hasta el punto en el que comienza el campo de extensión 24.

2. Desde ese punto, el decodificador explora el flujo de bits y busca el patrón específico de bits (el “código de inicio”) que indica dónde acaban los datos de extensión 24 y empiezan los datos de segmento 25.

3. Después del punto en el que se encuentra el patrón específico de bits, el decodificador analiza y decodifica el segmento.

Así, en esta realización, el método dado a conocer en la Fig. 4 comprende, además, analizar la cabecera de segmento 21 desde el inicio de la cabecera de segmento 21 hasta el inicio del campo de extensión 24 en la cabecera de segmento 21. El método comprende, además, explorar la cabecera de segmento 21 desde el inicio del campo de extensión 24 hasta alcanzar el indicador de longitud 23, que está en forma de un código específico de inicio. El método también comprende identificar el fin del campo de extensión 24 como código específico de inicio.

En una quinta realización ejemplar se usa cualquiera de las realizaciones ejemplares primera a tercera, pero con la diferencia de que el elemento de sintaxis 23 *slice_header_extension_length* no es señalizado en la cabecera de segmento 21. En vez de ello, es señalizado en una estructura de datos a la que puede hacer referencia la cabecera de segmento 21, tal como el SPS, el VPS, el PPS o el APS.

En una sexta realización ejemplar, se usa cualquiera de las realizaciones ejemplares anteriormente presentadas, pero con la diferencia de que la longitud es expresada en bytes en vez de en bits.

En una séptima realización ejemplar, se usa cualquiera de las realizaciones ejemplares segunda a sexta con un requisito del flujo de bits de que el elemento de sintaxis *slice_header_extension_present_flag*

deba estar puesto igual a 0. Un codificador puede estar configurado para poner el elemento de sintaxis *slice_header_extension_present_flag* a 0.

En una realización particular, un codificador compatible con cierto perfil, en particular el perfil base, es obligado, preferentemente, a poner *slice_header_extension_present_flag* a 0. Sin embargo, un decodificador compatible con el mismo perfil debe usar el indicador de longitud e ignorar los datos de extensión.

En una octava realización ejemplar, se usa cualquiera de las realizaciones ejemplares segunda a sexta con un significado específico introducido para el campo de datos de extensión 24. Tal significado puede corresponder a información adicional en una extensión de vistas múltiples o en una extensión de redimensionamiento.

Un codificador puede usar las realizaciones ejemplares poniendo el elemento de sintaxis *slice_header_extension_present_flag* a 1 y, para cada cabecera de segmento 21, codificar valores del campo 23 de datos de extensión según el significado que ha sido introducido. El codificador está configurado, además, para calcular la longitud del campo de datos de extensión 24 y señalar eso en el elemento de sintaxis 23 *slice_header_extension_length*. Alternativamente, si la longitud de la extensión 24 es conocida a priori (se usa, por ejemplo, una extensión de longitud fija), la longitud puede ser señalizada en el elemento de sintaxis 23 *slice_header_extension_length* sin calcularla primero.

Un decodificador que reconozca el significado introducido (y haga uso del mismo) para los datos 24 de extensión puede ser configurado para llevar a cabo las siguientes etapas:

1. Se usa el elemento de sintaxis *slice_header_extension_present_flag* del SPS activo.

2. Si *slice_header_extension_present_flag* es igual a 1:

    a. Se analiza el elemento de sintaxis 23 *slice_header_extension_length* (o se lo ignora, si ya se conoce la longitud de la extensión).

    b. El decodificador analiza y usa los valores con número de bits igual a *slice_header_extension_length* que sigan inmediatamente después del elemento de sintaxis 23 *slice_header_extension_length* según el significado introducido de los bits.

    c. El decodificador analiza y decodifica los datos de segmento 25 empezando con el primer bit que sigue después del salto del número de bits igual a *slice_header_extension_length* (en la posición de bit en la que acabó el salto en la etapa b).

3. Si no (*slice_header_extension_present_flag* es igual a 0), el decodificador analiza y decodifica los datos de segmento 25 sin analizar un elemento de sintaxis 23 *slice_header_extension_length* y sin efectuar un salto en el flujo 20 de bits. No estarán presentes en el segmento ni el elemento de sintaxis 23 *slice_header_extension_length* ni ningún dato de extensión 24.

Un decodificador que no reconozca el significado introducido (o no haga uso del mismo) para los datos de extensión 24 puede estar configurado para llevar a cabo las etapas presentadas en la segunda realización ejemplar.

En una novena realización ejemplar, se usa la octava realización ejemplar con el significado específico del campo de datos de extensión 24 ligado a uno o más perfiles de HEVC. Es decir, un decodificador que decodifique un flujo 20 de bits que se atenga a un perfil que usa el campo de datos de extensión 24 con un fin específico (con un significado especificado) decodificará los datos de extensión 24 según se describe en la octava realización y usará los datos de extensión 24 según el significado definido.

Un decodificador que decodifique un flujo 20 de bits que se atenga a un perfil que no usa los datos de extensión 24 con un fin específico llevará a cabo las etapas presentadas en la segunda realización ejemplar.

En una décima realización ejemplar, se usa cualquiera de las realizaciones ejemplares segunda, tercera, octava o novena, pero, cuando no hay ningún elemento de sintaxis *slice_header_extension_present_flag,* se infiere, en cambio, que es 1. No se señaliza ninguna extensión al señalizar una longitud de extensión igual a cero.

En una undécima realización ejemplar, puede usarse cualquiera de las realizaciones ejemplares anteriores, pero en la que no se señaliza el tamaño de la extensión 24. En vez de ello, se define una serie de códigos de longitud variable (*Variable Length Codes*, VLC) de tamaño fijo y/o variable.

En una duodécima realización ejemplar, podrían usarse dos mecanismos de cabecera de segmento para un segmento dado. En este caso, se señaliza un primer indicador de longitud en un conjunto de parámetros, tal como PPS, mientras que se señaliza un segundo indicador de longitud en la cabecera de segmento. En una realización, una bandera en el PPS indica si la cabecera de segmento comprende o no el segundo indicador de longitud.

Por lo tanto, la sintaxis de la RBSP del PPS puede tener este aspecto:

pic_parameter_set_rbsp( ) {	**Descriptor**
…	
num_slice_header_extension_bits	u(5)
slice_header_extension_present_flag	u(1)
…	

En este ejemplo, *num_slice_header_extension_bits* representa el primer indicador de longitud. Un valor de este primer indicador de longitud igual a 0 especifica que no hay presente ningún bit extra de cabecera de segmento en la RBSP de la cabecera del segmento para imágenes codificadas que se refieran al PPS. El elemento de sintaxis *num_slice_header_extension_bits* debería ser igual a 0 para flujos de bits que se atengan a la especificación del perfil base. La bandera *slice_header_extension_present_flag* es usada para señalizar la presencia del segundo indicador de longitud en la cabecera de segmento. Un valor de 0 de esta bandera especifica que no hay presente ningún elemento de sintaxis de la extensión de cabecera de segmento en las cabeceras de segmento para imágenes codificadas que se refieran al PPS. La bandera *slice_header_extension_present_flag* debería ser igual a 0 para flujos de bits que se atengan a la especificación del perfil base.

Entonces, la sintaxis de la RBSP de la capa de segmentos puede tener este aspecto:

```
slice_layer_rbsp( ) {
    slice_header( )
    slice_data( )
    rbsp_slice_trailing_bits( )
}
```

En correspondencia con ello, la sintaxis de cabecera de segmento sería, entonces, según se define a continuación:

slice_header( ) {	**Descriptor**
…	
for(i = 0; i < *num_slice_header_extension_bits*; i++)	
slice_header_extension_data_flag	u(1)
…	
if(*slice_header_extension_present_flag*) {	
slice_header_extension_length	ue(v)
for(i = 0; i < *slice_header_extension_length*; i++)	
slice_header_extension_data_byte	u(8)
}	
byte_alignment( )	
}	

En este ejemplo, el primer campo de extensión en la cabecera de segmento está compuesto por el elemento de sintaxis *slice_header_extension_data_flag*. El segundo indicador de longitud, *slice_header_extension_length*, especifica la longitud del segundo campo de extensión, es decir, *slice_header_extension_data_byte,* sin incluir los bits usados para señalizar el propio elemento de sintaxis *slice_header_extension_length*. En un ejemplo, el valor de *slice_header_extension_length* podría estar en el intervalo de 0 a 256, inclusive. El segundo campo de extensión en la cabecera de segmento puede tener cualquier valor.

Una decimotercera realización ejemplar es similar a la duodécima realización ejemplar dada a conocer anteriormente, pero con la siguiente sintaxis de RBSP de la capa de segmentos.

slice_layer_rbsp( ) {	**Descriptor**
slice_header( )	
slice_data( )	
if(*slice_header_extension_present_flag*) {	
slice_extension_flag	u(1)
if(*slice_extension_flag*)	
while(more_rbsp_data( ))	
slice_extension_data_flag	u(1)
}	
rbsp_slice_trailing_bits( )	
}	

En una implementación ejemplar de las realizaciones duodécima y decimotercera, la especificación base podría tener este aspecto:

for(i = 0; i < *num_slice_header_extension_bits*; i++)

slice_header_extension_data_flag

Una especificación de extensiones puede, entonces, hacer esto:

inter_layer_prediction_flag

Que *inter_layer_prediction_flag* sea igual a 0 indica que la imagen asociada con la unidad NAL no es usada como referencia en ninguna imagen perteneciente a una capa espacial más alta o de vistas múltiples. Que *inter_layer_prediction_flag* sea igual a 1 indica que la imagen asociada con la unidad NAL puede ser usada como referencia en una imagen perteneciente a una capa espacial más alta o de vistas múltiples.

Así, en esta implementación ejemplar, un decodificador que se atuviera a la especificación de extensiones podría interpretar que el primer campo de extensión en la cabecera de segmento indica el elemento de sintaxis *inter_layer_prediction_flag*.

En una realización de implementación, en la especificación de las extensiones, el campo de extensión o los campos de extensión en la cabecera de segmento son interpretados como palabras de código en sí, o sustituidos por ellas, en vez de, por ejemplo, la bandera de extensión y, con referencia a las realizaciones ejemplares duodécima y decimotercera, *slice_header_extension_data_flag* y *slice_header_extension_data_byte*. Las palabras de código pueden entonces estar en forma de banderas, palabras de código de longitud fija (*Fixed Length Code*, FLC), palabras de código de longitud variable (VLC) o incluso una mezcla de las mismas.

También es posible que un decodificador compatible con extensiones ignore algunos de los bits del campo de extensión. Por ejemplo, puede darse el caso de que las palabras de código de un campo de extensión sumen hasta 12 bits, pero que la longitud total del campo de extensión señalizada por el indicador de longitud esté puesta en 14 bits. Entonces los dos bits restantes son simplemente ignorados por el decodificador compatible con extensiones.

Según se ha mencionado en lo que precede, el campo de extensión 24 podría comprender, en un ejemplo, datos de extensión de vistas múltiples codificados y/o datos de extensión de redimensionamiento codificados.

En el caso anterior, el flujo de vídeo es un flujo de imágenes procedente de múltiples vistas (de cámara), por ejemplo una vista de cámara derecha y una vista de cámara izquierda en el caso de vídeo estereoscópico. En tal caso, una de estas múltiples, es decir, al menos dos, vistas podría ser una vista base, estando codificadas la una o más vistas adicionales de manera relativa a la vista base y posiblemente de forma mutua. Sin embargo, preferentemente, la vista base es decodificable independientemente, es decir, la vista base no usa como referencia imágenes de otras vistas. En tal caso, un decodificador preexistente podría decodificar correctamente la vista base aunque haya datos de extensión relativos a vistas adicionales o a una estructura de vistas múltiples presentes en las cabeceras de segmento. Esto es posible debido al indicador de longitud, usado según las diversas realizaciones dadas a conocer en la presente memoria.

En el último caso, el flujo de vídeo es un flujo de imágenes procedente de múltiples capas. En tal caso, una de estas múltiples capas es la capa base con una o más capas adicionales. Preferentemente, entonces, la capa base es decodificable independientemente, es decir, carece de referencia a imágenes en ninguna de las capas adicionales. En correspondencia con ello, una imagen que pertenezca a la capa adicional N puede preferentemente hacer referencia a imágenes en la capa base y en capas hasta la capa N, pero no por encima de la misma. En tal caso, un decodificador preexistente podría decodificar correctamente la capa base aunque haya datos de extensión relativos a las capas adicionales o a la estructura de múltiples capas presentes en las cabeceras de segmento. Esto es posible debido al indicador de longitud, usado según las diversas realizaciones dadas a conocer en la presente memoria.

Las realizaciones no están limitadas a HEVC, sino que pueden ser aplicadas a cualquier extensión de HEVC, tal como una extensión de redimensionamiento o una extensión de vistas múltiples, o a un códec de vídeo diferente, o, ciertamente, a cualquier otro estándar de codificación de vídeo.

La Fig. 9 es un diagrama esquemático de flujo de un método de codificación de un segmento 3 de una imagen 2, tal como una imagen 2 en un flujo de vídeo 1. El método comprende generar un indicador de longitud 23 en la etapa S40. Este indicador de longitud 23 es indicativo de una longitud de un campo de extensión 24 presente en una cabecera de segmento 21 del segmento 3. En una etapa S41 siguiente, el segmento 3 es codificado creando una representación codificada 20 del segmento 3 que comprende la cabecera de segmento 21 y datos de segmento 25. El indicador de longitud 23 es asociado con la representación codificada 20 en la etapa S42. La etapa S42 puede ser llevada a cabo antes de la etapa S41, sustancialmente en paralelo con ella o después de la misma.

La representación codificada 20 puede ser procesada ulteriormente para formar una unidad NAL 11, que opcionalmente puede ser empaquetada formando un paquete 10 de datos con una o más cabeceras 12, según se muestra en la Fig. 2.

Si la cabecera de segmento ha de contener múltiples campos de extensión, las etapas S40 y S42 de la Fig. 9 son realizadas, preferentemente, una vez para cada campo de extensión tal.

La Fig. 10 es un diagrama de flujo que ilustra etapas adicionales opcionales del método de la Fig. 9. En la etapa S51 se determina la longitud del campo de extensión 24 presente en la cabecera de segmento 21. El indicador de longitud 23 es generado, a continuación, en la etapa S40 de la Fig. 9 en función de la longitud del campo de extensión 24. Por ende, en una realización particular, se da al indicador de longitud 23 un valor que representa la longitud del campo de extensión 24 en bytes o bits.

En una etapa opcional S50 se determinan los valores del campo de extensión 24 para ver cuán largo se hace el campo de extensión 24. Alternativamente, el campo de extensión 24, dependiendo del tipo de datos de extensión, tales como datos de extensión de vistas múltiples o datos de extensión de redimensionamiento, podría tener una longitud predefinida. Según se ha mencionado anteriormente, los valores del campo de extensión 24 representan, preferentemente, datos de extensión de vistas múltiples y/o datos de extensión de redimensionamiento para el segmento 3.

El indicador de longitud 23 generado en la etapa S40 es insertado, preferentemente, en la cabecera de segmento 21 en la etapa S42. Alternativamente, el indicador de longitud 23 podría ser insertado en un conjunto de parámetros que se aplique al segmento actual 3. En tal caso, la cabecera de segmento 21 comprende, preferentemente, un identificador del conjunto de parámetros que permite la identificación del conjunto relevante de parámetros que comprende el indicador de longitud 23.

Si el indicador de longitud 23 es insertado en la cabecera de segmento 21 en la etapa S42, el indicador de longitud 23 es insertado, preferentemente, directamente antes del campo de extensión 24 en la cabecera de segmento 21.

En una realización, se usa una bandera de presencia de extensión en la cabecera de segmento para señalizar la presencia o la ausencia de un campo de extensión 24 en una cabecera de segmento 21. Esta realización se ilustra esquemáticamente en la Fig. 11. El método se inicia en la etapa S60, en la que la bandera de presencia de extensión en la cabecera de segmento es puesta a un valor predefinido, tal como 1_{bin}, para indicar la presencia del campo de extensión 24 en la cabecera de segmento 21. En una etapa S61 siguiente, la bandera de presencia de extensión en la cabecera de segmento es asociada con la representación codificada 20. Esta asociación de la etapa S61 podría ser la inserción de la bandera de presencia de extensión en la cabecera de segmento en un conjunto de parámetros que sea aplicable al segmento actual 3. En tal caso, la cabecera de segmento 21 comprende, preferentemente, un identificador del conjunto de parámetros que permite la identificación del conjunto relevante de parámetros que comprende el indicador de longitud 23.

La Fig. 12 ilustra una realización particular de la etapa S61 de asociación de la Fig. 11. En esta realización la bandera de presencia de extensión en la cabecera de segmento es insertada en un conjunto de parámetros, por ejemplo un PPS, en la etapa S70. Preferentemente, un identificador del conjunto de parámetros 26 que identifica el conjunto de parámetros que comprende la bandera de presencia de extensión en la cabecera de segmento es asociado con la representación codificada 20 en la etapa S71, tal como insertando el identificador del conjunto de parámetros 26 en la cabecera de segmento 21 de la representación codificada 20.

En un enfoque alternativo, la bandera de presencia de extensión en la cabecera de segmento es insertada en la cabecera de segmento 21.

La Fig. 13 es un diagrama esquemático de bloques de un decodificador 40 según una realización. El decodificador 40 comprende una unidad de análisis 41, también denominada analizador o medio o módulo de análisis, configurada para analizar un indicador de longitud asociado con una representación codificada de un segmento de una imagen. El indicador de longitud es indicativo de una longitud de un campo de extensión presente en una cabecera de segmento de la representación codificada. El decodificador 40 también comprende una unidad de procesamiento 42, también denominada procesador o medio o módulo de procesamiento, configurada para hacer que el decodificador 40 ignore, durante la decodificación de la representación codificada, cualquier valor del campo de extensión identificado en función del indicador de longitud.

En una realización opcional, el decodificador 40 también comprende una unidad de determinación 43, también denominada determinador o medio o módulo de determinación, configurada para determinar la longitud del campo de extensión en función del indicador de longitud. Una unidad opcional de identificación 44, también denominada identificador o medio o módulo de identificación, del decodificador 40 está configurada, preferentemente, para identificar el fin del campo de extensión en la cabecera de segmento en función de la longitud determinada por la unidad de determinación 43.

Preferentemente, la unidad de análisis 41 está configurada para analizar el indicador de longitud presente en la cabecera de segmento de la representación codificada. Alternativamente, la unidad de análisis 41 podría analizar el indicador de longitud en un conjunto de parámetros identificado en función de un identificador del conjunto de parámetros analizado en la cabecera de segmento de la representación codificada. En este caso, la unidad de análisis 41, si el conjunto de parámetros con el indicador de longitud ya ha sido analizado y decodificado para una representación codificada anteriormente, podría recuperar de una memoria asociada el indicador de longitud, según se ha divulgado anteriormente en la presente memoria.

En una realización adicional, la unidad de procesamiento 42 está configurada para hacer que el decodificador 40 ignore, durante la decodificación de la representación codificada, cualquier valor del campo de extensión que sigue directamente al indicador de longitud en la cabecera de segmento y hasta el fin del campo de extensión identificado en función del indicador de longitud, por ejemplo mediante la unidad de identificación 44.

La unidad de análisis 41 también está configurada, preferentemente, para analizar una bandera de la extensión de cabecera de segmento asociada con la representación codificada. En tal caso, la unidad de análisis 41 está configurada para analizar el indicador de longitud y la unidad de procesamiento 42 está configurada para hacer que el decodificador 40 ignore cualquier valor del campo de extensión si la bandera de la extensión de cabecera de segmento tiene un valor predefinido, preferentemente 1_{bin}.

En una realización opcional, el decodificador 40 comprende, además, una unidad 45 de identificación de conjuntos, también denominada identificador de conjuntos o medio o módulo de identificación de conjuntos, configurada para identificar un conjunto de parámetros, preferentemente un PPS, asociado con la representación codificada en función de un identificador del conjunto de parámetros, preferentemente un identificador de PPS, obtenible en función de la cabecera de segmento.

Preferentemente, en una realización particular, la unidad 45 de identificación de conjuntos recupera el identificador del conjunto de parámetros de la cabecera de segmento de la representación codificada y lo usa para identificar el conjunto relevante de parámetros que lleva la bandera de presencia de extensión en la cabecera de segmento. En tal caso, la unidad de análisis 41 está configurada para analizar la bandera de presencia de extensión en la cabecera de segmento presente en el conjunto de parámetros, preferentemente un PPS, identificado por la unidad 45 de identificación de conjuntos.

La unidad de identificación 44 anteriormente mencionada está configurada, en una realización opcional, para identificar el inicio de los datos de segmento en la representación codificada en función del fin del campo de extensión identificado en función del indicador de longitud, por ejemplo mediante la unidad de identificación 44. El decodificador 40 comprende, entonces, una unidad de decodificación 46 , también denominada decodificador de datos o medio o módulo de decodificación, configurada para analizar y decodificar los datos de segmento comenzando desde el inicio de los datos de segmento identificado por la unidad de identificación 44.

La unidad de decodificación 46 está configurada, preferentemente, para analizar y decodificar los datos de segmento ignorando y, por ello, no usando cualquier valor del campo de extensión en la cabecera de segmento. Por ende, la unidad de decodificación 46 lleva a cabo la decodificación de la representación codificada 20 sin usar ningún valor en el campo de extensión o representado por el mismo.

En una realización particular el decodificador 40 comprende una unidad 41 para analizar una bandera que indica si se usa un indicador de longitud para indicar la longitud de la cabecera de segmento. Además, el decodificador 40 comprende una unidad 41 para analizar el indicador de longitud y, en función de eso, determinar la longitud de un campo de extensión en la cabecera de segmento. El decodificador 40 puede entonces usar esa información cuando decodifica la unidad NAL codificada.

La Fig. 14 es un diagrama esquemático de bloques de un decodificador 90 según otra realización para decodificar una representación codificada de un segmento de una imagen. El decodificador 90 comprende una unidad de análisis 91, también denominada analizador o medio o módulo de análisis, configurada para analizar un indicador de longitud asociado con la representación codificada. El indicador de longitud es indicativo de una longitud de un campo de extensión presente en una cabecera de segmento de la representación codificada. Una unidad de identificación del campo de extensión 92, también denominada identificador de campo de extensión o medio o módulo de identificación, está configurada para identificar el campo de extensión en la cabecera de segmento en función del indicador de longitud analizado por la unidad de análisis 91. El decodificador 90 también comprende una unidad de decodificación 93, también denominada decodificador de datos o medio o módulo de decodificación. Esta unidad de decodificación 93 está configurada para decodificar la representación codificada del segmento en función de al menos un valor del campo de extensión en la cabecera de segmento identificado por la unidad de identificación del campo de extensión 92.

Los decodificadores 40, 90 de las Figuras 13 y 14, con sus unidades 41-46, 91-93 de inclusión, podrían ser implementados en soporte físico. Hay numerosas variantes de elementos de circuitería que pueden ser usados y combinados para lograr las funciones de las unidades 41-46, 91-93 del decodificador 40, 90. Tales variantes están abarcadas por las realizaciones. Ejemplos particulares de una implementación de soporte físico del decodificador 40, 90 es la implementación en un soporte físico procesador de señales digitales (DSP) y en tecnología de circuitos integrados, que incluye tanto circuitería electrónica de uso general como circuitería para aplicaciones específicas.

El decodificador 50 descrito en la presente memoria podría ser implementado de forma alternativa, por ejemplo, por uno o más de un procesador 52 y soporte lógico adecuado con un almacenamiento o memoria 54 adecuado al efecto, un dispositivo lógico programable (*Programmable Logic Device*, PLD) u otro u otros componentes electrónicos, según se muestra en la Fig. 15. Además, el decodificador 50 comprende, preferentemente, una entrada o unidad 51 de entrada configurada para recibir las representaciones codificadas de segmentos, tales como en forma de unidades NAL. Una correspondiente salida o unidad 53 de salida está configurada para dar salida a los segmentos decodificados.

El decodificador 32 puede estar situado, por ejemplo, en un receptor , tal como en una videocámara, un convertidor de señal de TV o un dispositivo de visualización, por ejemplo en un dispositivo móvil, según se muestra en la Fig. 16. El receptor comprende, entonces, una entrada o unidad 31 de entrada configurada para recibir un flujo de bits codificado, tal como paquetes de datos de unidades NAL, según se muestra en la Fig. 2. Las representaciones codificadas de las unidades NAL son decodificadas por el decodificador 32 según se divulga en la presente memoria. Preferentemente, el decodificador 32 comprende una memoria intermedia 34 de imágenes de referencia, o está conectado a la misma, que almacena temporalmente imágenes ya decodificadas que han de usarse como imágenes de referencia para otras imágenes en el flujo de vídeo. El receptor da salida a las imágenes decodificadas, tal como desde la memoria intermedia 34 de imágenes de referencia, por medio de una salida o unidad 33 de salida. Estas imágenes que salen son enviadas para ser presentadas a un usuario

en una pantalla o dispositivo de visualización del receptor o conectada con el mismo, incluso inalámbricamente.

Por lo tanto, una realización versa sobre un receptor que comprende un decodificador para decodificar una representación codificada de un segmento de una imagen. La representación codificada comprende una cabecera de segmento y datos de segmento. El decodificador del receptor comprende una unidad de análisis configurada para analizar un indicador de longitud asociado con la representación codificada e indicativo de una longitud de un campo de extensión en la cabecera de segmento. El decodificador también comprende una unidad de procesamiento configurada para hacer que el decodificador ignore, durante la decodificación de la representación codificada, cualquier valor del campo de extensión identificado en función del indicador de longitud.

Otra realización versa sobre un receptor que comprende un decodificador para decodificar una representación codificada de un segmento de una imagen. La representación codificada comprende una cabecera de segmento y datos de segmento. El decodificador del receptor comprende una unidad de análisis configurada para analizar un indicador de longitud asociado con la representación codificada e indicativo de una longitud de un campo de extensión en la cabecera de segmento. El decodificador también comprende una unidad de identificación del campo de extensión configurada para identificar el campo de extensión en la cabecera de segmento en función del indicador de longitud. Una unidad de decodificación del decodificador está configurada para decodificar la representación codificada del segmento en función de al menos un valor del campo de extensión en la cabecera de segmento identificado por la unidad de identificación del campo de extensión.

La Fig. 17 es un diagrama esquemático de bloques de un codificador 70 configurado para codificar un segmento de una imagen según una realización. El codificador 70 comprende una unidad 71 de generación, también denominada generador o medio o módulo de generación, configurada para generar un indicador de longitud indicativo de una longitud de un campo de extensión presente en una cabecera de segmento del segmento. Una unidad 72 de codificación, también denominada codificador o medio o módulo de codificación de segmentos, del codificador 70 está configurada para codificar el segmento creando una representación codificada del segmento que comprende la cabecera de segmento y datos de segmento. El codificador 70 también comprende una unidad 73 de asociación, también denominada asociador o medio o módulo de asociación, configurada para asociar el indicador de longitud con la representación codificada.

En una realización opcional, el codificador 70 comprende una unidad 74 de determinación, también denominada determinador o medio o módulo de determinación, configurada para determinar la longitud del campo de extensión presente en la cabecera de segmento. La unidad 71 de generación está configurada, entonces, para generar el indicador de longitud en función de la longitud del campo de extensión determinada por la unidad 74 de determinación.

Una realización opcional tiene una unidad 73 de asociación que está configurada para insertar el indicador de longitud en la cabecera de segmento, preferentemente antes de que la unidad 72 de codificación codifique el segmento. En tal caso, la unidad 73 de asociación inserta el indicador de longitud, preferentemente, inmediatamente antes del campo de extensión en la cabecera de segmento.

El codificador 70 comprende, en una realización opcional, una unidad 75 de configuración, también denominada configurador o medio o módulo de configuración, configurada para poner una bandera de presencia de extensión en la cabecera de segmento en un valor predefinido, preferentemente 1_{bin}, para indicar la presencia del campo de extensión en la cabecera de segmento. La unidad 73 de asociación está configurada, entonces, para asociar la bandera de presencia de extensión en la cabecera de segmento con la representación codificada. En una realización particular, la unidad 73 de asociación está configurada para insertar la bandera de presencia de extensión en la cabecera de segmento in a conjunto de parámetros, preferentemente un PPS. La unidad 73 de asociación está configurada, además, para asociar un identificador del conjunto de parámetros, preferentemente un identificador de PPS, que identifica el conjunto de parámetros, preferentemente PPS, con la representación codificada. En particular, la unidad 73 de asociación está configurada para insertar el identificador del conjunto de parámetros, preferentemente un identificador de PPS, en la cabecera de segmento.

En una realización particular, el codificador 70 comprende una unidad 74 para determinar la longitud de un campo de extensión en la cabecera de segmento y una unidad 73 para insertar un indicador de longitud.

El codificador 70 de la Fig. 17, con sus unidades 71-75 de inclusión, podría ser implementado en soporte físico. Hay numerosas variantes de elementos de circuitería que pueden ser usados y combinados para lograr las funciones de las unidades 71-75 del codificador 70. Tales variantes están abarcadas por las realizaciones. Ejemplos particulares de una implementación de soporte físico del codificador 70 es la

implementación en un soporte físico procesador de señales digitales (DSP) y en tecnología de circuitos integrados, que incluye tanto circuitería electrónica de uso general como circuitería para aplicaciones específicas.

El codificador 80 descrito en la presente memoria podría ser implementado de forma alternativa, por ejemplo, por uno o más de un procesador 82 y soporte lógico adecuado con un almacenamiento o memoria 84 adecuado al efecto, un dispositivo lógico programable (PLD) u otro u otros componentes electrónicos, según se muestra en la Fig. 18. Además, el codificador 80 comprende, preferentemente, una entrada o unidad 81 de entrada configurada para recibir las imágenes del flujo de vídeo. Una correspondiente salida o unidad 83 de salida está configurada para dar salida a las representaciones codificadas de los segmentos, preferentemente en forma de unidades NAL.

El codificador 62 puede estar situado, por ejemplo, en un transmisor 60 en una videocámara por ejemplo, en un dispositivo móvil, según se muestra en la Fig. 19. El transmisor 60 comprende, entonces, una entrada o unidad 61 de entrada configurada para recibir imágenes de un flujo de vídeo que han de ser codificadas. Las imágenes son codificadas por el codificador 62 según se divulga en la presente memoria. Las imágenes codificadas salen del transmisor 60 por una salida o unidad 63 de salida en forma de un flujo codificado de bits, tal como de unidades NAL o paquetes de datos que llevan tales unidades NAL, según se muestra en la Fig. 2.

Una realización versa sobre un transmisor 60 que comprende un codificador para codificar un segmento de una imagen. El codificador comprende una unidad de generación configurada para generar un indicador de longitud indicativo de una longitud de un campo de extensión presente en la cabecera de segmento del segmento. El codificador también comprende una unidad de codificación configurada para codificar el segmento creando una representación codificada del segmento que comprende la cabecera de segmento y datos de segmento. Una unidad de asociación del codificador está configurada para asociar el indicador de longitud con la representación codificada.

Las realizaciones se aplican a un decodificador, a un codificador y a cualquier elemento que opere en un flujo de bits, tal como un nodo de red o un elemento de red con conciencia de medios.

Ha de entenderse que la elección de unidades o módulos de interactuación, así como la denominación de las unidades, son únicamente para un fin ejemplar, y que pueden ser configurados de varias formas alternativas para poder ejecutar las acciones del procedimiento divulgado.

Debería hacerse notar también que las unidades o módulos descritos en esta divulgación han de ser considerados entidades lógicas y no necesariamente como entidades físicas separadas. Se apreciará que el alcance de la tecnología divulgada en la presente memoria abarca plenamente otras realizaciones que pueden llegar a ser obvias para los expertos en la técnica, y, en consecuencia, que no ha de limitarse el alcance de esta divulgación.

No se pretende que la referencia a un elemento en singular signifique “uno y solo uno”, a no ser que ello sea afirmado explícitamente, sino más bien "uno o más." Además, para que esté abarcado por la misma, no es necesario que un dispositivo o un método aborden cada uno de los problemas, ni su totalidad, cuya solución se busca mediante la tecnología dada a conocer en la presente memoria.

En la descripción precedente, con fines de explicación y no de limitación, se definen detalles específicos, tales como arquitecturas particulares, interfaces, técnicas, etc., para proporcionar una comprensión cabal de la tecnología divulgada. Sin embargo, resultará evidente para los expertos en la técnica que la tecnología dada a conocer puede ser puesta en práctica en otras realizaciones y/o combinaciones de realizaciones que se aparten de estos detalles específicos. Es decir, los expertos en la técnica serán capaces de idear diversas disposiciones que, aunque no estén descritas o mostradas explícitamente en la presente memoria, implementan los principios de la tecnología dada a conocer. En algunos casos se omiten descripciones detalladas de dispositivos, circuitos y métodos muy conocidos para no ofuscar con detalle innecesario la descripción de la tecnología divulgada. Se pretende que todas las declaraciones de la presente memoria que enumeran principios, aspectos y realizaciones de la tecnología dada a conocer, así como ejemplos específicos de la misma, abarquen los equivalentes tanto estructurales como funcionales de la misma. Además, se pretende que tales equivalentes incluyan tanto los equivalentes actualmente conocidos como los equivalentes desarrollados en el futuro; por ejemplo, cualesquiera elementos desarrollados que lleven a cabo la misma función, con independencia de su estructura.

Así, por ejemplo, los expertos en la técnica apreciarán que los diagramas de bloque del presente documento pueden representar vistas conceptuales de circuitería ilustrativa u otras unidades funcionales que implementen los principios de la tecnología. De manera similar, se apreciará que cualquier diagrama de flujo, diagrama de transición de estado, pseudocódigo y similares representan diversos procedimientos que pueden ser sustancialmente representados en un medio legible en ordenador y, por

ello, ejecutados por un ordenador o procesador, con independencia de que tal ordenador o procesador sea o no mostrado explícitamente.

Las funciones de los diversos elementos, incluyendo los bloques funcionales, pueden ser proporcionadas a través del uso de soporte físico, tal como soporte físico de circuitos, y/o soporte físico capaz de ejecutar soporte lógico en forma de instrucciones codificadas almacenadas en un medio legible en ordenador. Así, ha de entenderse que tales funciones y tales bloques funcionales ilustrados están implementados ya sea mediante soporte físico y/o por ordenador, y que, así, están implementados por máquina.

La Fig. 20 es una ilustración esquemática de un ordenador 100 que comprende un procesador 110 y un medio 120 legible en ordenador que tiene en el mismo un programa informático 1.

En una realización, el programa informático 1 está configurado para decodificar una representación codificada de un segmento de una imagen. La representación codificada comprende una cabecera de segmento y datos de segmento. El programa informático 1 comprende un medio de código que, cuando es ejecutado por el procesador 110, hace que el procesador 110 analice un indicador de longitud asociado con la representación codificada e indicativo de una longitud de un campo de extensión en la cabecera de segmento. También se hace que el procesador 110 determine ignorar, durante la decodificación de la representación codificada, cualquier valor de dicho campo de extensión identificado en función del indicador de longitud.

En otra realización, el programa informático 1 está configurado para decodificar una representación codificada de un segmento de una imagen. La representación codificada comprende una cabecera de segmento y datos de segmento. El programa informático 1 comprende un medio de código que, cuando es ejecutado por el procesador 110, hace que el procesador 110 analice un indicador de longitud asociado con la representación codificada e indicativo de una longitud de un campo de extensión en la cabecera de segmento. También se hace que el procesador 110 identifique, en función del indicador de longitud, el campo de extensión en dicha cabecera de segmento y decodifique la representación codificada del segmento en función de al menos un valor del campo de extensión identificado en función del indicador de longitud.

En una realización adicional, el programa informático 1 está configurado para codificar un segmento de una imagen. El programa informático 1 comprende un medio de código que, cuando es ejecutado por el procesador 110, hace que el procesador 110 genere un indicador de longitud indicativo de una longitud de un campo de extensión presente en una cabecera de segmento del segmento. También se hace que el procesador 110 codifique el segmento creando una representación codificada del segmento que comprende la cabecera de segmento y datos de segmento. Además, se hace que el procesador 110 asocie el indicador de longitud con la representación codificada.

Una realización versa sobre un producto de programa informático que comprende un medio 120 legible en ordenador y un programa informático 1 definido según cualquiera de las realizaciones dadas a conocer anteriormente almacenado en el medio 120 legible en ordenador.

En términos de una implementados por soporte físico, los bloques funcionales pueden incluir o abarcar, sin limitación, soporte físico procesador de señales digitales (DSP), un procesador con un conjunto reducido de instrucciones, circuitería de soporte físico (por ejemplo, digital o analógica), incluyendo, sin limitación, uno o varios circuitos integrados para aplicaciones específicas (ASIC) y (cuando sea apropiado), máquinas de estado capaces de llevar a cabo tales funciones.

Las realizaciones descritas en lo que antecede han de ser entendidas como algunos ejemplos ilustrativos de la presente invención. Los expertos en la técnica entenderán que pueden realizarse diversos cambios, modificaciones y combinaciones a las realizaciones sin apartarse del alcance de la presente invención. En particular, soluciones de partes diferentes de las diferentes realizaciones pueden ser combinadas en otras configuraciones, cuando sea técnicamente posible. El alcance de la presente invención, sin embargo, está definido por las reivindicaciones adjuntas.

REIVINDICACIONES

1. Un receptor (30) que comprende una unidad de entrada (31) configurada para recibir una representación codificada (20) de un segmento (3) de una imagen (2), un decodificador (40, 50) para decodificar la representación codificada (20) y una unidad de salida (33) configurada para dar salida a imágenes decodificadas por el decodificador (40), comprendiendo dicha representación codificada (20) una cabecera de segmento (21) y datos de segmento (25), y comprendiendo dicho decodificador (40):

una unidad de identificación de conjuntos (45) configurada para identificar un conjunto de parámetros asociado con dicha representación codificada (20) en base a un identificador de conjunto de parámetros (26) obtenible en función de dicha cabecera de segmento (21);

una unidad de análisis (41) configurada para analizar una bandera de presencia de extensión en la cabecera de segmento presente en dicho conjunto de parámetros y analizar, si dicha bandera de presencia de extensión en la cabecera de segmento tiene un valor predefinido, un indicador de longitud (23) en un código de longitud variable asociado con dicha representación codificada (20) e indicativo de una longitud en bits o bytes de un campo de extensión (24) en dicha cabecera de segmento (21); y

una unidad de procesamiento (42) configurada para hacer que, si dicha bandera de presencia de extensión en la cabecera de segmento tiene dicho valor predefinido, dicho decodificador (40) ignore, durante la decodificación de dicha representación codificada (20), cualquier valor de dicho campo de extensión (24) identificado en función de dicho indicador de longitud (23).

2. El receptor (30) según la reivindicación 1 que, además, comprende:

una unidad de determinación (43) configurada para determinar una longitud de dicho campo de extensión (24) en función de dicho indicador de longitud (23); y

una unidad de identificación (44) configurada para identificar un final de dicho campo de extensión (24) en dicha cabecera de segmento (21) en función de dicha longitud.

3. El receptor (30) según la reivindicación 1, en el que dicha unidad de análisis (41) está configurada para analizar, si dicha bandera de presencia de extensión en la cabecera de segmento tiene dicho valor predefinido, dicho indicador de longitud (23) que está presente en dicha cabecera de segmento (21).

4. El receptor según la reivindicación 3, en el que dicha unidad de procesamiento (42) está configurada para hacer que, si dicha bandera de presencia de extensión en la cabecera de segmento tiene dicho valor predefinido, dicho decodificador (40) ignore, durante la decodificación de dicha representación codificada (20), cualquier valor de dicho campo de extensión (24) que siga directamente a dicho indicador de longitud (23) en dicha cabecera de segmento (21) hasta un final de dicho campo de extensión (24) identificado en función de dicho indicador de longitud (23).

5. El receptor según la reivindicación 1 o 2, en el que dicha unidad de análisis (41) está configurada para analizar, si dicha bandera de presencia de extensión en la cabecera de segmento tiene dicho valor predefinido, dicho indicador de longitud (23) presente en un conjunto de parámetros identificado en función de un identificador de conjunto de parámetros (26) obtenible en función de dicha cabecera de segmento (21).

6. El receptor según cualquiera de las reivindicaciones 1 a 5, que comprende además:

una unidad de identificación (44) configurada para identificar un inicio de dichos datos de segmento (25) en función de un final de dicho campo de extensión (24) identificado en función del identificador de longitud (23); y

una unidad de decodificación (46) configurada para analizar y decodificar dichos datos de segmento (25) comenzando desde dicho inicio de dichos datos de segmento (25).

7. El receptor según la reivindicación 6, en el que dicha unidad de decodificación (46) está configurada para analizar y decodificar dichos datos de segmento ignorando dicho valor cualquiera de dicho campo de extensión (24) en dicha cabecera de segmento (21).

8. El receptor según cualquiera de las reivindicaciones 1 a 7, en el que el receptor está ubicado en una cámara de vídeo, un decodificador de señal de TV (set-top-box) o un visualizador, por ejemplo en un dispositivo móvil.

9. Un transmisor (60) que comprende una unidad de entrada (61) configurada para recibir una imagen (2) de un flujo de video a codificar, un codificador (70) para codificar un segmento (3) de la imagen (2), y una unidad de salida (63) configurada para dar salida a la imagen codificada, comprendiendo dicho codificador (70):

una unidad de configuración (75) configurada para poner una bandera de presencia de extensión en la cabecera de segmento a un valor predefinido para indicar la presencia de un campo de extensión (24) en una cabecera de segmento (21) de dicho segmento (3);

una unidad de generación (71) configurada para generar un indicador de longitud (23) en un código de longitud variable indicativo de una longitud en bits o bytes de dicho campo de extensión (24) presente en dicha cabecera de segmento (21) de dicho segmento (3);

una unidad de codificación (72) configurada para codificar dicho segmento (3) en forma de una representación codificada (20) de dicho segmento (3) que comprende dicha cabecera de segmento (21) y dichos datos de segmento (25); y

una unidad de asociación (73) configurada para asociar dicho indicador de longitud (23) con dicha representación codificada (20), insertar dicha bandera de presencia de extensión en la cabecera de segmento en un conjunto de parámetros y asociar un identificador del conjunto de parámetros, que identifica a dicho conjunto de parámetros, con dicha representación codificada (20).

10. El transmisor según la reivindicación 9, que comprende además:

una unidad de determinación (74) configurada para determinar una longitud de dicho campo de extensión (24) presente en dicha cabecera de segmento (21), en el que dicha unidad de generación (71) está configurada para generar dicho indicador de longitud (23) en función de dicha longitud de dicho campo de extensión (24).

11. El transmisor según la reivindicación 9 o 10, en el que dicha unidad de asociación (73) está configurada para insertar dicho indicador de longitud (23) en dicha cabecera de segmento (21).

12. El transmisor según la reivindicación 11, en el que dicha unidad de asociación (73) está configurada para insertar dicho indicador de longitud (23) inmediatamente antes de dicho campo de extensión (24) en dicha cabecera de segmento (21).

13. El transmisor según la reivindicación 9 o 10, en el que dicha unidad de asociación (73) está configurada para insertar dicho indicador de longitud (23) en un conjunto de parámetros y asociar un identificador de conjunto de parámetros (26), que identifica a dicho conjunto de parámetros, con dicha representación codificada (20).

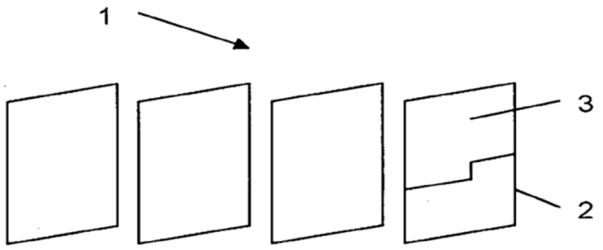

Fig. 1

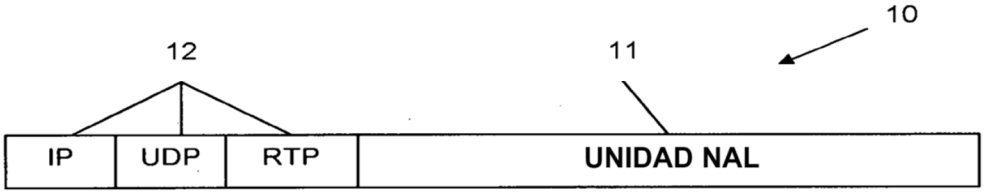

Fig. 2

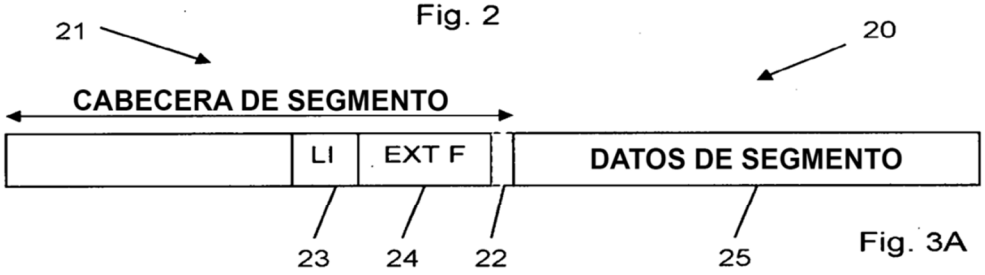

Fig. 3A

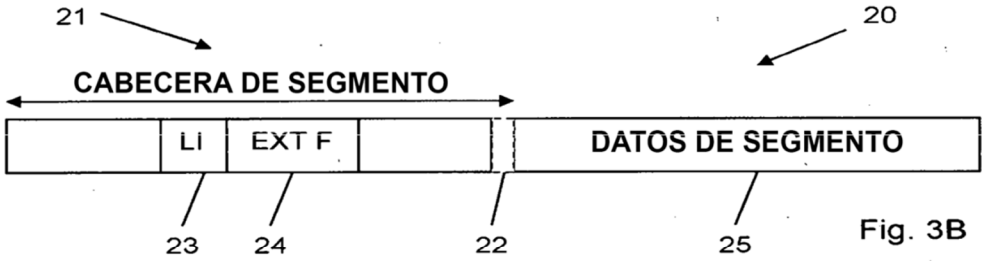

Fig. 3B

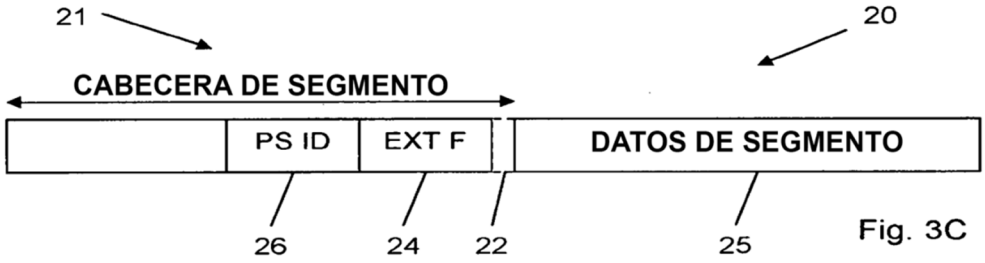

Fig. 3C

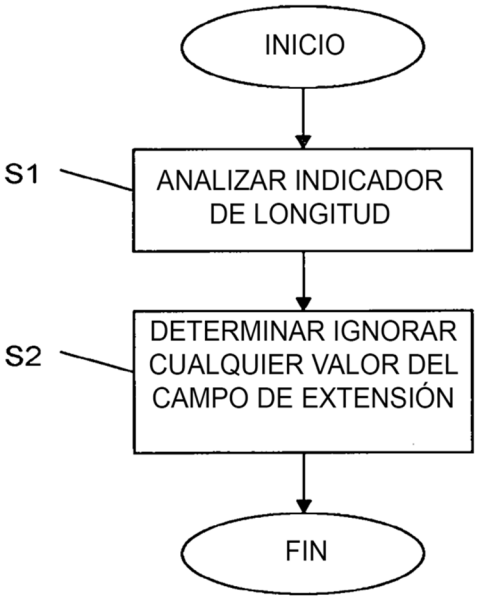

Fig. 4

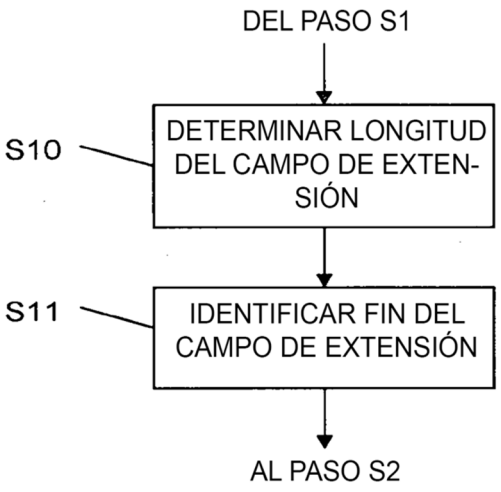

Fig. 5

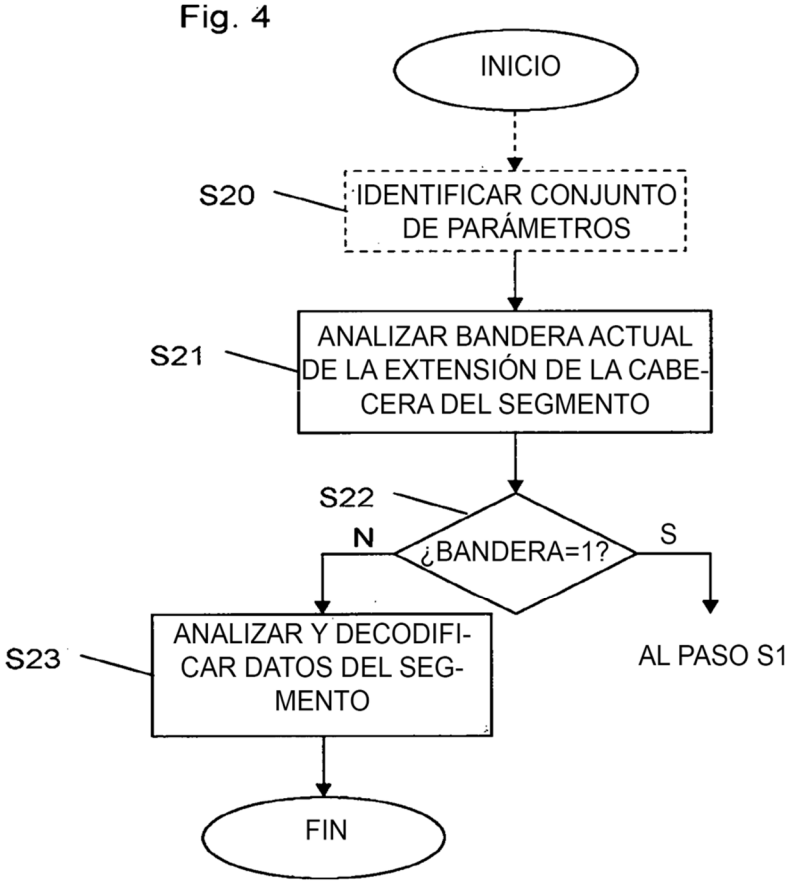

Fig. 6

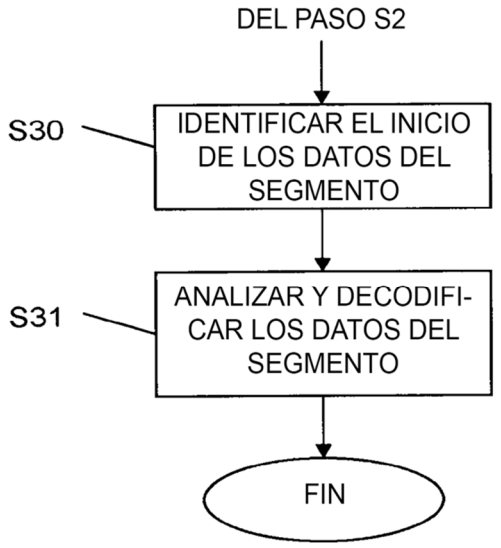

Fig. 7

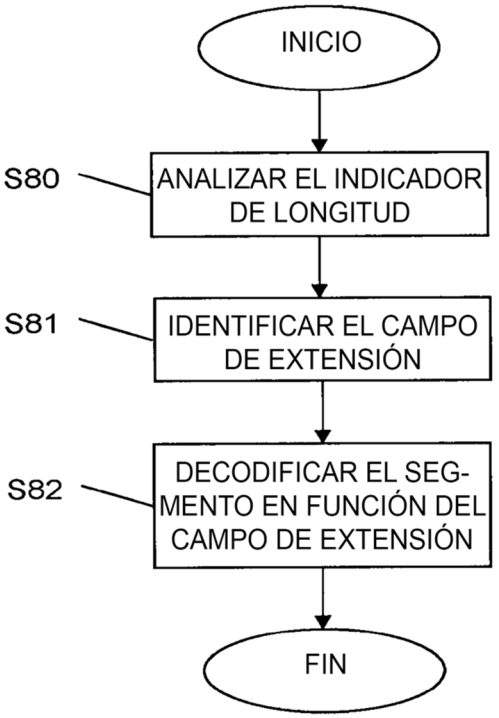

Fig. 8

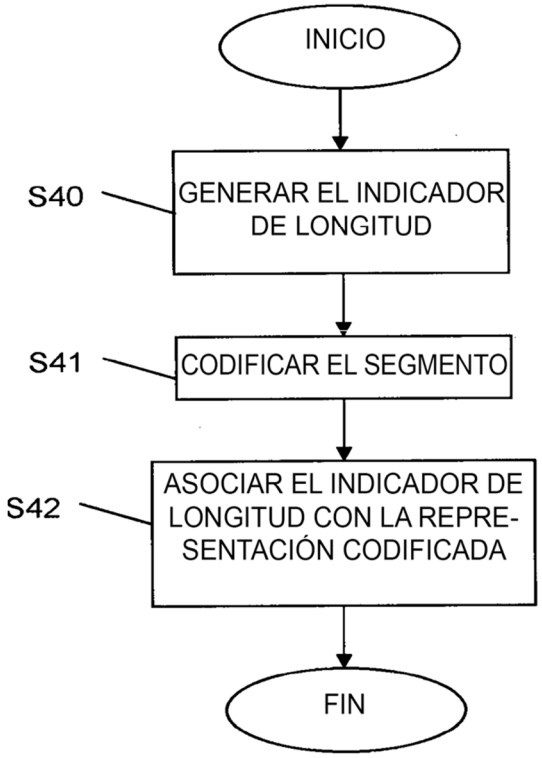

Fig. 9

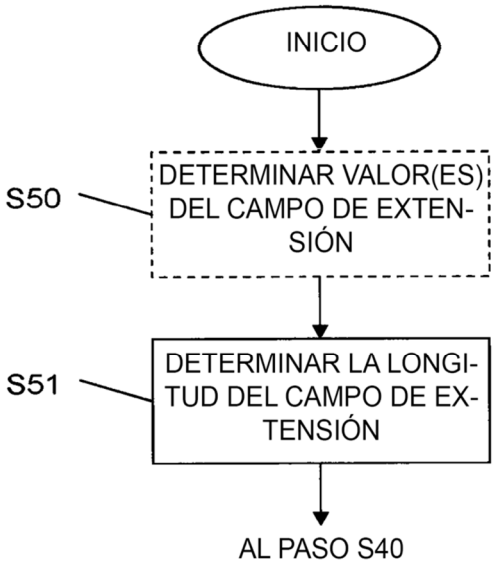

Fig. 10

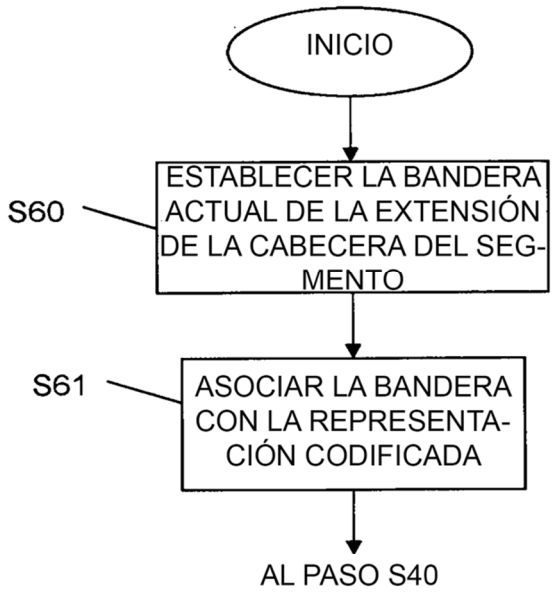

Fig. 11

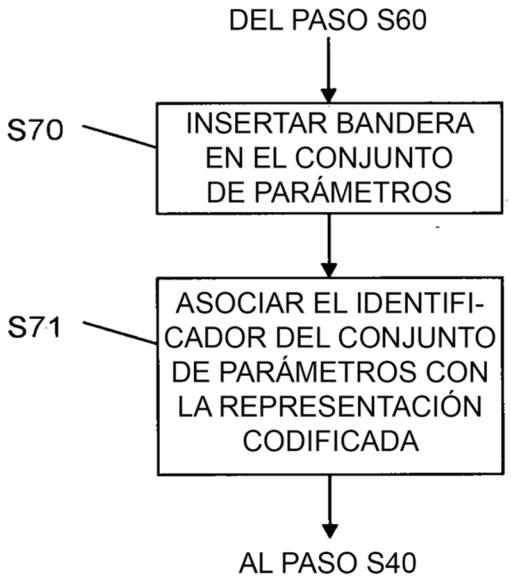

Fig. 12

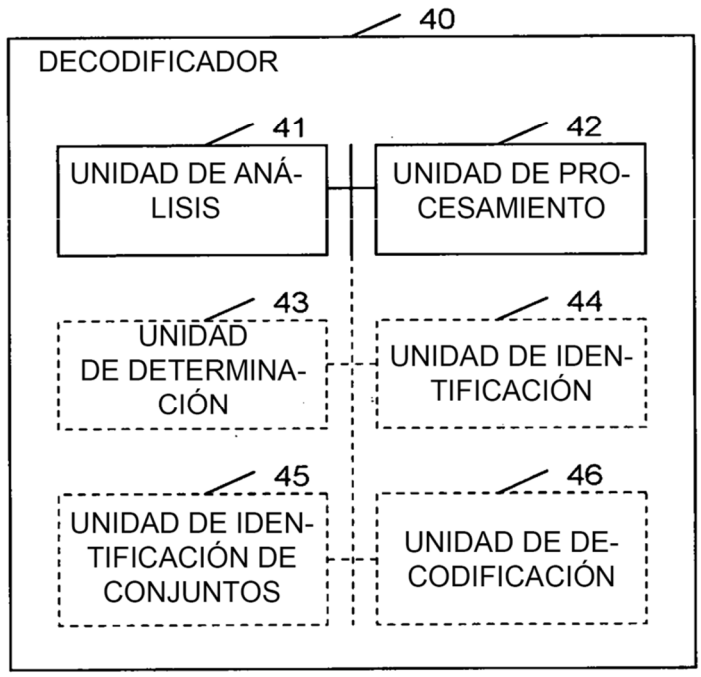

Fig. 13

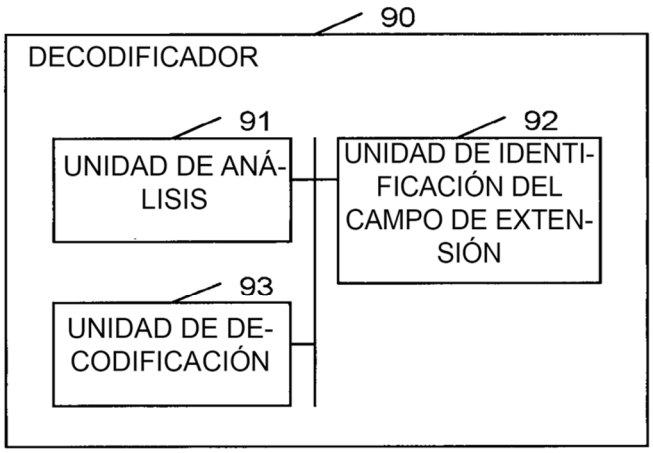

Fig. 14

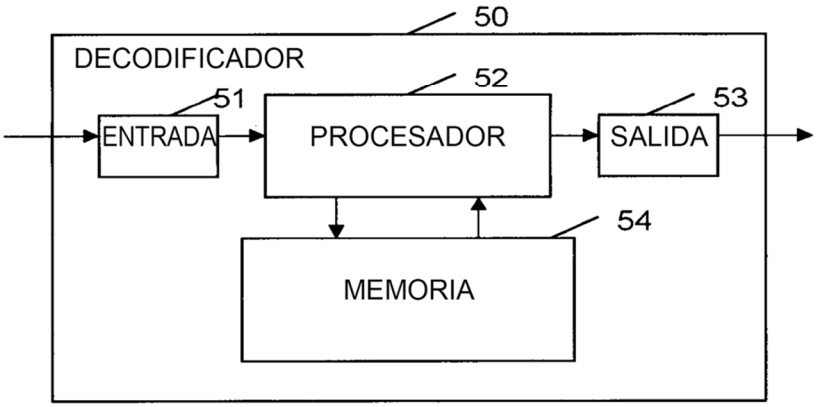

Fig. 15

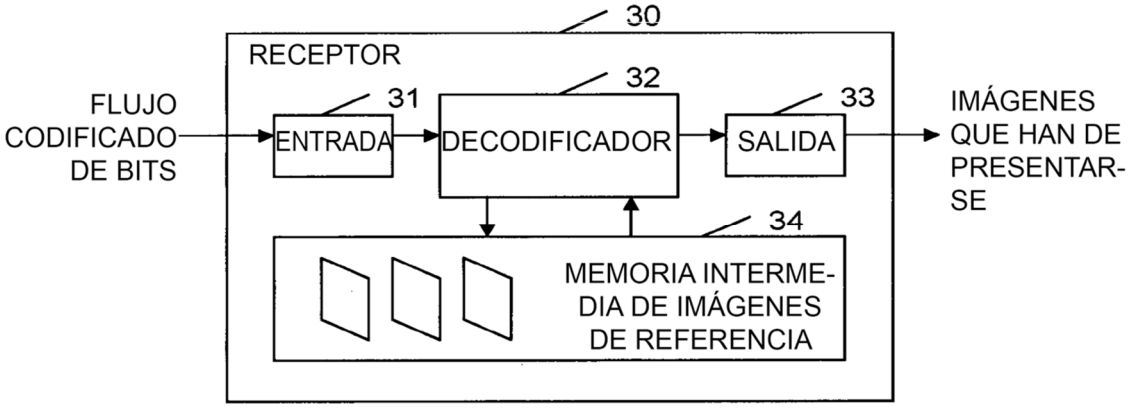

Fig. 16

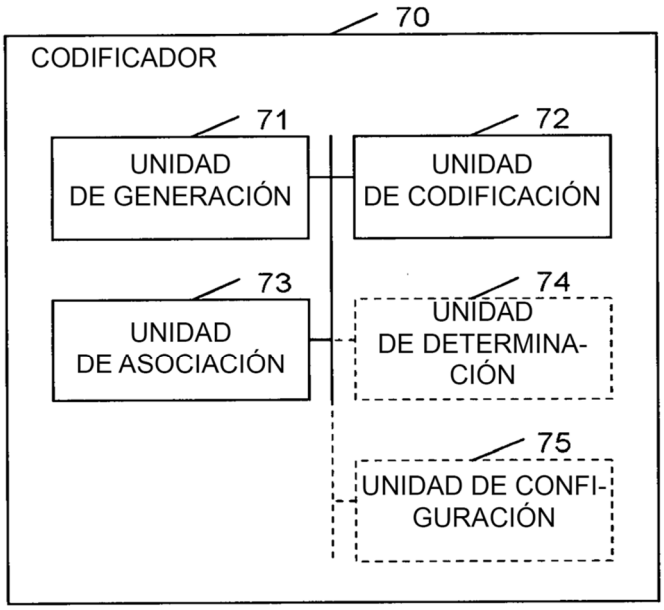

Fig. 17

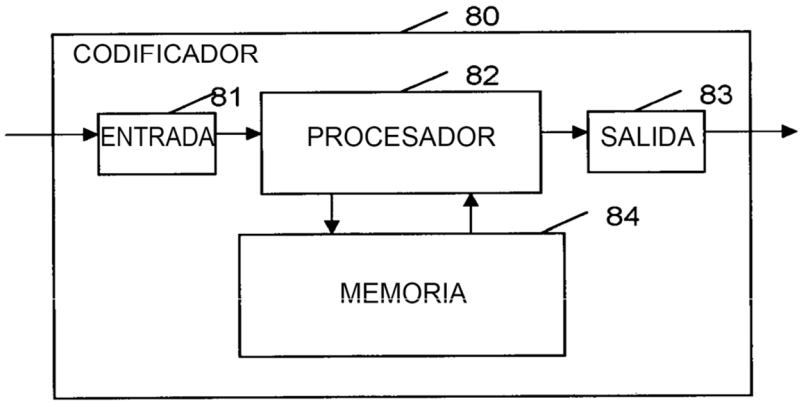

Fig. 18

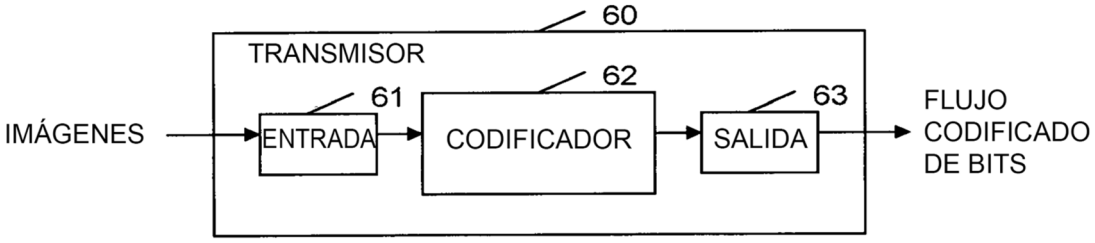

Fig. 19

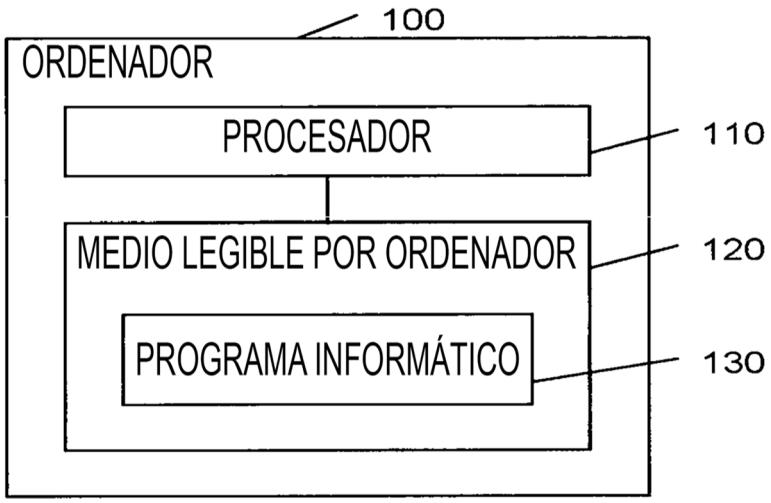
100
ORDENADOR
PROCESADOR
110
MEDIO LEGIBLE POR ORDENADOR
120
PROGRAMA INFORMÁTICO
130

Fig. 20