(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 767 933**

(51) Int. CI.:

H04N 19/174 (2014.01)
H04N 19/70 (2014.01)
H04N 19/46 (2014.01)
H04N 19/86 (2014.01)
H04N 19/82 (2014.01)
H04N 19/61 (2014.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **03.09.2004 PCT/US2004/029033**

(87) Fecha y número de publicación internacional: **24.03.2005 WO05027495**

(96) Fecha de presentación y número de la solicitud europea: **03.09.2004 E 04783323 (1)**

(97) Fecha y número de publicación de la concesión europea: **27.11.2019 EP 1656793**

(54) Título: **Capa de sectores en códec de vídeo**

(30) Prioridad:

07.09.2003 US 501081 P
03.09.2004 US 933960

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
19.06.2020

(73) Titular/es:

MICROSOFT TECHNOLOGY LICENSING, LLC (100.0%)
One Microsoft Way
Redmond, WA 98052, US

(72) Inventor/es:

REGUNATHAN, SHANKAR;
HSU, POHSIANG;
WANG, CE;
LIN, CHIH-LUNG, BRUCE;
LIANG, JIE y
SRINIVASAN, SRIDHAR

(74) Agente/Representante:

CARPINTERO LÓPEZ, Mario

DESCRIPCIÓN

Capa de sectores en códec de vídeo

Campo técnico

La presente invención se refiere a técnicas para codificar, decodificar y procesar digitalmente vídeo, imágenes y otros contenidos de medios digitales.

Antecedentes

El vídeo digital consume grandes cantidades de almacenamiento y capacidad de transmisión. Una secuencia de vídeo digital sin formato convencional incluye 15 o 30 fotogramas por segundo. Cada fotograma puede incluir decenas o cientos de miles de píxeles (también denominados pels). Cada píxel representa un pequeño elemento de la imagen. Básicamente, un ordenador representa comúnmente un píxel como un conjunto de tres muestras con un total de 24 bits. Por ejemplo, un píxel puede comprender una muestra de luminancia de 8 bits (también denominada muestra de luma) que define el componente de escala de grises del píxel y dos valores de muestra de crominancia de 8 bits (también denominada muestra de croma) que definen el componente de color del píxel. Por lo tanto, el número de bits por segundo, o la velocidad de bits, de una secuencia de vídeo digital sin procesar convencional puede ser de 5 millones de bits por segundo o más.

Muchos ordenadores y redes de ordenadores carecen de los recursos para procesar vídeo digital sin procesar. Por esta razón, los ingenieros usan la compresión (también denominada codificación o cifrado) para reducir la velocidad de bits del vídeo digital. La compresión disminuye el coste de almacenar y transmitir vídeo al convertir el vídeo a una forma de velocidad de bits más baja. La descompresión (también denominada decodificación) reconstruye una versión del vídeo original a partir de la forma comprimida. Un "códec" es un sistema codificador/decodificador. La compresión puede ser sin pérdidas, en la que la calidad del vídeo no sufre, pero las disminuciones en la velocidad de bits se ven limitadas por la cantidad inherente de variabilidad (a veces denominada entropía) de los datos de vídeo. O bien, la compresión puede ser con pérdida, en la que la calidad del vídeo sufre, pero las reducciones alcanzables en la velocidad de bits son más dramáticas. La compresión con pérdida se usa a menudo junto con la compresión sin pérdida: en un diseño de sistema en el que la compresión con pérdida establece una aproximación de la información y se aplican técnicas de compresión sin pérdida para representar la aproximación.

Por lo general, las técnicas de compresión de vídeo incluyen la compresión "intra-imagen" y la compresión "entre-imagen", en la que una imagen es, por ejemplo, un fotograma de vídeo escaneado progresivamente, un fotograma de vídeo entrelazado (que tiene líneas alternas para campos de vídeo) o un campo de vídeo entrelazado. Para fotogramas progresivos, las técnicas de compresión intra-imagen comprimen fotogramas individuales (convencionalmente denominados fotogramas I o fotogramas clave), y las técnicas de compresión inter-imagen comprimen fotogramas (convencionalmente denominados fotogramas predichos, fotogramas P o fotogramas B) con referencia a los fotogramas anteriores y/o siguientes (convencionalmente denominados fotogramas de referencia o de anclaje).

Los fotogramas predichos pueden dividirse en regiones denominadas macrobloques. Una región coincidente en un fotograma de referencia para un macrobloque particular se especifica enviando información del vector de movimiento al macrobloque. Un vector de movimiento indica la ubicación de la región en el fotograma de referencia cuyos píxeles se utilizarán como predictor del macrobloque actual de píxeles. La diferencia píxel por píxel, a menudo denominada señal de error o residual, entre el macrobloque actual (o los bloques del mismo) y el predictor de macrobloque se deriva. Esta señal de error tiene por lo general una entropía más baja que la señal original. Por lo tanto, la información puede codificarse a una velocidad menor. Un codificador realiza la estimación de movimiento determinando un vector de movimiento para una región de un fotograma buscando una región coincidente en uno o más fotogramas de referencia para usarse como predictor. Un codificador o decodificador realiza la compensación de movimiento aplicando el vector de movimiento para encontrar el predictor en uno o más fotogramas de referencia.

El valor del vector de movimiento para un macrobloque se correlaciona a menudo con los vectores de movimiento para macrobloques espacialmente circundantes. Por lo tanto, la compresión de los datos utilizados para transmitir la información del vector de movimiento se puede lograr codificando el diferencial entre el vector de movimiento y un predictor del vector de movimiento formado a partir de vectores de movimiento vecinos.

A menudo, en las técnicas de compresión de vídeo, los bloques de píxeles u otros datos de vídeo de dominio espacial, como los restos, se transforman en datos de dominio de transformación, que a menudo son datos de dominio de frecuencia (es decir, espectrales). Los bloques resultantes de coeficientes de datos espectrales se pueden cuantificar y la entropía se codifica después.

Cuando los datos se descomprimen antes de que se muestre el vídeo resultante, un decodificador realiza por lo general la inversa de las operaciones de compresión. Por ejemplo, un decodificador puede realizar la decodificación de entropía, cuantización inversa y una transformación inversa mientras descomprime los datos. Cuando se usa la compensación de movimiento, el decodificador (y el codificador) reconstruyen un fotograma a partir de uno o más fotogramas previamente reconstruidos (que ahora se usan como fotogramas de referencia), y el fotograma recientemente reconstruido se puede usar como un fotograma de referencia para la compensación de movimiento

para próximos fotogramas.

Muchos escenarios de uso típicos para vídeo digitalmente codificado implican la transmisión del vídeo codificado entre dispositivos y con frecuencia entre ubicaciones geográficamente distantes. Además, muchos sistemas de transmisión de datos de uso común utilizan protocolos de transmisión basados en paquetes, en los que una transmisión de datos se divide en unidades enrutadas por separado denominadas "paquetes". Estos diversos sistemas de transmisión que transportan vídeo digital a menudo están sujetos a ruido y otras fuentes de errores de transmisión, y pueden experimentar "pérdida de paquetes". Tales errores y la pérdida de paquetes pueden conducir a un fallo en la decodificación de un fotograma individual, o múltiples fotogramas relacionados de la secuencia de vídeo.

Por lo tanto, puede ser deseable codificar regiones parciales de una imagen en una secuencia de vídeo como una unidad independientemente decodificable. Esto ayuda a habilitar la paquetización de la transmisión de vídeo. Además, esto introduce redundancia adicional en el flujo de bits de vídeo comprimido que aumenta su resistencia a los errores de transmisión y la pérdida de paquetes. Por ejemplo, la pérdida de decodificación por un error de transmisión o un paquete perdido puede limitarse a la región parcial, en lugar de a una imagen completa de la secuencia de vídeo. Sin embargo, esta resistencia se logra a costa de la eficacia de compresión.

Numerosas empresas han producido códecs de vídeo. Por ejemplo, Microsoft Corporation ha producido un codificador y decodificador de vídeo lanzado para Windows Media Video 8. Además de estos productos, numerosos estándares internacionales especifican aspectos de decodificadores de vídeo y formatos para información de vídeo comprimida. Estos estándares incluyen los estándares H.261, MPEG-1, H.262, H.263 y MPEG-4. Directamente o por implicación, estos estándares también especifican ciertos detalles del codificador, pero no se especifican otros detalles del codificador. Estos productos y estándares usan (o apoyan el uso de) diferentes combinaciones de las técnicas de compresión y descompresión descritas anteriormente. En particular, estos productos y estándares proporcionan diversas técnicas para la codificación de unidades de imágenes parciales.

Una de esas técnicas divide un fotograma dentro de la secuencia de vídeo en sectores. Un sector se define para contener una o más filas contiguas de macrobloques en su orden original de izquierda a derecha. Un sector comienza en el primer macrobloque de una fila y termina en el último macrobloque de la misma u otra fila.

Diversos estándares, por ejemplo, MPEG-1, MPEG-2, H.263 (con los GOB siendo aproximadamente equivalentes a sectores o con el modo de codificación estructurado de sectores del Anexo K), MPEG-4 parte 2 y H.264/JVT/MPEG-4part10, todos tienen sectores como parte de su sintaxis. Entre estos, todos deshabilitan la predicción intra y la predicción del vector de movimiento y la mayoría de las otras formas de predicción a través de los límites de sectores por razones de robustez de pérdida/error. Entre estos, solo H.263 (Anexo J) y H.264 / JVT incluyen filtros de bucle. El manejo de H.263 del entrelazado es bastante primitivo (la codificación de campo solo utiliza indicaciones de mejora suplementarias del Anexo W). H.264 tiene una estructura de encabezado más resistente a errores y permite que el codificador seleccione si se aplicará o no el filtro de bucle a través de los límites de sector.

La implementación de sectores en estos diversos estándares de decodificación de vídeo consigue cada uno un equilibrio diferente entre resistencia y eficacia de codificación.

Yao Wang *et al.*: "Comunicaciones de vídeo en tiempo real a través de redes poco confiables", revista de procesamiento de señales IEEE, centro de servicio IEEE, Piscataway, NJ, EE. UU., Vol. 17, n.º 4, 1 de julio de 2000, páginas 61 a 82 se refieren a técnicas de codificación de vídeo resistentes a errores. El mismo desvela herramientas de resincronización para MPEG-4. Se afirma que el enfoque de paquete de vídeo para la resincronización en MPEG-4 es muy similar en principio al sector adaptativo de MPEG-2 y al modo estructurado por sectores de H.263. Cuando se utiliza el enfoque de paquete de vídeo para la resincronización, las longitudes de los paquetes de vídeo ya no se basan en el número de Macrobloques (MB) como en el caso del modo resistente sin errores de MPEG-4 o la línea de base H.263, sino en cambio en el número de bits contenidos en ese paquete. El marcador de resincronización colocado al comienzo de un nuevo paquete de vídeo es distinguible de todas las palabras de código VLC posibles. La información del encabezado se proporciona también al comienzo de un paquete de vídeo. Este encabezado contiene la información necesaria para reiniciar el procedimiento de decodificación, incluida la dirección del primer MB contenido en este paquete y el parámetro de cuantificación (QP) necesario para decodificar ese primer MB. El número MB proporciona la resincronización espacial necesaria, mientras que el QP permite resincronizar el procedimiento de decodificación diferencial. Después del QP está el código de extensión del encabezado (HEC). Como su nombre lo indica, el HEC es un solo bit para indicar si habrá información adicional de nivel de plano de objeto de vídeo (VOP) en este encabezado. Si el HEC es igual a uno, la siguiente información adicional (que es constante para cada VOP y se transmite con el encabezado de VOP) está disponible en este encabezado de paquete: información de temporización, referencia temporal, tipo de predicción de VOP y alguna otra información. La función de extensión del encabezado permite que el decodificador utilice correctamente los datos contenidos en el paquete actual sin referencia al paquete que contiene el encabezado VOP.

"Recomendación H.263: Codificación de vídeo para comunicación de baja velocidad de bits", la recomendación ITU-T H.263, 1 de febrero de 1998, páginas 1 a 167 define el estándar H.263. Define grupos de bloques (GOB) y sectores, así como sus correspondientes estructuras de codificación. También se describe un modo de predicción avanzado que comprende en particular una compensación de movimiento superpuesta para la luminancia.

Sumario

Un códec de vídeo y una sintaxis de flujo de bits descritos en el presente documento incluyen una capa de sectores diseñada para ser flexible y proporcionar una combinación efectiva de resistencia a errores y eficacia de compresión. Esta capa de sectores proporciona las siguientes características clave:

a) un mecanismo eficaz de direccionamiento de sector que funciona con procedimientos de codificación progresivos, de fotogramas entrelazados y campo entrelazado,
b) un mecanismo flexible y eficaz para retransmitir el encabezado de la imagen en la capa de sectores, y
c) decodificar la independencia deshabilitando todas las formas de predicción, superposición y filtrado de bucles a través de los límites de sector, de modo que un sector codificado en modo intra puede reconstruirse sin errores, independientemente de los errores en otras regiones de la imagen. La invención se define mediante las reivindicaciones independientes.

Las características y ventajas adicionales de la invención se evidenciarán a partir de la siguiente descripción detallada de realizaciones que sigue con referencia a los dibujos adjuntos.

Breve descripción de los dibujos

La Figura 1 es un diagrama de bloques de un codificador de vídeo que emplea la codificación de capa de sectores descrita en el presente documento.
La Figura 2 es un diagrama de bloques de un decodificador de vídeo que emplea la codificación de capa de sectores descrita en el presente documento.
La Figura 3 es un diagrama que ilustra una disposición jerárquica de elementos de una secuencia de vídeo representada en un flujo de bits comprimido utilizado por el codificador/decodificador de vídeo que emplea la codificación de capa de sectores.
La Figura 4 es un diagrama de sintaxis de nivel de secuencia de una sintaxis de codificación del flujo de bits comprimido utilizado por el codificador/decodificador de vídeo que emplea la codificación de capa de sectores.
La Figura 5 es un diagrama de sintaxis a nivel de fotograma de una sintaxis de codificación del flujo de bits comprimido utilizado por el codificador/decodificador de vídeo que emplea la codificación de capa de sectores.
La Figura 6 es un diagrama de sintaxis de nivel de capa de sectores de una sintaxis de codificación del flujo de bits comprimido utilizado por el codificador/decodificador de vídeo que emplea la codificación de capa de sectores.
La Figura 7 es un diagrama que representa un ejemplo de suavizado de superposición realizado en los límites de bloque.
La Figura 8 es un diagrama que muestra un ejemplo de píxeles de límite de bloque horizontal en una imagen I en la que se realiza el filtrado de desbloqueo en bucle.
La Figura 9 es un diagrama que muestra un ejemplo de píxeles de límite de bloque vertical en una imagen I en la que se realiza el filtrado de desbloqueo en bucle.
La Figura 10 es un diagrama de bloques de un entorno informático adecuado para el codificador/decodificador de vídeo de las Figuras 1 y 2.
La Figura 11 es un diagrama que representa un conjunto de segmentos de cuatro píxeles utilizados en el filtrado de bucle.
La Figura 12 es un diagrama que representa los píxeles utilizados en una operación de filtrado.
La Figura 13 es una lista de pseudocódigo de una operación de filtrado en un tercer par de píxeles de un segmento.
La Figura 14 es una lista de pseudocódigo de una operación de filtrado en un primer, segundo y cuarto pares de píxeles de un segmento.

Descripción detallada

La siguiente descripción está dirigida a implementaciones de una capa de sectores en un códec de vídeo y sintaxis de flujo de bits, que está diseñada para ser flexible y proporcionar una combinación efectiva de resistencia a errores y eficacia de compresión. Una aplicación ejemplar de la codificación de la capa de sectores es en un codificador y decodificador de imagen o vídeo. En consecuencia, la codificación de la capa de sectores se describe en el contexto de un codificador y decodificador de imagen o vídeo generalizado, pero como alternativa se puede incorporar en la sintaxis de flujo de bits de diversos otros códecs de imagen y vídeo que pueden variar en detalles de esta sintaxis de flujo de bits ejemplar que se describe a continuación.

1. Codificador y decodificador de vídeo generalizado

La Figura 1 es un diagrama de bloques de un codificador (100) de vídeo generalizado y la Figura 2 es un diagrama de bloques de un decodificador (200) de vídeo generalizado, en el que se pueden incorporar las transformaciones WMV9/VC-9.

Las relaciones que se muestran entre los módulos dentro del codificador y el decodificador indican el flujo principal de información en el codificador y el decodificador; otras relaciones no se muestran por simplicidad. En particular, las Figuras 1 y 2 por lo general no muestran información secundaria que indique la configuración, modos, tablas, etc. del codificador utilizados para una secuencia, fotograma, macrobloque, bloque, etc. de vídeo. Dicha información secundaria se envía en el flujo de bits de salida, convencionalmente después de la codificación de entropía de la

información lateral. El formato del flujo de bits de salida puede ser un formato de Windows Media Video u otro formato.

El codificador (100) y el decodificador (200) se basan en bloques y usan un formato de macrobloque 4:2:0 con cada macrobloque incluyendo 4 bloques de luminancia 8x8 de luminancia (a veces tratados como un macrobloque de 16x16) y dos bloques de crominancia de 8x8. Como alternativa, el codificador (100) y el decodificador (200) se basan en objetos, usan un macrobloque o formato de bloque diferente, o realizan operaciones en conjuntos de píxeles de tamaño o configuración diferente de los bloques de 8x8 y los macrobloques de 16x16.

Dependiendo de la implementación y del tipo de compresión deseada, los módulos del codificador o decodificador pueden agregarse, omitirse, dividirse en múltiples módulos, combinarse con otros módulos y/o reemplazarse con módulos similares. En realizaciones alternativas, el codificador o decodificadores con diferentes módulos y/u otras configuraciones de módulos realizan una o más de las técnicas descritas.

A. Codificador de vídeo

La Figura 1 es un diagrama de bloques de un sistema (100) codificador de vídeo general. El sistema (100) codificador recibe una secuencia de fotogramas de vídeo, incluyendo un fotograma actual (105), y produce información (195) de vídeo comprimida como salida. Las realizaciones particulares de los codificadores de vídeo usan convencionalmente una versión variada o complementada del codificador (100) generalizado.

El sistema (100) codificador comprime los fotogramas predichos y los fotogramas clave. En aras de la presentación, la Figura 1 muestra una trayectoria para los fotogramas clave a través del sistema (100) codificador y una trayectoria para fotogramas predichos. Muchos de los componentes del sistema (100) codificador se utilizan para comprimir los fotogramas clave y los fotogramas predichos. Las operaciones exactas realizadas por esos componentes pueden variar de acuerdo con el tipo de información que se esté comprimiendo.

Un fotograma predicho [también denominado fotograma p, fotograma b para predicción bidireccional o fotograma intercodificado] se representa en términos de predicción (o diferencia) de uno o más de otros fotogramas. Un resto de predicción es la diferencia entre lo que se predijo y el fotograma original. Por el contrario, un fotograma clave [también denominado fotograma i, fotograma intracodificado] se comprime sin referencia a otros fotogramas.

Si el fotograma (105) actual es un fotograma predicho hacia delante, un estimador (110) de movimiento estima el movimiento de macrobloques u otros conjuntos de píxeles del fotograma (105) actual con respecto a un fotograma de referencia, que es el fotograma (125) anterior reconstruido almacenado en el almacenamiento (120) de fotogramas. En realizaciones alternativas, el fotograma de referencia es un fotograma posterior o el fotograma actual se predice bidireccionalmente. El estimador (110) de movimiento sale como información (115) de movimiento de información lateral tal como vectores de movimiento. Un compensador (130) de movimiento aplica la información (115) de movimiento al fotograma (125) anterior reconstruido para formar un fotograma (135) actual compensado en movimiento. Sin embargo, la predicción rara vez es perfecta, y la diferencia entre el fotograma (135) actual compensado en movimiento y el fotograma (105) actual original es el resto (145) de predicción. Como alternativa, un estimador de movimiento y un compensador de movimiento aplican otro tipo de estimación/compensación de movimiento.

Un transformador (160) de frecuencia convierte la información de vídeo del dominio espacial en datos del dominio de frecuencia (es decir, espectrales). Para loa fotogramas de vídeo basados en bloques, el transformador (160) de frecuencia aplica una transformación descrita en las siguientes secciones que tiene propiedades similares a la transformada discreta del coseno ["DCT"]. En algunas realizaciones, el transformador (160) de frecuencia aplica una transformación de frecuencia a los bloques de restos de predicción espacial para los fotogramas clave. El transformador (160) de frecuencia puede aplicar transformaciones de frecuencia de 8x8, 8x4, 4x8 u otro tamaño.

Un cuantificador (170) cuantifica después los bloques de coeficientes de datos espectrales. El cuantificador aplica una cuantificación escalar uniforme a los datos espectrales con un tamaño de paso que varía fotograma por fotograma u otra base. Como alternativa, el cuantificador aplica otro tipo de cuantización a los coeficientes de datos espectrales, por ejemplo, una cuantificación no uniforme, vectorial o no adaptativa, o cuantifica directamente datos de dominio espacial en un sistema codificador que no utiliza transformaciones de frecuencia. Además de la cuantificación adaptativa, el codificador (100) puede usar caída, filtrado adaptativo de fotogramas u otras técnicas para el control de la velocidad.

Cuando se necesita un fotograma actual reconstruido para la estimación/compensación de movimiento posterior, un cuantificador (176) inverso realiza una cuantización inversa en los coeficientes de datos espectrales cuantificados. Un transformador (166) de frecuencia inversa realiza la inversa de las operaciones del transformador (160) de frecuencia, produciendo un resto de predicción reconstruido (para un fotograma predicho) o un fotograma clave reconstruido. Si el fotograma (105) actual era un fotograma clave, el fotograma clave reconstruido se toma como el fotograma actual reconstruido (no mostrado). Si el fotograma (105) actual era un fotograma predicho, el resto de predicción reconstruido se agrega al fotograma (135) actual compensado en movimiento para formar el fotograma actual reconstruido. El almacenamiento (120) de fotogramas almacena el fotograma actual reconstruido para su uso en la predicción del siguiente fotograma. En algunas realizaciones, el codificador aplica un filtro de desbloqueo al fotograma reconstruido para suavizar las discontinuidades de forma adaptativa en los bloques del fotograma.

El codificador (180) de entropía comprime la salida del cuantificador (170) así como cierta información lateral (por ejemplo, información (115) de movimiento, tamaño del paso de cuantificación). Las técnicas de codificación de entropía convencionales incluyen codificación aritmética, codificación diferencial, codificación Huffman, codificación de longitud de ejecución, codificación LZ, codificación de diccionario y combinaciones de las anteriores. El codificador (180) de entropía usa convencionalmente diferentes técnicas de codificación para diferentes tipos de información (por ejemplo, coeficientes DC, coeficientes AC, diferentes tipos de información secundaria), y puede elegir entre múltiples tablas de códigos dentro de una técnica de codificación particular.

El codificador (180) de entropía coloca información (195) de vídeo comprimida en el búfer (190). Un indicador de nivel del búfer se retroalimenta a los módulos adaptativos de velocidad de bits. La información (195) de vídeo comprimida se agota del búfer (190) a una velocidad de bits constante o relativamente constante y se almacena para su transmisión posterior a esa velocidad de bits. Como alternativa, el sistema (100) codificador transmite información de vídeo comprimida inmediatamente después de la compresión.

Antes o después del búfer (190), la información (195) de vídeo comprimida se puede codificar por canal para su transmisión a través de la red. La codificación del canal puede aplicar datos de detección y corrección de errores a la información (195) de vídeo comprimida.

B. Decodificador de vídeo

La Figura 2 es un diagrama de bloques de un sistema (200) decodificador de vídeo general. El sistema (200) decodificador recibe información (295) para una secuencia comprimida de fotogramas de vídeo y produce una salida que incluye un fotograma

(205) reconstruido. Las realizaciones particulares de decodificadores de vídeo usan convencionalmente una versión variada o complementada del decodificador (200) generalizado.

El sistema (200) decodificador descomprime fotogramas predichos y fotogramas clave. En aras de la presentación, la Figura 2 muestra una trayectoria para los fotogramas clave a través del sistema (200) decodificador y una trayectoria para los fotogramas predichos. Muchos de los componentes del sistema (200) decodificador se utilizan para comprimir fotogramas clave y fotogramas predichos. Las operaciones exactas realizadas por esos componentes pueden variar de acuerdo con el tipo de información que se esté comprimiendo.

Un búfer (290) recibe la información (295) para la secuencia de vídeo comprimida y hace que la información recibida esté disponible para el decodificador (280) de entropía. El búfer (290) recibe por lo general la información a una velocidad que es bastante constante en el tiempo e incluye un búfer de fluctuación para suavizar las variaciones a corto plazo en el ancho de banda o la transmisión. El búfer (290) puede incluir un búfer de reproducción y otros búferes también. Como alternativa, el búfer (290) recibe información a una velocidad variable. Antes o después del búfer (290), la información de vídeo comprimida puede decodificarse y procesarse en el canal para la detección y corrección de errores.

El decodificador (280) de entropía decodifica entrópicamente datos cuantificados codificados por entropía así como la información lateral codificada por entropía (por ejemplo, información de movimiento, tamaño de paso de cuantificación), aplicando convencionalmente la inversa de la codificación de entropía realizada en el codificador. Las técnicas de decodificación de entropía incluyen decodificación aritmética, decodificación diferencial, decodificación Huffman, decodificación de longitud de ejecución, decodificación LZ, decodificación de diccionario y combinaciones de lo anterior. El decodificador (280) de entropía utiliza con frecuencia diferentes técnicas de decodificación para diferentes tipos de información (por ejemplo, coeficientes DC, coeficientes AC, diferentes tipos de información secundaria), y puede elegir entre múltiples tablas de códigos dentro de una técnica de decodificación particular.

Si el fotograma (205) a reconstruir es un fotograma predicho hacia delante, un compensador (230) de movimiento aplica información (215) de movimiento a un fotograma (225) de referencia para formar una predicción (235) del fotograma (205) que se está reconstruyendo. Por ejemplo, el compensador (230) de movimiento usa un vector de movimiento de macrobloque para encontrar un macrobloque en el fotograma (225) de referencia. Un búfer (220) de fotogramas almacena los fotogramas reconstruidos anteriores para su uso como fotogramas de referencia. Como alternativa, un compensador de movimiento aplica otro tipo de compensación de movimiento. La predicción por el compensador de movimiento rara vez es perfecta, por lo que el decodificador (200) también reconstruye los restos de predicción.

Cuando el decodificador necesita un fotograma reconstruido para la compensación de movimiento posterior, el almacenamiento (220) de fotogramas almacena el fotograma reconstruido para su uso en la predicción del siguiente fotograma. En algunas realizaciones, el codificador aplica un filtro de desbloqueo al fotograma reconstruido para suavizar las discontinuidades de forma adaptativa en los bloques del fotograma.

Un cuantificador (270) inverso cuantifica inversamente los datos decodificados por entropía. En general, el cuantificador inverso aplica cuantificación inversa uniforme y escalar a los datos decodificados por entropía con un tamaño de paso que varía fotograma por fotograma u otra base. Como alternativa, el cuantificador inverso aplica otro tipo de cuantización inversa a los datos, por ejemplo, una cuantificación no uniforme, vectorial o no adaptativa, o

directamente cuantifica inversamente datos de dominio espacial en un sistema decodificador que no utiliza transformaciones de frecuencia inversa.

Un transformador (260) de frecuencia inversa convierte los datos cuantificados de dominio de frecuencia en información de vídeo de dominio espacial. Para las tramas de vídeo basadas en bloques, el transformador (260) de frecuencia inversa aplica una transformación inversa descrita en las siguientes secciones. En algunas realizaciones, el transformador (260) de frecuencia inversa aplica una transformación de frecuencia inversa en los bloques de restos de predicción espacial para fotogramas clave. El transformador (260) de frecuencia inversa puede aplicar transformaciones de frecuencia inversa de 8x8, 8x4, 4x8 u otro tamaño.

2. Codificación de la capa de sectores

Como se ha descrito en la sección de antecedentes anterior, una técnica para prevenir o minimizar las fallas de decodificación debido a la pérdida de paquetes y los errores de transmisión es proporcionar redundancia adicional a través de la codificación en unidades de imagen parcial decodificables de forma independiente, como los sectores proporcionados en algunos estándares de códec de vídeo anteriores. En general, un sector es una porción de una imagen que incluye una o más filas contiguas de macrobloques.

Uno de los principales desafíos de los sectores es permitir que el códec logre el equilibrio correcto entre la resistencia al error y la compresión. La razón es que algunas aplicaciones de códec de vídeo o escenarios de uso tienen que superar una cantidad significativa de pérdida de paquetes y, por lo tanto, otorgan una gran importancia a la capacidad de recuperación de errores. Otras aplicaciones requieren una resistencia mínima a los errores, pero requieren una compresión eficaz. En implementaciones de la sintaxis de flujo de bits de un códec de vídeo descrito en el presente documento, la sintaxis incorpora una capa de sectores u otra capa de unidad de imagen parcial diseñada para que se pueda lograr la elección óptima de resistencia y eficacia. Esta capacidad se logra de las siguientes maneras:

A) Reconstrucción perfecta de un Intra-Sector: La capa de sectores de la sintaxis ilustrada se ha diseñado de tal manera que las operaciones como el filtrado de bucles y la superposición no funcionan a través de los sectores. Por lo tanto, si todos los macrobloques de un sector están intracodificados, y si se conoce el encabezado de la imagen correspondiente a ese sector, ese sector puede reconstruirse exactamente sin errores, independientemente de los errores en otros sectores (regiones) de la imagen. Esto permite la reconstrucción perfecta (sin errores) de un intra-sector y proporciona una capacidad de resistencia al error significativa.
B) Procedimiento de bajo costo para repetir el encabezado del fotograma: La repetición de la información del encabezado de la imagen aumenta la probabilidad

de que el encabezado de la imagen se reciba en el decodificador, pero tiene el coste de disminuir la eficacia de compresión. La sintaxis ilustrada de la capa de sectores indica si el encabezado de la imagen se transmite en un sector usando un indicador de 1 bit. Esto permite que el codificador elija tanto el número de sectores como el sector particular en el que se retransmite el encabezado de la imagen.

A. Jerarquía de sintaxis

Más específicamente, la sintaxis de códec de vídeo ilustrada representa el vídeo que utiliza una estructura de sintaxis jerárquica que descompone cada fotograma de la secuencia de vídeo en tres capas jerárquicas básicas: imagen 310, macrobloque 340 y bloque 350, como se muestra en la Figura 3. La imagen 310 incluye un canal 330 de luminancia (Y) y los canales 331-332 de crominancia (Cr y Cb). La capa 310 de imagen está formada por filas de macrobloques 340. Cada macrobloque contiene por lo general seis bloques: un grupo de bloques 2x2 de la capa de luminancia y un bloque de cada uno de los canales de crominancia. Los bloques consisten por lo general en muestras de luminancia o crominancia de 8x8 (aunque bloques de transformación de 4x8, 8x4 y 4x4 se pueden usar también en la sintaxis de códec de vídeo ilustrada), a los que se aplica una transformación para la codificación basada en la transformación.

Además, una cuarta capa opcional, denominada capa 320 de sectores, puede estar presente entre la capa 310 de imagen y la capa 340 de macrobloque. Una división se define para contener una o más filas contiguas de macrobloques que se escanean en orden de exploración de ráster. Por tanto, una imagen 310 puede descomponerse en sectores 320, que a su vez, pueden descomponerse en macrobloques 340. En esta sintaxis de códec de vídeo ilustrada, un sector siempre comienza en el primer macrobloque de una fila y termina en el último macrobloque de la misma u otra fila. Por lo tanto, un sector contiene un número entero de filas completas. Además, las imágenes y los sectores siempre están alineados en bytes en esta sintaxis de flujo de bits de códec de vídeo ilustrado, y se transmiten en una unidad decodificable (IDU) independiente como se describe a continuación. Se detecta una nueva imagen, o una porción, a través de códigos de inicio como se describe a continuación.

B. Definición de la capa de sectores

Una división representa una o más filas contiguas de macrobloques que se escanean en orden de exploración de ráster. La capa de sectores en la sintaxis ilustrada es opcional y se puede omitir codificando una imagen como una sola unidad decodificable (IDU) independiente. Cuando una imagen se codifica en múltiples IDU, se utilizan sectores. Tenga en cuenta que un sector siempre comienza en el primer macrobloque de una fila y termina en el último macrobloque de la misma fila u otra. Por lo tanto, un sector contiene un número entero de filas completas. Un sector

siempre está alineado en bytes, y cada sector se transmite en una IDU diferente. El comienzo de un nuevo sector se detecta mediante la búsqueda de códigos de inicio como se describe a continuación.

Cuando comienza un nuevo sector, los predictores de vector de movimiento, los predictores para los coeficientes AC y DC, y los predictores para los parámetros de cuantificación se reinician. En otras palabras, con respecto a la predicción, la primera fila de macrobloques en el sector se considera la primera fila de macrobloques en la imagen. Esto asegura que no haya dependencia entre sectores en los predictores. Además, cuando se utilizan sectores, toda la información del plano de bits se transporta en modo sin procesar, lo que garantiza que cada macrobloque lleve su propia información local.

C. Estructura de sintaxis de la capa de sectores

Con referencia a las Figuras 4 y 5, el flujo 195 de bits de vídeo comprimido (Figura 1) incluye información para una secuencia de fotogramas de vídeo progresivos comprimidos u otras imágenes (por ejemplo, fotogramas entrelazados o imágenes en formato de campo entrelazado). El flujo de bits está organizado en diversas capas jerárquicas que son decodificadas por un decodificador como el decodificador (200) de la Figura 2. La capa más alta es la capa de secuencia, que tiene información para la secuencia general de fotogramas. Además (como se ha resumido anteriormente), cada fotograma de vídeo comprimido está compuesto de datos estructurados en tres capas jerárquicas: imagen, macrobloque y bloque (de arriba a abajo); y opcionalmente una capa de sectores entre la imagen y las capas de macrobloque.

La Figura 4 es un diagrama de sintaxis para la capa 400 de secuencia, que incluye un encabezado 410 de secuencia seguido de datos para la capa 500 de imagen (véase Figura 5). El encabezado 410 de secuencia incluye diversos elementos de nivel de secuencia que son procesados por el decodificador y utilizados para decodificar la secuencia.

La Figura 5 es un diagrama de sintaxis para la capa 500 de imagen para un fotograma intracodificada entrelazado ["fotograma I entrelazado"]. Los diagramas de sintaxis para otras imágenes, como fotogramas I progresivos, fotogramas P y fotogramas B, tienen muchos elementos de sintaxis similares. La capa 500 de imagen incluye un encabezado 510 de imagen seguido de datos para la capa 520 de macrobloque. El encabezado 510 de imagen incluye varios elementos de nivel de imagen que son procesados por el decodificador y utilizados para decodificar el fotograma correspondiente. Algunos de esos elementos solo están presentes si su presencia está señalada o implicada por un elemento de nivel de secuencia o un elemento de nivel de imagen anterior.

La Figura 6 es un diagrama de sintaxis para la capa 600 de sectores, que incluye un encabezado 610 del sector seguido de datos para la capa 520 de macrobloque. Los elementos que componen el encabezado 610 del sector incluyen un elemento 620 de dirección del sector (SLICE_ADDR) y un elemento 630 de indicador de presencia de encabezado de imagen (PIC_HEADER_FLAG), como también se muestra en la siguiente tabla 1.

El elemento 620 de dirección del sector es un elemento de sintaxis de 9 bits de longitud fija. La dirección de fila de la primera fila de macrobloques en el sector está codificada en binario en este elemento de sintaxis. En la implementación ilustrada, el intervalo de este elemento de sintaxis es de 1 a 511, en el que el tamaño máximo de la imagen de 8192 corresponde a un máximo de 512 filas de macrobloques.

PIC_HEADER_FLAG 630 es un elemento de sintaxis de 1 bit que está presente en el encabezado del sector. Si PIC_HEADER_FLAG = 0, la información del encabezado de la imagen no se repite en el encabezado del sector. Si PIC_HEADER_FLAG = 1, la información del encabezado 510 de imagen (Figura 5) que aparece en la capa de imagen que contiene este sector se repite en el encabezado del sector.

Tabla 1: Flujo de bits de la capa de sectores

SECTOR() {	Número de bits	
SLICE_ADDR	9	
PIC_HEADER_FLAG	1	
Si (PIC_HEADER_FLAG == 1) {		
PICTURE_LAYER ()		
}		
para ('todos los macrobloques') {		
MB_LAYER ()		

(continuación)

SECTOR() {	Número de bits	
}		
}		

3. Códigos de inicio unitarios independientemente decodificables

En la sintaxis de flujo de bits del codificador/decodificador de vídeo ilustrado, una unidad independientemente decodificable (IDU) de datos de vídeo comprimido comienza con un identificador denominado código de inicio (SC).

Una IDU podría referirse a una sola imagen, o un sector (es decir, un grupo de macrobloques en una imagen), o un grupo de imágenes (GOP), o un encabezado de secuencia.

El código de inicio es una secuencia de cuatro bytes, que consiste en un prefijo de código de inicio (SCP) único de tres bytes y un sufijo de código de inicio (SCS) de un byte. El SCP es la secuencia única de tres bytes (0x000001). El SCS se utiliza para identificar el tipo de IDU que sigue al código de inicio. Por ejemplo, el sufijo del código de inicio antes de una imagen es diferente del sufijo del código de inicio antes de un sector. Los códigos de inicio siempre están alineados con bytes.

Se describe un mecanismo de encapsulación (EM) para evitar la emulación del prefijo del código de inicio en el flujo de bits. Los datos comprimidos antes de la encapsulación se denominan unidad de decodificación independiente sin procesar (RIDU), mientras que la IDU encapsulada (EIDU) se refiere a los datos después de la encapsulación.

La siguiente sección proporciona una perspectiva del lado del codificador sobre cómo funciona el código de inicio y la encapsulación. La sección E.2 especifica la detección de códigos de inicio y EIDU en el decodificador. La sección E.3 trata de la extracción de una RIDU de una EIDU. La sección E.4 especifica sufijos de código de inicio para diversos tipos de IDU.

A. Códigos de inicio y encapsulación: un punto de vista del codificador

La encapsulación de una RIDU para obtener una EIDU se describe a continuación.

Etapa 1: Se agrega un bit '1' de salida al final de la RIDU. El EM ahora agrega entre 0 y 7 bits al final de la IDU de tal manera que la IDU termine en una ubicación alineada con bytes. El valor de estos bits de "relleno" es '0'. Como resultado, al final de esta etapa, la IDU se representa en un número entero de bytes, en el que el último byte de la IDU no puede ser un byte de valor cero. La cadena de bytes resultante se denomina bytes de carga útil de la IDU.

Etapa 2: El prefijo de código de inicio de tres bytes (0x000001) y el sufijo de código de inicio apropiado que identifica el tipo de IDU se colocan al comienzo de la EIDU.

Etapa 3: El resto de la EIDU se forma procesando los bytes de carga útil de la IDU a través del siguiente procedimiento de prevención de emulación. La emulación de los prefijos del código de inicio en IDU se elimina mediante el relleno de bytes. El procedimiento de prevención de emulación es equivalente a la siguiente operación:

1) Reemplazar cada cadena dentro de la carga útil de 2 bytes consecutivos de valor 0x00 seguido de un byte que contiene valores cero en sus seis MSB (independientemente de los valores LSB) con 2 bytes de valor 0x00 seguido de un byte igual a 0x03 seguido de un byte igual al último byte de la cadena de datos original de tres bytes. Este procedimiento se ilustra en la Tabla 2.

Tabla 2: Reemplazo del patrón de prevención de emulación

Patrón a reemplazar	Patrón de reemplazo
0x00, 0x00, 0x00	0x00, 0x00, 0x03, 0x00
0x00, 0x00, 0x01	0x00, 0x00, 0x03, 0x01
0x00, 0x00, 0x02	0x00, 0x00, 0x03, 0x02
0x00, 0x00, 0x03	0x00, 0x00, 0x03, 0x03

Etapa 3: El prefijo de código de inicio de tres bytes (0x000001) y el sufijo de código de inicio apropiado que identifica el tipo de IDU se adjuntan al comienzo de la IDU. La carga útil resultante es una IDU encapsulada.

El codificador también puede insertar cualquier número de bytes de relleno de valor cero después del final de una EIDU. De manera equivalente, se puede insertar cualquier cantidad de bytes de relleno de valor cero antes de un prefijo de código de inicio. El código de inicio está estructurado de tal manera que puede ser detectado por un decodificador incluso en presencia de estos bytes de relleno de valor cero. En algunos entornos de transmisión como H.320, el codificador puede usar esta función para insertar bytes de relleno de bytes adicionales de valor cero según lo deseado, lo que puede permitir que el decodificador recupere rápidamente la ubicación de los códigos de inicio, incluso si ha perdido la noción de la alineación prevista del flujo de bits a los límites de bytes. Además, estos bytes de relleno de valor cero también pueden ser útiles para empalmar flujos de bits, llenar un canal de velocidad de bits constante, etc. Los bytes de relleno de valor cero antes de los códigos de inicio, o al final de una EIDU, no se procesan a través de la encapsulación mecanismo: solo los datos de RIDU requieren dicho procesamiento.

B. Detección de códigos de inicio y EIDU

La detección de un EIDU comienza con la búsqueda del prefijo del código de inicio.

Detección de códigos de inicio a partir de posiciones alineadas con bytes. En un decodificador que no puede perder la alineación de bytes, o una vez que se ha establecido la alineación de bytes, la detección del código de inicio se realiza de la siguiente manera.

1. Cada vez que se encuentra una cadena de dos o más bytes de valor 0x00 seguido de un byte de valor 0x01, se declara una detección de prefijo de código de inicio.

Cuando se detectan 2 prefijos de códigos de inicio sucesivos, el flujo de bits de carga útil entre ellos se declara como una nueva EIDU.

Detección de códigos de inicio después de la pérdida de alineación de bytes en un decodificador. En un decodificador que ha perdido la alineación de bytes (como puede suceder en algunos entornos de transmisión), la detección de prefijo de código de inicio y la detección de alineación de bytes se realizan de la siguiente manera. Cada vez que se encuentra una cadena de tres o más bytes de valor 0x00, seguido de cualquier byte distinto de cero, se declara una detección de prefijo de código de inicio y se entiende que la alineación de bytes se recupera de modo que el primer bit distinto de cero en el distinto de cero byte es el último bit de un código de inicio alineado con bytes.

C. Extracción de RIDU de EIDU

La extracción de una IDU sin procesar de una IDU encapsulada se describe a continuación.

Etapa 1: El sufijo de código de inicio se utiliza para identificar el tipo de IDU.

Etapa 2: El primer paso es eliminar los bytes de relleno de valor cero al final de EIDU. Después de este paso, el último byte de la IDU debe tener un valor distinto de cero.

Etapa 3: Los bytes utilizados para la prevención de la emulación se detectan y eliminan. El procedimiento es el siguiente: Siempre que una cadena de dos bytes de valor 0x00 sea seguida por un byte igual a 0x03, el byte igual a 0x03 se entiende como un byte de prevención de emulación y se descarta.

Este procedimiento se ilustra en la Tabla 3.

Tabla 3: Eliminación de datos de prevención de emulación del decodificador

Patrón a reemplazar	Patrón de reemplazo
0x00, 0x00, 0x03, 0x00	0x00, 0x00, 0x00
0x00, 0x00, 0x03, 0x01	0x00, 0x00, 0x01
0x00, 0x00, 0x03, 0x02	0x00, 0x00, 0x02
0x00, 0x00, 0x03, 0x03	0x00, 0x00, 0x03

Los siguientes patrones de bytes, si se ven dentro del flujo de bits, representan condiciones de error (observando que la pérdida de la alineación de bytes adecuada por parte del decodificador se considera una condición de error):

a) Una cadena de dos bytes de valor 0x00 seguido de un byte igual a 0x02 indica condición de error.
b) Una cadena de tres o más bytes de valor 0x00, si no es seguida por un byte de 0x01 es una condición de error (tenga en cuenta que si dos o más bytes iguales a cero son seguidos por un byte de valor 0x01 y la alineación de bytes no ha sido perdido, se declara la detección de un código de inicio posterior).
c) Una cadena de dos bytes de valor 0x00, seguida de un byte de valor 0x03, seguido de un byte que no es uno de 0x00, 0x01 o 0x02, o 0x03.

Etapa 4: En el último byte de la IDU, se identifica el último bit distinto de cero, y ese bit distinto de cero, y todos los bits "cero" que siguen se descartan. El resultado es un IDU sin procesar.

D. Sufijos de código de inicio para tipos de IDU

Los sufijos del código de inicio para varios tipos de IDU se presentan en la Tabla 4.

Tabla 4: Inicio de sufijos de código para diversos tipos de IDU

Sufijo de código de inicio	Tipo de IDU
0x00	SMPTE Reservado
0x01-0x09	SMPTE Reservado

0x0A	Fin de secuencia
0x0B	Sector
0x0C	Campo
0x0D	Fotograma
0x0E	Encabezado de punto de entrada
0x0F	Encabezado de secuencia
0x10-0x1A	SMPTE Reservado
0x1B	Datos de usuario del nivel de sector
0x1C	Datos de usuario del nivel de campo
0x1D	Datos de usuario a nivel de fotograma
0x1E	Datos de usuario del nivel de punto de entrada
0x1F	Datos de usuario del nivel de secuencia
0x20-0x7F	SMPTE Reservado
0x80-0xFF	Prohibido

El sufijo SequenceHeader se envía para identificar las IDU que llevan un encabezado 410 de secuencia (Figura 4).

El sufijo de Encabezado De Punto De Entrada se envía para identificar las IDU que llevan un encabezado de punto de entrada.

El sufijo de Imagen se envía para identificar las IDU que contienen la imagen 320 (Figura 3 y el encabezado 510 de imagen (Figura 5).

El sufijo de Campo se envía para identificar las IDU que contienen el segundo campo de una imagen que se codifica como dos campos separados.

El sufijo de Segmento se envía para identificar las IDU que llevan sectores 320 (Figura 3) y el encabezado de sector 610 (Figura 6).

Los sufijos de datos del nivel de Secuencia, Punto de Entrada, Fotograma, Campo y Sector se utilizan para transmitir cualquier dato definido por el usuario asociado con la secuencia, el punto de entrada, el fotograma, el campo y sector, respectivamente.

El "Fin de secuencia" es un sufijo opcional que indica que la secuencia actual ha finalizado, y no se transmitirán más datos para esta secuencia. Tenga en cuenta que la transmisión de un "fin de secuencia" puede estar presente, pero el final de una secuencia se deducirá del encabezado de la siguiente secuencia.

4. Independencia de la capa de sectores

La capa 320 de sectores ilustrada (Figura 3) también logra la decodificación independiente y la reconstrucción independiente. Esto permite que el sector se reconstruya sin errores en el decodificador, independientemente de los errores de transmisión o la pérdida de paquetes en otras regiones de la imagen 310 (Figura 3).

A. Decodificabilidad independiente

El contenido de una capa 320 de sectores se decodifica independientemente del contenido de la imagen en otros sectores o regiones de la imagen. Cuando comienza un nuevo sector, el codificador 100 y el decodificador 200 restablecen los predictores del vector de movimiento, los predictores para los coeficientes de AC y DC, y los predictores para los parámetros de cuantificación. En otras palabras, con respecto a la predicción, la primera fila de macrobloques en el sector se trata como si fuera la primera fila de macrobloques en la imagen. Esto ayuda a garantizar que no haya dependencia entre sectores en los predictores. Además, cuando se usan sectores, la información de nivel de macrobloque que de otro modo se codifica (por ejemplo, usando la codificación de plano de bits) en la capa de imagen (como el modo de vector de movimiento e indicadores para la predicción de ac) se transporta localmente con otra información de nivel de macrobloque como los coeficientes de transformación. Esto permite que cada sector se decodifique de forma independiente (es decir, sin depender de los datos decodificados de otros sectores de la imagen).

B. Reconstrucción independiente:

Además, el procedimiento de reconstrucción de un sector se realiza independientemente de la reconstrucción de cualquier otro sector (por ejemplo, sectores adyacentes) en una imagen. En consecuencia, cualquier procedimiento (como el desbloqueo en el bucle o el filtrado de superposición, que se describe a continuación) que de otro modo se aplicaría a través de los límites entre los sectores adyacentes en una imagen no está permitido. En otras palabras, las filas de macrobloques superior e inferior de cada sector se tratan como si fueran las filas de macrobloques superior e inferior de la imagen en dichos procedimientos de límite.

Suavizado de superposición

Las transformaciones superpuestas son transformaciones basadas en bloques modificados que intercambian información a través del límite del bloque. Con una transformación superpuesta bien diseñada, se pueden minimizar los artefactos de bloqueo. Para intra bloques, el códec de vídeo ilustrado simula una transformación superpuesta al acoplar una transformación de bloque de 8x8 con una operación de filtrado (denominada suavizado de superposición). Los bordes de un bloque de 8x8 que separan dos intra bloques se suavizan; de hecho, se implementa una transformación superpuesta en esta interfaz. Excepto, el suavizado de superposición no se realiza a través de los límites de sector en ningún caso.

Si el elemento de sintaxis de capa de secuencia OVERLAP 420 (Figura 4) se establece en 1, entonces se puede realizar una operación de filtrado condicionalmente a través de los bordes de dos intra bloques vecinos, tanto para los canales de luminancia como de crominancia. Esta operación de filtrado (denominada *suavizado de superposición)* se realiza después de decodificar el fotograma y antes del desbloqueo en bucle. Sin embargo, el suavizado de superposición se puede realizar después de decodificar los sectores de macrobloques relevantes, puesto que esto es funcionalmente equivalente al suavizado después de decodificar todo el fotograma.

La Figura 7 muestra un ejemplo de suavizado de superposición realizado en una parte de un fotograma P con bloques I. Este podría ser el canal de luminancia o crominancia. Los bloques I son grises (o sombreados) y los bloques P son blancos. En esta ilustración, la interfaz de borde sobre la que se aplica el suavizado de superposición se marca con un patrón de sombreado. El suavizado de superposición se aplica a dos píxeles a cada lado del límite de separación. El área inferior derecha del fotograma se muestra aquí como un ejemplo. Los píxeles ocupan celdas individuales y los bloques están separados por líneas gruesas. El círculo oscuro marca el sub-bloque de esquina de 2x2 píxeles que se filtra en ambas direcciones.

El recuadro inferior de la Figura 7 muestra cuatro píxeles etiquetados, a0 y a1 están a la izquierda y b1, b0 a la derecha del borde del bloque vertical. El recuadro superior muestra los píxeles marcados como p0, p1, q1 y q0 a horcajadas en un borde horizontal. La siguiente sección describe el filtro aplicado a estas ubicaciones de cuatro píxeles.

El suavizado de superposición se lleva a cabo en la reconstrucción no sujetada de 16 bits. Esto es necesario porque el procedimiento directo asociado con el suavizado de superposición puede dar como resultado una expansión del intervalo más allá del intervalo permitido de 8 bits para valores de píxeles. El resultado del suavizado de superposición se limita a 8 bits, en línea con el resto de los píxeles no tocados por el suavizado de superposición.

Los bordes verticales (píxeles a0, a1, b1, b0 en el ejemplo anterior) se filtran primero, seguidos de los bordes horizontales (píxeles p0, p1, q1, q0). El resultado intermedio que sigue a la primera etapa de filtrado (suavizado de borde vertical) se almacena en 16 bits. Los filtros principales aplicados a los cuatro píxeles que se extienden a ambos lados se muestran a continuación:

$$\begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{pmatrix} = \left(\begin{pmatrix} 7 & 0 & 0 & 1 \\ -1 & 7 & 1 & 1 \\ 1 & 1 & 7 & -1 \\ 1 & 0 & 0 & 7 \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} r_0 \\ r_1 \\ r_0 \\ r_1 \end{pmatrix} \right) >> 3$$

Los píxeles originales que se filtran son (x0, x1, x2, x3). r0 y r1 son parámetros de redondeo, que toman valores alternos de 3 y 4 para garantizar un redondeo estadísticamente imparcial. La matriz filtra los valores originales con entradas que son claramente fáciles de implementar. Estos valores, después de agregar los factores de redondeo, se desplazan tres bits en bits para obtener la salida filtrada (y0, y1, y2, y3).

Para los filtros de borde horizontal y vertical, los valores de redondeo son r0 = 4, r1 = 3 para columnas y filas indexadas de manera impar, respectivamente, suponiendo que la numeración dentro de un bloque comience en 1. Para columnas/filas con índice par, r0 = 3 y r1 = 4. El filtrado se define como una operación *in situ* de 16 bits, por lo que los píxeles originales se sobrescriben después del suavizado. Para el filtrado de bordes verticales, los píxeles (a0, a1, b1, b0) corresponden a (x0, x1, x2, x3), que a su vez se filtran a (y0, y1, y2, y3). Del mismo modo, para el filtrado de borde horizontal, la correspondencia es con (p0, p1, q1, q0) respectivamente.

Los píxeles en la esquina de 2x2, mostrados por el círculo oscuro en la Figura 7, se filtran en ambas direcciones. El orden de filtrado determina sus valores finales y, por lo tanto, es importante mantener el orden (filtrado de borde vertical seguido de filtrado de borde horizontal) para la exactitud de los bits. Conceptualmente, la sujeción se debe realizar después de las dos etapas de filtrado direccional, en todos los píxeles que se filtran. Sin embargo, puede haber alguna ventaja computacional para combinar la sujeción con el filtrado.

Filtro de desbloqueo en bucle

El filtrado en bucle es un procedimiento realizado por el codificador/decodificador de vídeo en los límites del bloque para suavizar las discontinuidades. Si el elemento 430 de sintaxis de capa de secuencia LOOPFILTER (Figura 4) se establece en 1, se realiza una operación de filtrado en cada fotograma reconstruida. Esta operación de filtrado se realiza antes de usar el fotograma reconstruido como referencia para la codificación predictiva de movimiento.

Puesto que la intención del filtrado en bucle es suavizar las discontinuidades en los límites del bloque, el procedimiento de filtrado funciona en los píxeles que bordean los bloques vecinos. Para las imágenes P, los límites del bloque pueden ocurrir cada 4ª, 8ª, 12ª, etc fila o columna de píxeles dependiendo de si se usa una Transformación inversa de 8x8, 8x4 o 4x8. Para las imágenes I, el filtrado ocurre cada 8ª, 16ª, 24ª, etc. fila y columna de píxeles.

Para las imágenes I, el filtrado de desbloqueo se realiza en todos los límites de bloque de 8x8, excepto que el filtrado de desbloqueo no se realiza en los límites de sector (que se tratan de manera similar a los bordes de imagen). Las Figuras 8 y 9 muestran los píxeles que se filtran a lo largo de las regiones de borde horizontal y vertical de un fotograma de imagen I. Las Figuras muestran la esquina superior izquierda de un plano componente (luma, Cr o Cb). Las cruces representan píxeles y las cruces dentro de un círculo representan los píxeles que se filtran.

Como muestran las Figuras, la línea horizontal superior y la primera línea vertical de una imagen o sector no se filtran. Aunque no se muestra, la línea horizontal inferior y la última línea vertical de una imagen o sector tampoco se filtran. En términos más formales, se filtran las siguientes líneas:

donde N = el número de bloques horizontales de 8x8 en el plano (N * 8 = tamaño de fotograma horizontal) y M = el número de bloques verticales de 8x8 en el fotograma (M * 8 = tamaño de fotograma vertical),
Líneas (7,8), (15,16) horizontales... ((N - 1)* 8 - 1, (N - 1)* 8) se filtran y las líneas (7, 8), (15, 16) verticales... ((M-1)* 8 - 1, (M - 1)* 8) se filtran.

El orden en que se filtran los píxeles es importante. Todas las líneas de límite horizontales en el fotograma se filtran primero, seguidas de las líneas de límite verticales.

Para las imágenes P, los bloques pueden ser Intra o Intercodificados. Los bloques intracodificados siempre usan una Transformación de 8x8 para transformar las muestras y los límites del bloque 8x8 siempre se filtran. Los bloques intercodificados pueden usar una transformación inversa de 8x8, 8x4, 4x8 o 4x4 para construir las muestras que representan el error residual. Dependiendo del estado de los bloques vecinos, el límite entre los bloques actuales y vecinos puede o no filtrarse. En cualquier caso, los bordes de límite de una imagen o sector no se filtran.

Operación del filtro

Esta sección describe la operación de filtrado que se realiza en los píxeles del límite del bloque en los fotogramas I y P, como se ha descrito anteriormente.

Puesto que el número mínimo de píxeles consecutivos que se filtrarán en una fila o columna es cuatro y el número total de píxeles en una fila o columna siempre será múltiplo de cuatro, la operación de filtrado se realiza en segmentos de cuatro píxeles.

Por ejemplo, si se filtran los ocho pares de píxeles que forman el límite vertical entre dos bloques, los ocho píxeles se dividen en dos segmentos 1100 de 4 píxeles como se muestra en la Figura 12. En cada segmento de 4 píxeles, el tercer par de píxeles se filtra primero como lo indican las X. El resultado de esta operación de filtro determina si los otros tres píxeles en el segmento también se filtran, como se describe a continuación.

La Figura 10 muestra los píxeles 1200 que se utilizan en la operación de filtrado realizada en el 3º par de píxeles. Los píxeles P4 y P5 son los pares de píxeles que se pueden cambiar en la operación del filtro.

El pseudocódigo 1300 de la Figura 13 muestra la operación de filtrado realizada en el 3º par de píxeles en cada segmento. El valor filter_other_3_pixels indica si los 3 pares de píxeles restantes en el segmento se filtran también. Si filter_other_3_pixels = true, se filtran los otros tres pares de píxeles. Si filter_other_3_pixels = false, entonces no se filtran y la operación de filtrado pasa al siguiente segmento de 4 píxeles. El pseudocódigo 1400 de la Figura 14 muestra la operación de filtrado que se realiza en el 1º, 2º y 4º par de píxeles si filter_other_3_pixels = verdadero.

Esta sección ha utilizado el límite vertical para fines ejemplares. La misma operación se utiliza para filtrar los píxeles del límite horizontal.

5. Entorno informático

Las implementaciones descritas anteriormente de la codificación de la capa de sectores se pueden realizar en cualquiera de una variedad de dispositivos en los que se realiza el procesamiento de señales de imagen y vídeo, incluyendo entre otros ejemplos, ordenadores; equipos de grabación, transmisión y recepción de imágenes y vídeos; reproductores de vídeo portátiles; videoconferencia; aplicaciones de transmisión de vídeo web; y etc. Las técnicas de codificación de imagen y vídeo pueden implementarse en circuitos de hardware (por ejemplo, en circuitos de un ASIC, FPGA, etc.), así como en un software de procesamiento de imagen y vídeo que se ejecuta dentro de un ordenador u otro entorno informático (ya sea ejecutado en la unidad central de procesamiento (CPU) o procesador de gráficos dedicado, tarjeta de vídeo o similar), como se muestra en la Figura 10.

La Figura 10 ilustra un ejemplo generalizado de un entorno (1000) informático adecuado en el que se puede implementar la codificación de capa de sectores descrita. El entorno (1000) informático no pretende sugerir ninguna limitación en cuanto al alcance de uso o funcionalidad de la invención, puesto que la presente invención puede implementarse en diversos entornos informáticos de propósito general o de propósito especial.

Con referencia a la Figura 10, el entorno (1000) informático incluye al menos una unidad (1010) de procesamiento y una memoria (1020). En la Figura 10, esta configuración (1030) más básica se incluye dentro de una línea discontinua. La unidad (1010) de procesamiento ejecuta instrucciones ejecutables por ordenador y puede ser un procesador real o virtual. En un sistema de procesamiento múltiple, varias unidades de procesamiento ejecutan instrucciones ejecutables por ordenador para aumentar la potencia de procesamiento. La memoria (1020) puede ser memoria volátil (por ejemplo, registros, caché, RAM), memoria no volátil (por ejemplo, ROM, EEPROM, memoria flash, etc.) o alguna combinación de ambas. La memoria (1020) almacena el software (1080) que implementa la codificación de capa de sectores descrita.

Un entorno informático puede tener características adicionales. Por ejemplo, el entorno (1000) informático incluye almacenamiento (1040), uno o más dispositivos (1050) de entrada, uno o más dispositivos (1060) de salida y una o más conexiones de comunicación (1070). Un mecanismo de interconexión (no mostrado) como un bus, controlador o red interconecta los componentes del entorno (1000) informático. Convencionalmente, el software del sistema operativo (no mostrado) proporciona un entorno operativo para otro software que se ejecuta en el entorno (1000) informático y coordina las actividades de los componentes del entorno (1000) informático.

El almacenamiento (1040) puede ser extraíble o no extraíble e incluye discos magnéticos, cintas o casetes magnéticos, CD-ROM, CD-RW, DVD o cualquier otro medio que pueda usarse para almacenar información y al que se pueda acceder en el entorno (1000) informático. El almacenamiento (1040) almacena instrucciones para el software (1080) que implementa el codificador de audio que realiza la codificación de la capa de sectores.

El uno o más dispositivos (1050) de entrada puede ser un dispositivo de entrada táctil, como un teclado, mouse, bolígrafo o bola de seguimiento, un dispositivo de entrada de voz, un dispositivo de escaneo u otro dispositivo que proporciona información al entorno (1000) informático. Para el audio, los dispositivos (1050) de entrada pueden ser una tarjeta de sonido o un dispositivo similar que acepte la entrada de audio en forma analógica o digital, o un lector de CD-ROM que proporcione muestras de audio al entorno informático. Los dispositivos (1060) de salida pueden ser una pantalla, impresora, altavoz, grabadora de CD u otro dispositivo que proporcione salida del entorno (1000) informático.

Las conexiones (1070) de comunicación permiten la comunicación a través de un medio de comunicación a otra entidad informática. El medio de comunicación transmite información tal como instrucciones ejecutables por ordenador, información de audio o vídeo comprimido u otros datos en una señal de datos modulada. Una señal de datos modulada es una señal que tiene una o más de sus características establecidas o cambiadas de un modo tal que se codifique la información de la señal. A modo de ejemplo, y sin limitación, los medios de comunicación incluyen técnicas cableadas o inalámbricas implementadas con un portador eléctrico, óptico, RF, infrarrojo, acústico u otro.

Las técnicas de codificación/decodificación de la capa de sectores en el presente documento pueden describirse en el contexto general de los medios legibles por ordenador. Los medios legibles por ordenador son cualquier medio disponible al que se pueda acceder dentro de un entorno informático. A modo de ejemplo, y sin limitación, con el entorno (1000) informático, los medios legibles por ordenador incluyen memoria (1020), almacenamiento (1040), medios de comunicación y combinaciones de cualquiera de los anteriores.

La codificación de la capa divisoria en el presente documento puede describirse en el contexto general de las instrucciones ejecutables por ordenador, como las incluidas en los módulos de programa, que se ejecutan en un entorno informático en un procesador real o virtual objetivo. Por lo general, los módulos de programa incluyen rutinas, programas, bibliotecas, objetos, clases, componentes, estructuras de datos, etc. que realizan tareas particulares o implementan tipos de datos abstractos particulares. La funcionalidad de los módulos de programa se puede combinar o dividir entre módulos de programa como se desee en diversas realizaciones. Las instrucciones ejecutables por ordenador para los módulos de programa pueden ejecutarse dentro de un entorno informático local o distribuido.

En aras de la presentación, la descripción detallada utiliza términos como "determinar", "generar", "ajustar" y "aplicar" para describir las operaciones informáticas en un entorno informático. Estos términos son abstracciones de alto nivel para las operaciones realizadas por un ordenador, y no deben confundirse con los actos realizados por un ser humano.

Las operaciones informáticas reales que corresponden a estos términos varían de acuerdo con la implementación.

En vista de las muchas realizaciones posibles a las que se pueden aplicar los principios de nuestra invención, reivindicamos como nuestra invención todas las realizaciones que puedan estar dentro del alcance de las siguientes reivindicaciones y equivalentes a las mismas.

REIVINDICACIONES

1. Un procedimiento de decodificación de vídeo e imágenes, que comprende:
decodificar una imagen de un flujo de bits codificado que tiene una jerarquía de sintaxis que comprende al menos una capa (310, 500) de imagen, un sector (320) independientemente decodificable que contiene una o más filas contiguas de macrobloques de la imagen y una capa (340, 520) de macrobloque, en el que una sintaxis (600) de codificación del sector independientemente decodificable incluye un código de inicio conformado de un prefijo de código de inicio, SCP, que es una secuencia única de tres bytes, y un sufijo de código de inicio de un byte, SCS, utilizado para identificar un sector, estando el código de inicio seguido por un encabezado (610) del sector que señala una dirección (620) de dicho sector y una indicación (630) si la información (510) del encabezado de imagen se repite en dicho sector, estando el encabezado del sector seguido por datos de carga útil para la capa de macrobloque, en el que se evita la emulación del SCP a través de un mecanismo de encapsulación según el cual cualquier cadena dentro de la carga útil de tres bytes consecutivos similar a los tres bytes del SCP se reemplaza con una cadena similar en la que un byte de valor fijo, complementario se ha insertado, en el que la decodificación de la imagen comprende:

decodificar dicha indicación señalizada si la información de encabezado de imagen se repite en dicho sector;
si se indica que la información del encabezado de la imagen se repite, decodificar la información del encabezado de la imagen;
decodificar la dirección señalada;
reconstruir el sector en una ubicación en la imagen indicada por la dirección señalada; y
realizar el suavizado (700) de superposición en al menos algunos bordes de bloque dentro de dicho sector, excepto en los bordes de límite de dicho sector, en el que el suavizado de superposición incluye operaciones de filtrado acopladas a una transformación basada en bloque para simular una transformación superpuesta.

2. El procedimiento de la reivindicación 1, en el que la sintaxis de codificación señala dicha dirección como un índice de fila de macrobloques del sector.

3. El procedimiento de la reivindicación 1, en el que la sintaxis de codificación señala dicha indicación de si la información del encabezado de la imagen se repite en dicho sector como un indicador de un solo bit.

4. El procedimiento de la reivindicación 1, en el que la decodificación de la imagen comprende además:
realizar el filtrado de desbloqueo de al menos algunos bordes de bloque dentro de dicho sector, excepto en los bordes de límite de dicho sector.

5. El procedimiento de la reivindicación 1, en el que la decodificación de la imagen comprende además:
restablecer los parámetros de codificación de predicción al comenzar la decodificación de dicho sector.

6. Un procedimiento de codificación de una imagen o vídeo, que comprende:
codificar una imagen en al menos un sector (320) independientemente decodificable que contiene una o más filas contiguas de macrobloques de la imagen, en el que la codificación de la imagen comprende:

al comienzo de la codificación de la información del contenido de la imagen de uno respectivo de dicho al menos un sector independientemente decodificable, restablecer los parámetros de codificación de predicción;
señalizar un inicio de uno respectivo de dicho al menos un sector independientemente decodificable de codificador a decodificador utilizando un patrón de código de inicio único conformado de un prefijo de código de inicio, SCP, que es una secuencia única de tres bytes y un sufijo de código de inicio de un byte, SCS, usado para identificar un sector, estando el código de inicio seguido por un encabezado (610) del sector, estando el encabezado del sector seguido por los datos de carga útil para una capa (340, 520) de macrobloque, en el que la emulación del SCP se evita a través de un mecanismo de encapsulación según el cual cualquier cadena dentro de la carga útil de tres bytes consecutivos similar a los tres bytes del SCP se reemplaza por una cadena similar en la que se ha insertado un byte de valor fijo complementario;
señalizar en el encabezado del sector una ubicación de uno respectivo de dichos al menos un sector independientemente decodificable del codificador al decodificador usando un parámetro (620) de dirección relacionado con una posición de macrobloque inicial de dicho sector respectivo dentro de la imagen;
señalizar en el encabezado del sector una indicación (630) de si la información (510) del encabezado de la imagen se repite en uno respectivo de dicho al menos un sector independientemente decodificable del codificador al decodificador; y
reconstruir el contenido de la imagen de dicho sector respectivo independiente del contenido de la imagen fuera de dicho sector respectivo, en el que dicha reconstrucción independiente comprende realizar el suavizado (700) de superposición de los bordes de bloque dentro de dicho contenido de imagen de dicho sector respectivo, excepto en los bordes de límite de dicho sector respectivo, en el que el suavizado de superposición incluye operaciones de filtrado acopladas a una transformación basada en bloques para simular una transformación superpuesta.

7. El procedimiento de la reivindicación 6, en el que el parámetro de dirección es un índice de fila de macrobloque inicial del sector.

8. El procedimiento de la reivindicación 6, en el que la indicación es un valor de referencia.

9. El procedimiento de la reivindicación 6, en el que dicha reconstrucción independiente comprende además:
realizar el desbloqueo de los bordes del bloque dentro de dicho contenido de imagen de dicho sector respectivo, excepto en los bordes del límite de dicho sector respectivo.

10. Al menos un programa legible por ordenador que lleva un medio que tiene el módulo de software que es ejecutable por una unidad de procesamiento para realizar un procedimiento de una de las reivindicaciones 1 a 5.

11. Al menos un programa legible por ordenador que lleva un medio que tiene el módulo de software que es ejecutable por una unidad de procesamiento para realizar un procedimiento de una de las reivindicaciones 6 a 9.

Figura 1

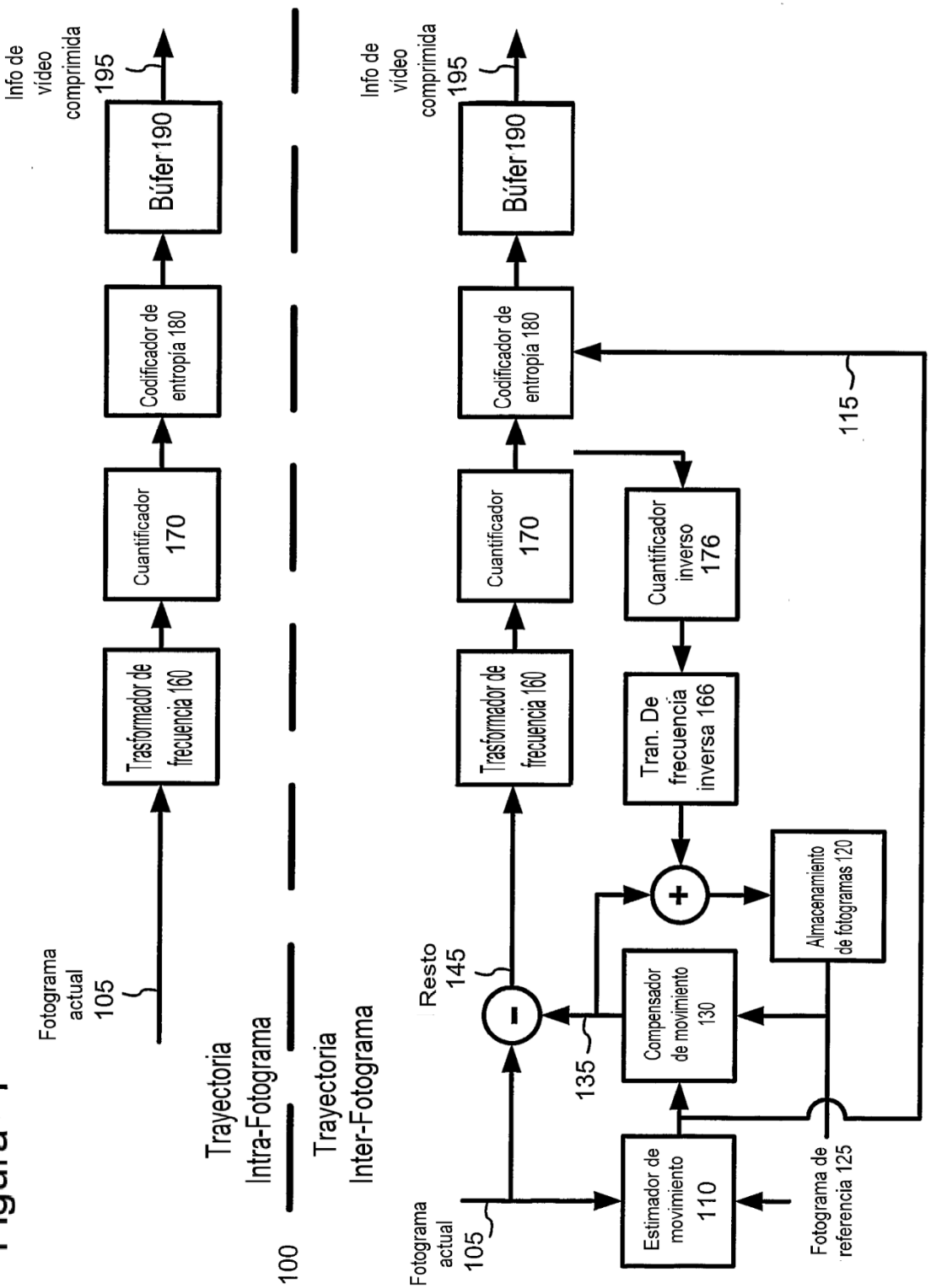

Info de vídeo comprimida 195
Búfer 190
Codificador de entropía 180
Cuantificador 170
Trasformador de frecuencia 160
Fotograma actual 105
Trayectoria Intra-Fotograma
Trayectoria Inter-Fotograma
100
Resto 145
Cuantificador inverso 176
Tran. De frecuencia inversa 166
Almacenamiento de fotogramas 120
Compensador de movimiento 130
Estimador de movimiento 110
135
115
Fotograma de referencia 125

Figura 2

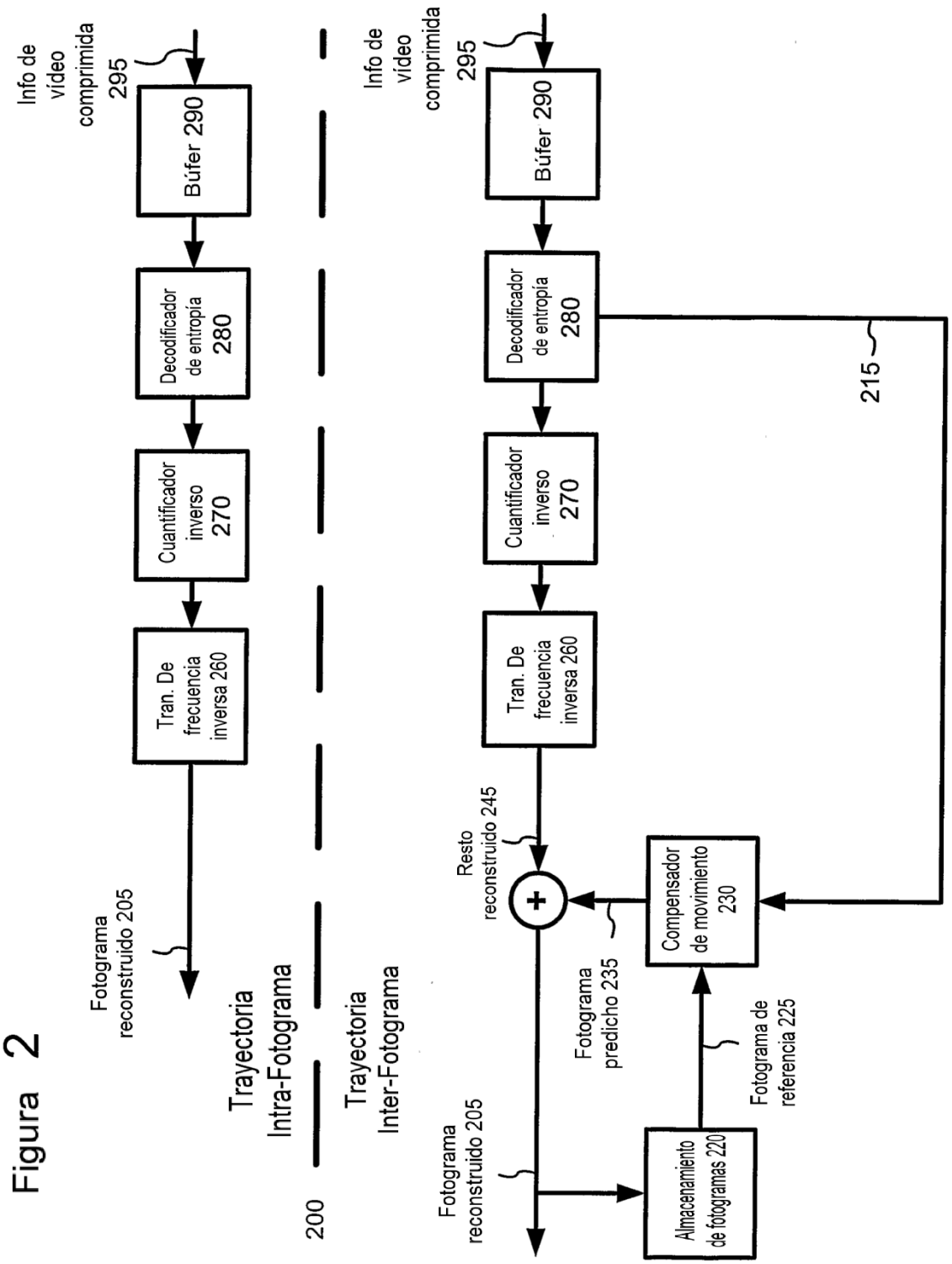

Info de vídeo comprimida 295
Búfer 290
Decodificador de entropía 280
Cuantificador inverso 270
Tran. De frecuencia inversa 260
Fotograma reconstruido 205
Trayectoria Intra-Fotograma
Trayectoria Inter-Fotograma
200
Info de vídeo comprimida 295
Búfer 290
Decodificador de entropía 280
215
Cuantificador inverso 270
Tran. De frecuencia inversa 260
Resto reconstruido 245
+
Compensador de movimiento 230
Fotograma predicho 235
Fotograma reconstruido 205
Almacenamiento de fotogramas 220
Fotograma de referencia 225

Figura 3

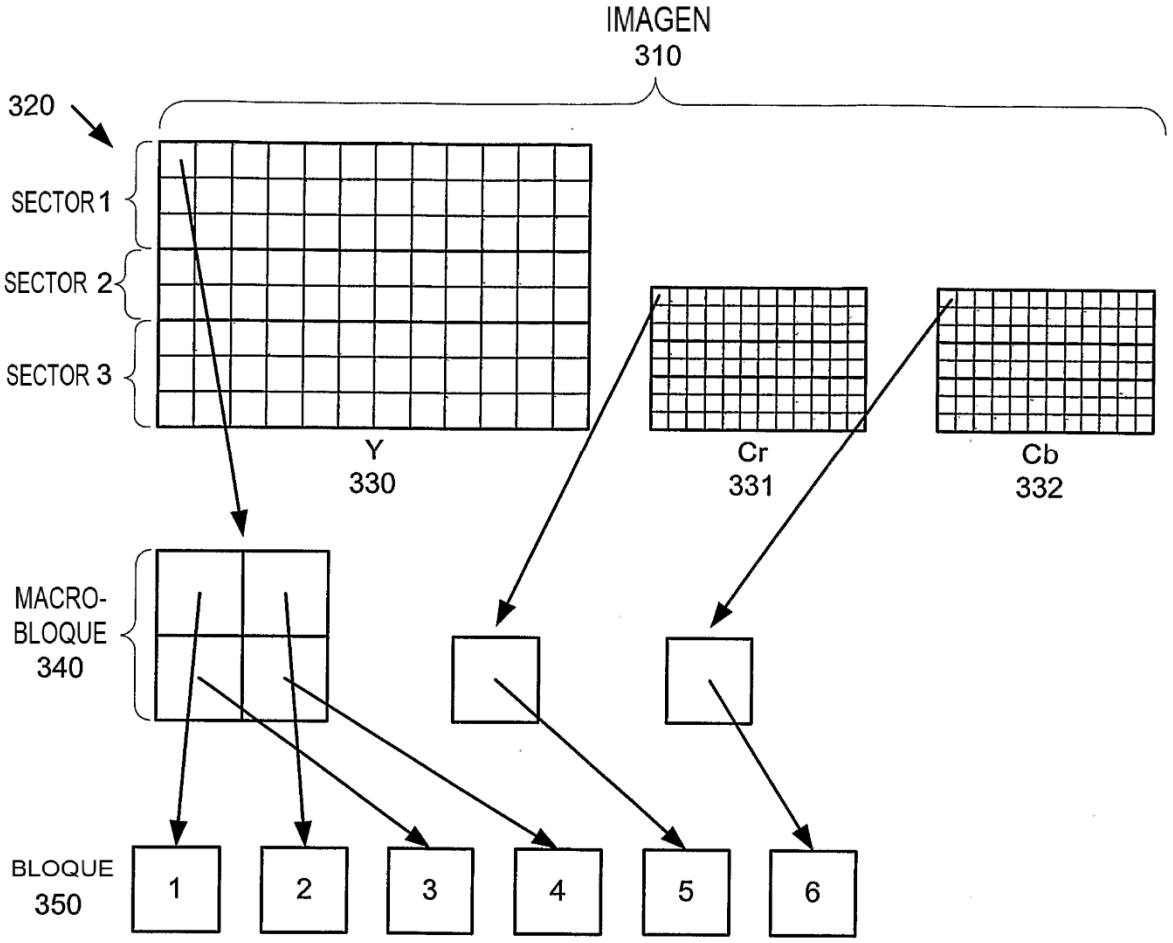


IMAGEN
310
320
SECTOR 1
SECTOR 2
SECTOR 3
Y
330
Cr
331
Cb
332
MACRO-
BLOQUE
340
BLOQUE
350
1
2
3
4
5
6

Figura 4

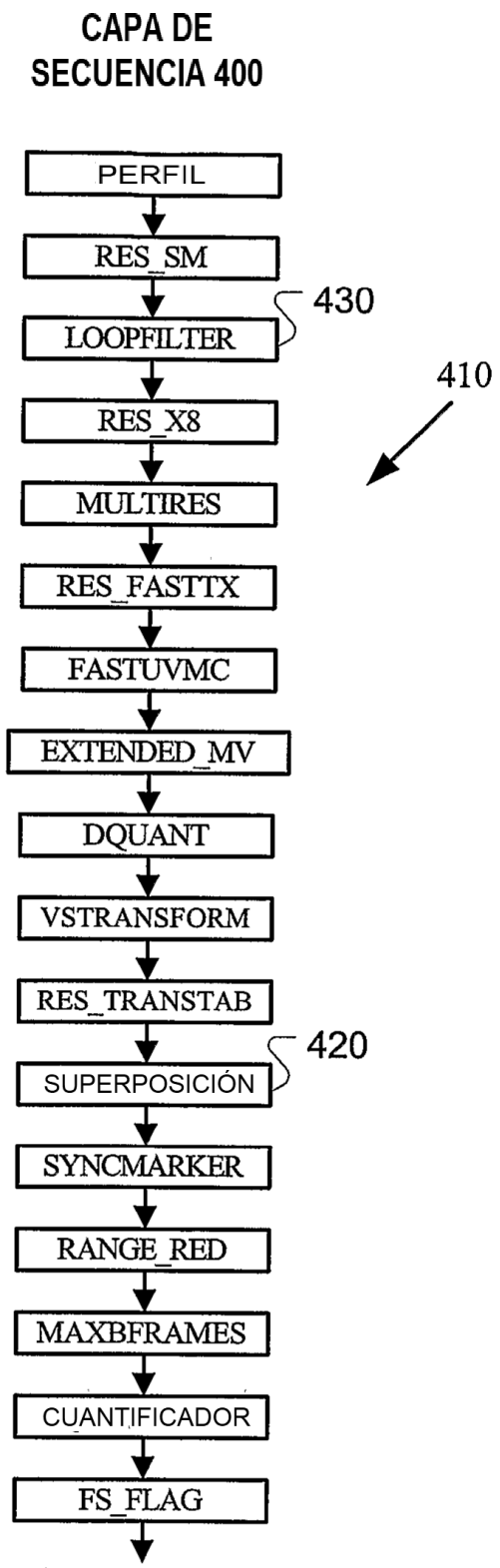

CAPA DE SECUENCIA 400
PERFIL
RES_SM
LOOPFILTER
RES_X8
MULTIRES
RES_FASTTX
FASTUVMC
EXTENDED_MV
DQUANT
VSTRANSFORM
RES_TRANSTAB
SUPERPOSICIÓN
SYNCMARKER
RANGE_RED
MAXBFRAMES
CUANTIFICADOR
FS_FLAG
430
410
420

Figura 5

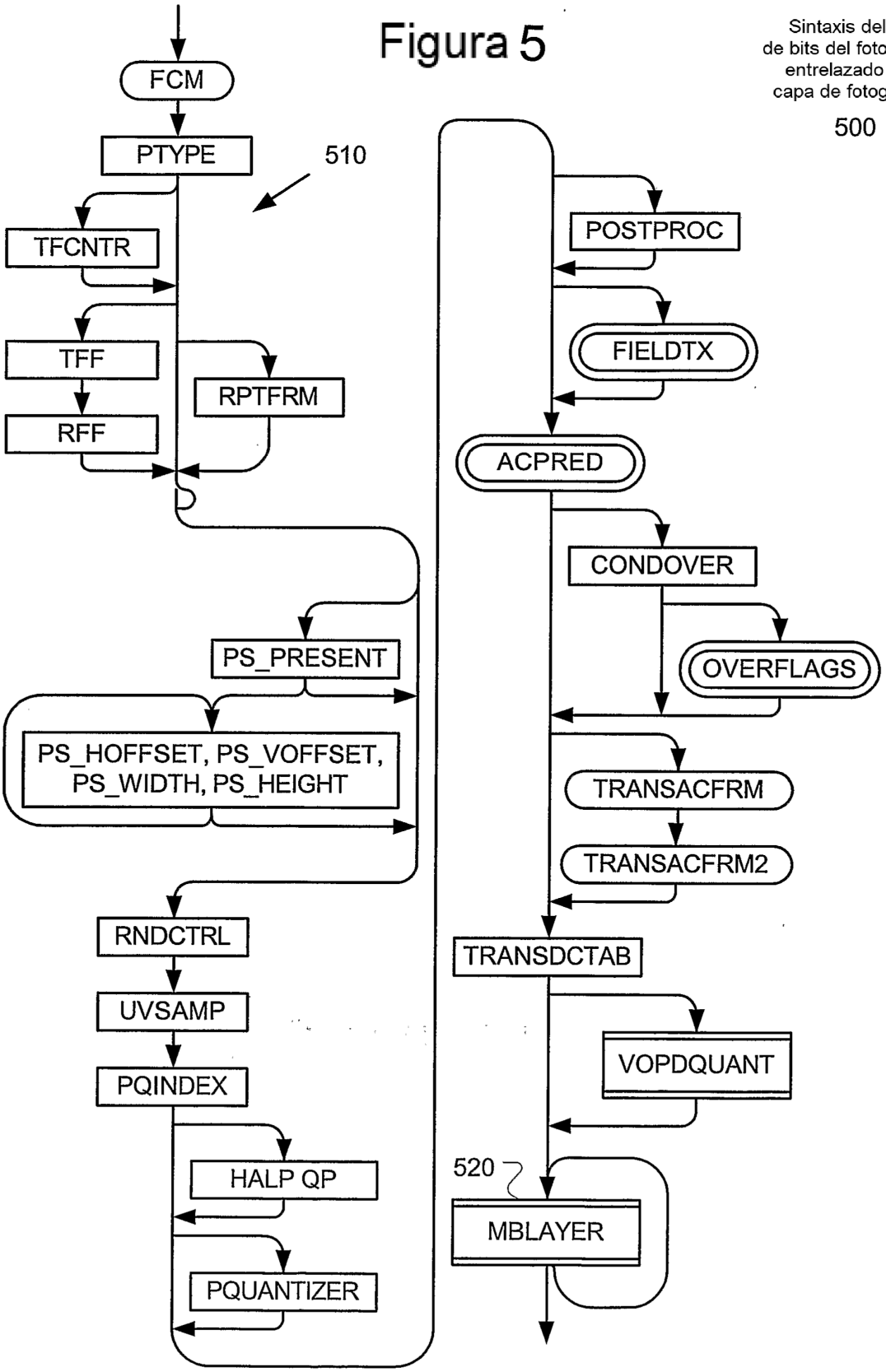

Sintaxis del flujo de bits del fotograma I entrelazado de la capa de fotogramas
500
FCM
PTYPE
510
TFCNTR
TFF
RFF
RPTFRM
PS_PRESENT
PS_HOFFSET, PS_VOFFSET, PS_WIDTH, PS_HEIGHT
RNDCTRL
UVSAMP
PQINDEX
HALP QP
PQUANTIZER
POSTPROC
FIELDTX
ACPRED
CONDOVER
OVERFLAGS
TRANSACFRM
TRANSACFRM2
TRANSDCTAB
VOPDQUANT
520
MBLAYER

Figura 6

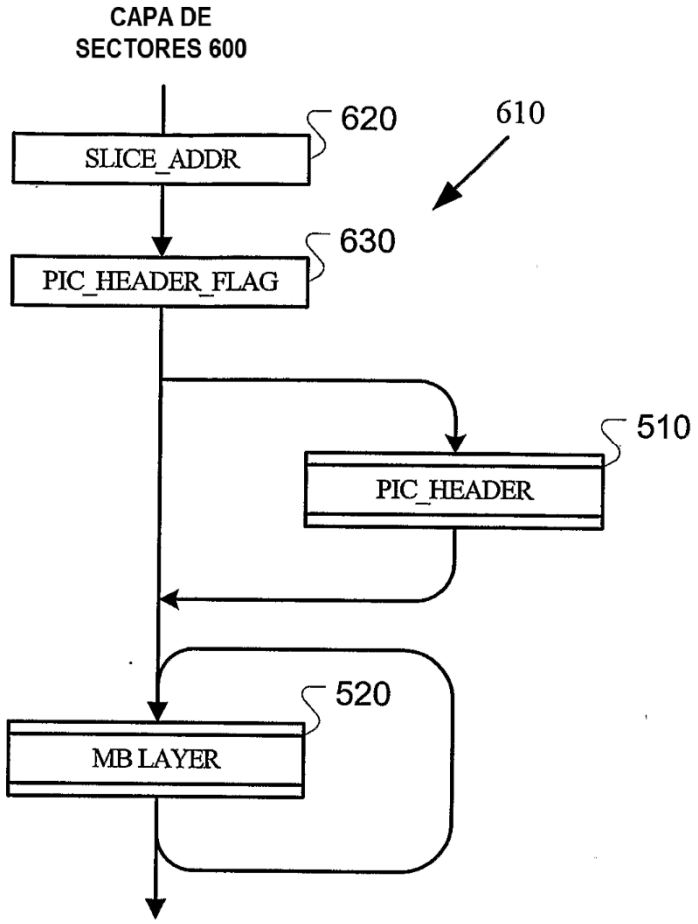

CAPA DE
SECTORES 600
620
610
SLICE_ADDR
630
PIC_HEADER_FLAG
510
PIC_HEADER
520
MB LAYER

Figura 7

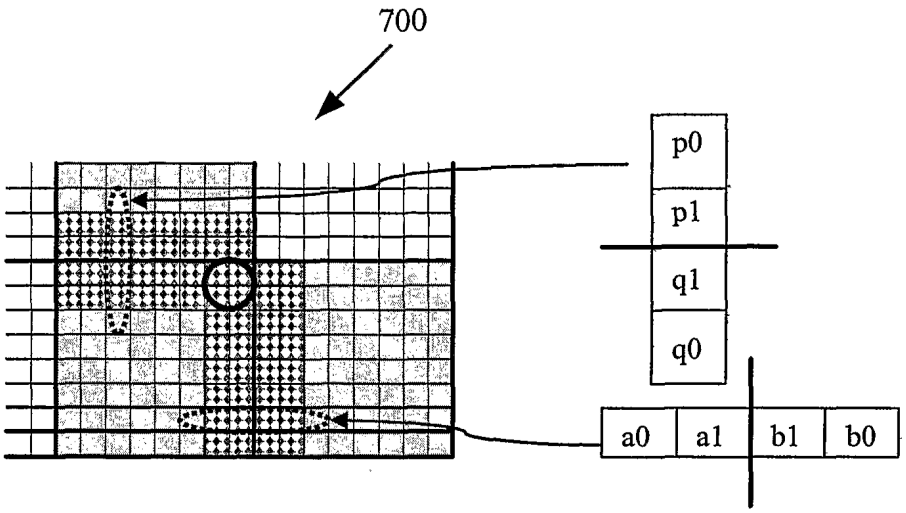
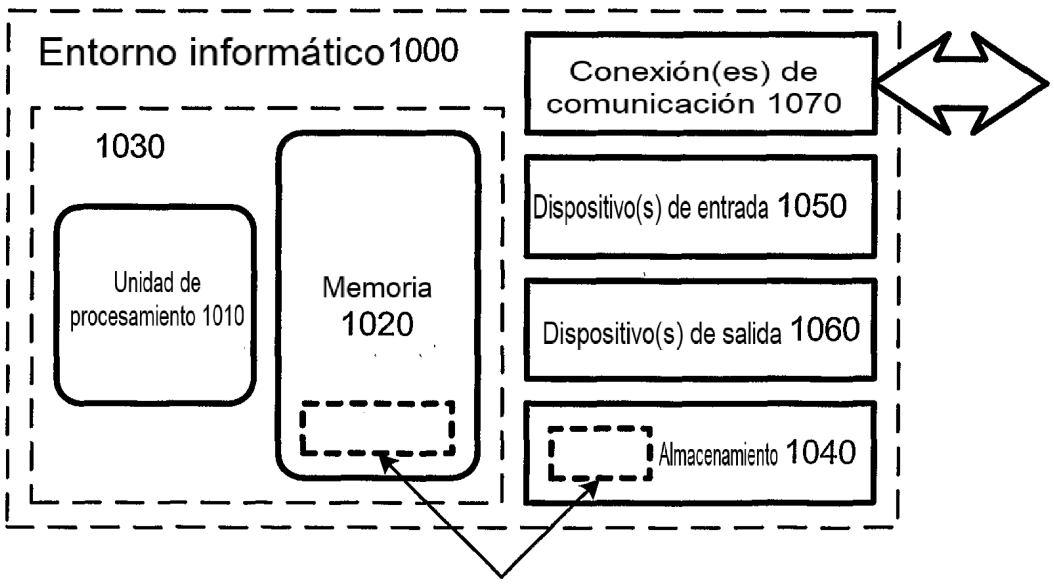


Figura 10



Software 1080 que implementa codificador/decodificador de vídeo con transformación de solapado condicional

Figura 8

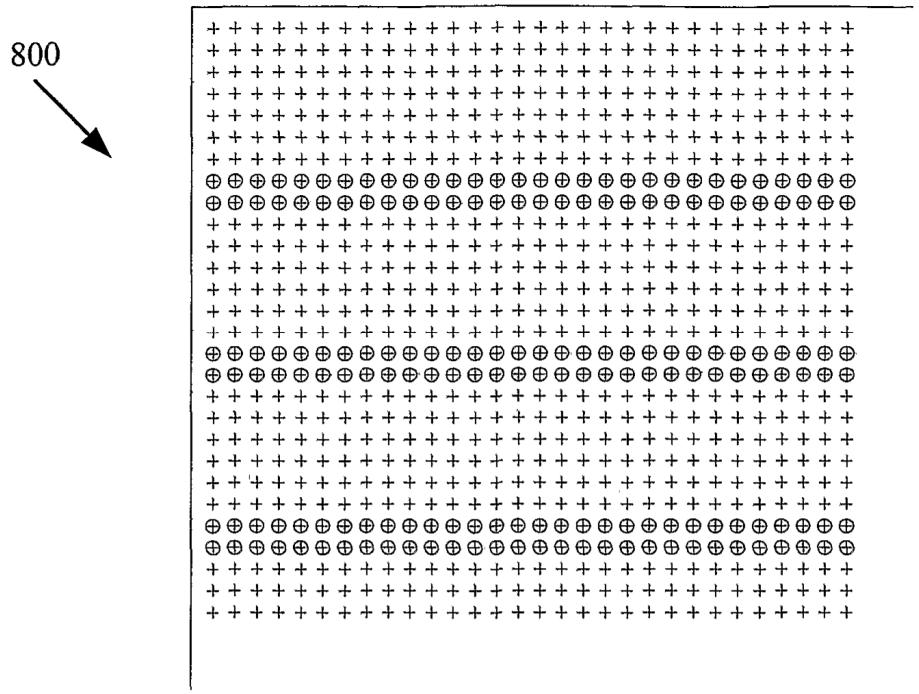


Figura 9

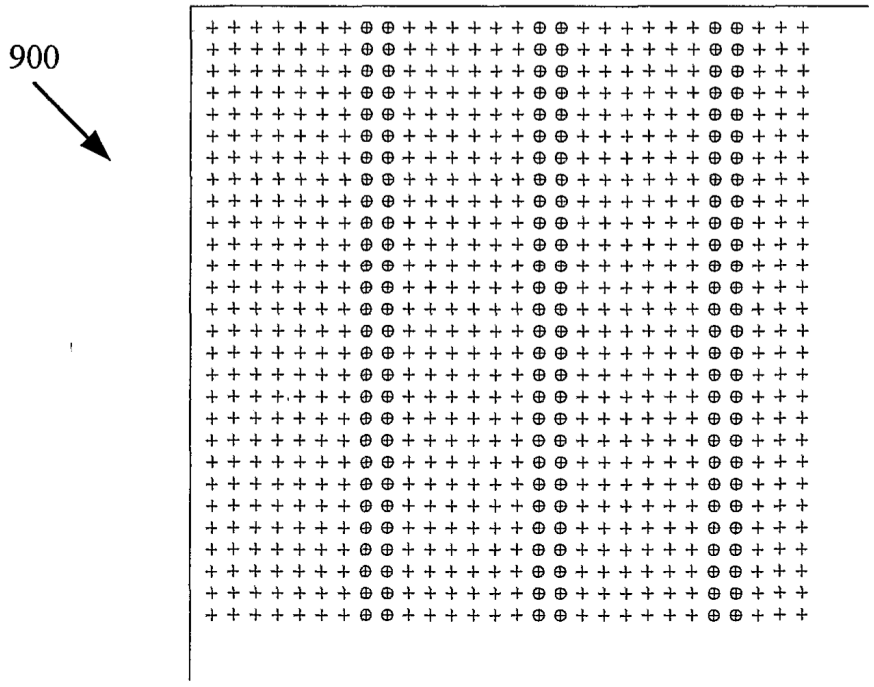


Figura 11

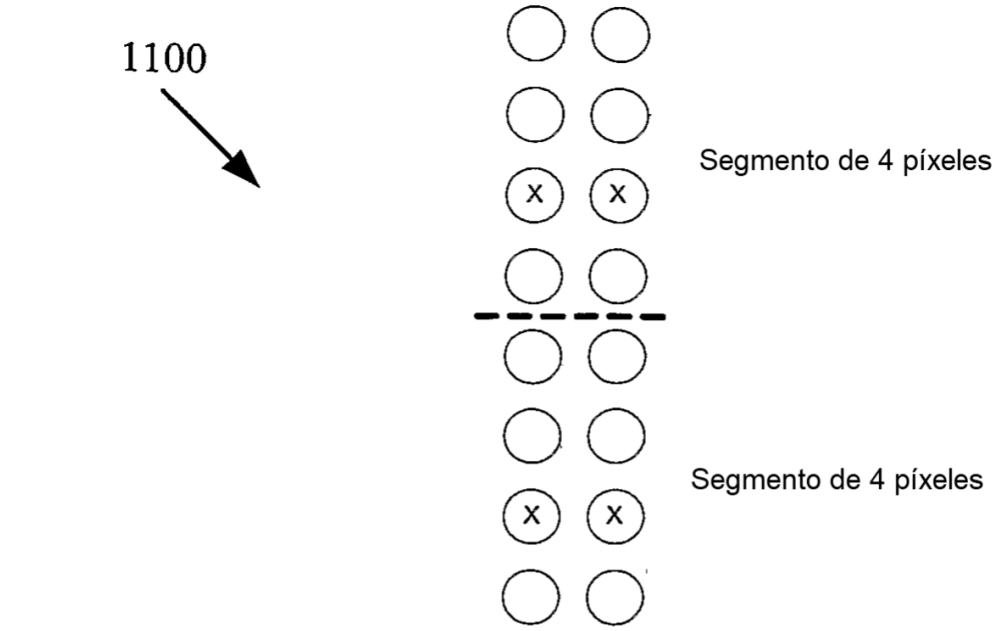


Figura 12

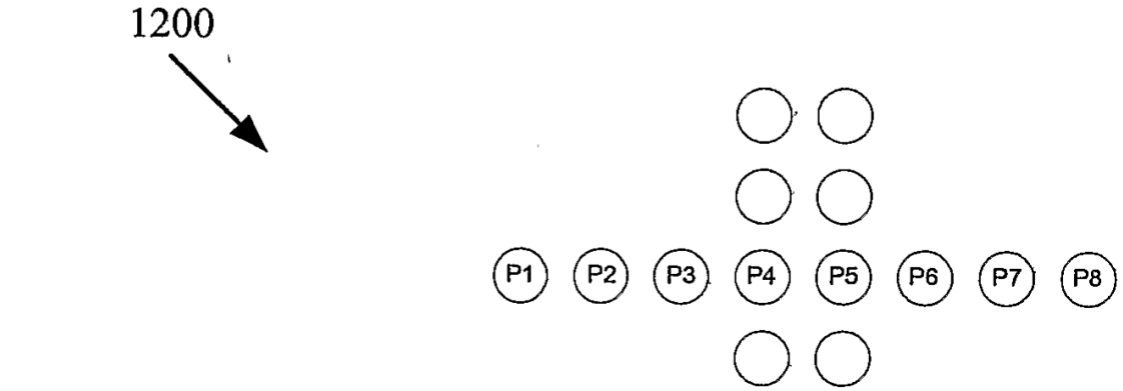
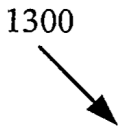
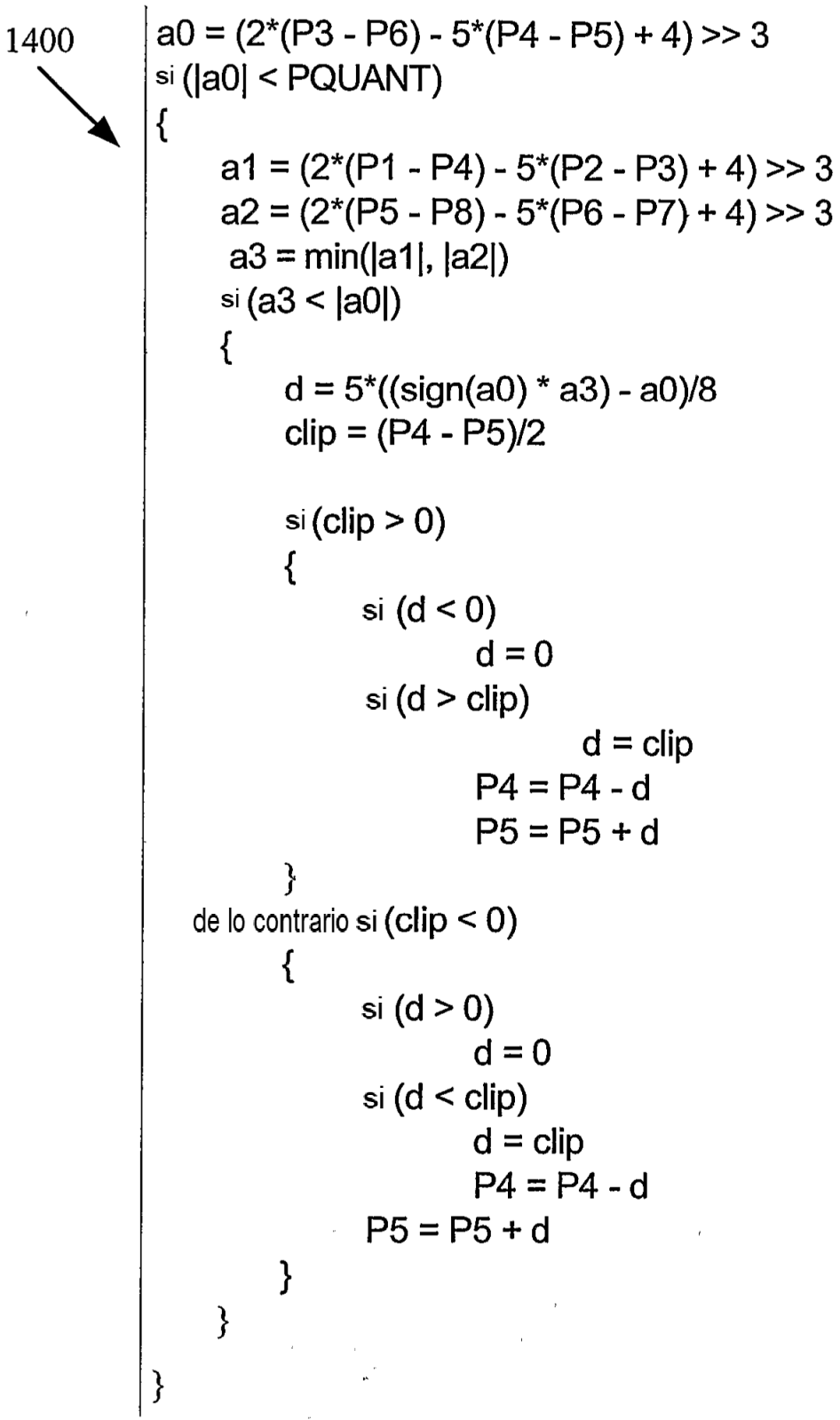


Figura 13




```
filter_other_3_pixels = verdadero
a0 = (2*(P3 - P6) - 5*(P4 - P5) + 4) >> 3
si (|a0| < PQUANT) {
    a1 = (2*(P1 - P4) - 5*(P2 - P3) + 4) >> 3
    a2 = (2*(P5 - P8) - 5*(P6 - P7) + 4) >> 3
    a3 = min(|a1|, |a2|)
    si (a3 < |a0|)
    {
        d = 5*((sign(a0) * a3) - a0)/8
        clip = (P4 - P5)/2
        si (clip == 0)
            filter_other_3_pixels = falso
        de lo contrario
        {
            si (clip > 0)
            {
                si (d < 0)
                    d = 0
                si (d > clip)
                    d = clip
            }
            de lo contrario
            {
                si (d > 0)
                    d = 0
                si (d < clip)
                    d = clip
            }
                P4 = P4 - d
                P5 = P5 + d
        }
    }
    de lo contrario
        filter_other_3_pixels = falso
}
de lo contrario
        filter_other_3_pixels = falso
```

Figura 14


1400
a0 = (2*(P3 - P6) - 5*(P4 - P5) + 4) >> 3
si (|a0| < PQUANT)
{
a1 = (2*(P1 - P4) - 5*(P2 - P3) + 4) >> 3
a2 = (2*(P5 - P8) - 5*(P6 - P7) + 4) >> 3
a3 = min(|a1|, |a2|)
si (a3 < |a0|)
{
d = 5*((sign(a0) * a3) - a0)/8
clip = (P4 - P5)/2
si (clip > 0)
{
si (d < 0)
d = 0
si (d > clip)
d = clip
P4 = P4 - d
P5 = P5 + d
}
de lo contrario si (clip < 0)
{
si (d > 0)
d = 0
si (d < clip)
d = clip
P4 = P4 - d
P5 = P5 + d
}
}
}