

19



OFICINA ESPAÑOLA DE
PATENTES Y MARCAS

ESPAÑA



11 Número de publicación: **2 963 964**

51 Int. Cl.:

H04N 19/523 (2014.01)

H04N 19/42 (2014.01)

H04N 19/82 (2014.01)

12

TRADUCCIÓN DE PATENTE EUROPEA

T3

96 Fecha de presentación y número de la solicitud europea: **22.06.2012** **E 21152044 (0)**

97 Fecha y número de publicación de la concesión europea: **04.10.2023** **EP 3826307**

54 Título: **Método para predicción con compensación de movimiento**

30 Prioridad:

24.06.2011 US 201161501112 P

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

03.04.2024

73 Titular/es:

NTT DOCOMO, INC. (100.0%)
Sanno Park Tower, 11-1 Nagata-cho 2-chome
Chiyoda-ku, Tokyo 100-6150, JP

72 Inventor/es:

BOSSEN, FRANK JAN

74 Agente/Representante:

BERTRÁN VALLS, Silvia

ES 2 963 964 T3

Aviso: En el plazo de nueve meses a contar desde la fecha de publicación en el Boletín Europeo de Patentes, de la mención de concesión de la patente europea, cualquier persona podrá oponerse ante la Oficina Europea de Patentes a la patente concedida. La oposición deberá formularse por escrito y estar motivada; sólo se considerará como formulada una vez que se haya realizado el pago de la tasa de oposición (art. 99.1 del Convenio sobre Concesión de Patentes Europeas).

DESCRIPCIÓN

Método para predicción con compensación de movimiento

5 Solicitudes relacionadas

El presente documento de patente reivindica el beneficio de la fecha de presentación según el 35 U.S.C §119(e) de la solicitud de patente estadounidense provisional con n.º de serie 6.501.122, presentada el 24 de junio de 2011.

10 Antecedentes de la invención

1. Campo de la invención

La presente invención se refiere a una compensación de movimiento para una predicción inter-tramas y en particular a una interpolación de muestras fraccionaria usada en la compensación de movimiento y que logra baja complejidad y alta precisión.

2. Descripción de la técnica relacionada

El vídeo digital requiere una gran cantidad de datos para representar todas y cada una de las tramas de una secuencia de vídeo digital (por ejemplo, series de tramas) de manera no comprimida. Para la mayoría de aplicaciones, no es factible transmitir vídeo digital no comprimido a través de redes informáticas debido a las limitaciones de ancho de banda. Además, el vídeo digital no comprimido requiere una gran cantidad de espacio de almacenamiento. El vídeo digital normalmente se codifica de alguna manera para reducir los requisitos de almacenamiento y reducir los requisitos de ancho de banda.

Una técnica para codificar vídeo digital es la predicción inter-tramas, o inter-predicción. La inter-predicción aprovecha las redundancias temporales entre diferentes tramas. Las tramas de vídeo adyacentes temporalmente incluyen normalmente bloques de píxeles que permanecen sustancialmente idénticos. Durante el proceso de codificación, un vector de movimiento interrelaciona el movimiento de un bloque de píxeles en una trama con un bloque de píxeles similares en otra trama. Por consiguiente, no se requiere que el sistema codifique el bloque de píxeles dos veces, sino que más bien codifica el bloque de píxeles una vez y proporciona un vector de movimiento para predecir el otro bloque de píxeles.

Otra técnica para codificar vídeo digital es la predicción intra-tramas o intra-predicción. La intra-predicción codifica una trama o una parte de la misma sin referencia a píxeles en otras tramas. La intra-predicción aprovecha las redundancias espaciales entre bloques de píxeles dentro de una trama. Dado que bloques de píxeles adyacentes espacialmente tienen en general atributos similares, se mejora la eficacia del proceso de codificación haciendo referencia a la correlación espacial entre bloques adyacentes. Esta correlación puede aprovecharse mediante la predicción de un bloque objetivo basándose en modos de predicción usados en bloques adyacentes.

En la inter-predicción, se predice una imagen recibida, basándose en estimación y compensación de movimiento. Los objetos en movimiento en vídeo a menudo aparecen de trama en trama en los que la totalidad o parte de los mismos están reubicados en esas tramas posteriores. A pesar de esas reubicaciones, la correlación entre la secuencia de las tramas es alta y da lugar a redundancia. Esta redundancia temporal puede reducirse comparando y relacionando las muestras en la trama actual con la ubicación del mismo objeto en las tramas de referencia. Específicamente, durante la estimación de movimiento, se compara la trama actual o una partición de la misma con tramas de referencia que pueden ser temporalmente anteriores o posteriores a la trama actual. Se compara un patrón de píxeles dentro de un intervalo de búsqueda establecido en la trama de referencia respectiva con el patrón de píxeles mostrado en la trama actual hasta que se encuentra una trama de referencia que contiene un patrón de píxeles que coincide de la mejor manera con el patrón de píxeles en la trama actual que va a codificarse. Basándose en los resultados de la comparación, se estima un vector de desplazamiento inter-tramas o un vector de movimiento. Usando el vector de movimiento estimado, la compensación de movimiento ofrece una predicción de la trama actual.

Pueden aumentarse la precisión de vector de movimiento y la eficacia de codificación aplicando interpolación a los píxeles en la imagen de referencia, que se denominan muestras en posiciones de número entero, o simplemente muestras de número entero, para aumentar la resolución de la imagen de referencia. La interpolación es para generar muestras fraccionarias entre cada muestra de número entero, usando los valores de las muestras de número entero. Cuantas más muestras fraccionarias se generen entre las muestras de número entero, más alta se volverá la resolución de la imagen de referencia, y más exacta y precisamente podrá compensarse un desplazamiento de muestra fraccionaria. Por ejemplo, con el fin de compensar de manera exacta un movimiento de un objeto en movimiento que sea un desplazamiento de solamente medio píxel, se necesita al menos interpolación de medio píxel (pel). Pueden realizarse estimación y compensación de movimiento usando varios tamaños de bloque diferentes. Pueden determinarse vectores de movimiento individuales para particiones que tengan 4x4, 4x8, 8x4, 8x8, 8x16, 16x8 ó 16x16 píxeles. La provisión de particiones de compensación de movimiento pequeñas mejora la capacidad de gestionar detalles de movimiento finos.

La norma H.264/AVC toma un enfoque de 2 etapas y logra una compensación de movimiento de hasta una resolución de un cuarto de píxel. En la norma H.264/AVC, la primera etapa usa un filtro de 6 derivaciones para generar valores intermedios a una resolución de medio píxel a partir de los valores de muestras de número entero circundantes. En la segunda etapa, se calcula el promedio de los valores de muestras de número entero y los valores intermedios o se calcula el promedio de los valores intermedios entre sí para generar muestras fraccionarias en posiciones de un cuarto de píxel, o simplemente muestras de un cuarto de píxel. En segmentos B, puede calcularse además el promedio de dos muestras fraccionarias de predicciones a partir de dos predicciones. Obsérvese, sin embargo, múltiples operaciones de promedio, cuando se realizan en cascada, introducen errores de redondeo que afectan de manera adversa a la exactitud y a la eficacia de compensación de movimiento. Las propuestas D321 y E242 del Equipo de Colaboración Conjunta de Codificación de Vídeo (Joint Collaborative Team on Video Coding, JCT-VC) abordan el problema del error de redondeo asociado con el cálculo de promedio bidireccional. Estos documentos sugieren que se limite una operación de redondeo a tener lugar en la última etapa de cálculo de promedio bidireccional después de añadir dos predicciones.

El borrador E603 del JCT-VC divulga el uso de un filtro de 8 derivaciones para lograr la resolución de un cuarto de píxel. En el documento E603, algunas de las muestras de un cuarto de píxel se derivan aplicando un filtro de 8 derivaciones a las muestras de número entero más cercanas y truncando los resultados filtrados hasta una profundidad de bits predeterminada. El resto de las muestras de un cuarto de píxel se deriva mediante dos procesos. En el primer proceso, se derivan valores intermedios aplicando el filtro de 8 derivaciones a las muestras de número entero más cercanas en la dirección horizontal. En el segundo proceso, se aplica el filtro de 8 derivaciones a los valores intermedios en la dirección horizontal y se truncan los resultados filtrados hasta una profundidad de bits predeterminada. Este enfoque de 2 procesos es ventajoso ya que no se requiere ningún orden fijo para el filtrado vertical y el filtrado horizontal en el segundo proceso, y por tanto no es necesaria ninguna señalización a un decodificador con respecto al orden del filtrado vertical y del filtrado horizontal en el segundo proceso. Sin embargo, la compensación de movimiento comentada en el documento E603 requiere la definición de operaciones de filtrado adicionales para generar los valores intermedios. La operación de filtrado aplicada a los valores intermedios es costosa y requiere una alta complejidad de computación, en particular para datos de vídeo con una alta profundidad de bits.

Además, en la compensación de movimiento comentada en el documento E603, no se truncan los valores intermedios para garantizar la exactitud de las muestras de un cuarto de píxel calculadas a partir de los mismos. Por tanto, la precisión en bits de los valores calculados no es constante durante la compensación de movimiento comentada en el documento E603. Al final del primer proceso expuesto anteriormente, se aumenta la precisión de los valores de muestra resultantes una cantidad determinada por la ganancia del filtro de 8 derivaciones. Aplicando el filtro de 8 derivaciones a los valores intermedios, se aumenta entonces de nuevo la precisión la misma cantidad que en el primer proceso antes del truncamiento para dar una precisión predeterminada. Por tanto, se necesita el doble de truncamiento de la resolución en el segundo proceso que en el primer proceso con el fin de devolver la resolución a la profundidad de bits original al final de la segunda etapa.

Sumario de la invención

En vista de la técnica anterior comentada anteriormente, la presente invención proporciona una operación compensación de movimiento novedosa para una predicción inter-tramas. La compensación de movimiento aplica una interpolación de muestras fraccionaria a muestras para generar muestras fraccionarias. En la compensación de movimiento según la presente invención, se aplica una primera interpolación de muestras fraccionaria a muestras en una primera dirección para generar un primer conjunto de muestras fraccionarias. Se aplica una segunda interpolación de muestras fraccionaria al primer conjunto de muestras fraccionarias en una segunda dirección para generar un segundo conjunto de muestras fraccionarias. En la presente invención, el primer conjunto de muestras fraccionarias se representa con una precisión independiente de una precisión de representación de las muestras de referencia. Las muestras de referencia se representan con una primera profundidad de bits y se convierten desde la primera profundidad de bits hasta una segunda profundidad de bits. La segunda profundidad de bits se establece igual a un número de bits disponibles para representar las muestras fraccionarias. Las muestras fraccionarias del primer y el segundo conjunto de muestras fraccionarias se representan con la segunda profundidad de bits. Se genera una muestra de predicción bidireccional calculando el promedio de dos de los segundos conjuntos de muestras fraccionarias generados, siendo los dos segundos conjuntos diferentes entre sí. Se recorta un valor de la muestra de predicción bidireccional para representarse con la primera profundidad de bits de las muestras de referencia.

En un aspecto de la presente invención, la interpolación de muestras fraccionaria aplica un filtro FIR de 8 derivaciones que tiene uno de los tres conjuntos siguientes de coeficientes para generar una muestra de un cuarto de píxel:

$[-1, 4, -10, 58, 17, -5, 1, 0];$

$[-1, 4, -11, 40, 40, -11, 4, -1];$ y

$[0, 1, -5, 17, 58, -10, 4, -1].$

En un aspecto de la presente invención, la interpolación de muestras fraccionaria comprende truncar cada uno de los resultados filtrados de modo que las muestras fraccionarias tengan una precisión deseada. La precisión deseada puede ser constante a lo largo del primer y el segundo procesos de interpolación fraccionaria.

- 5 El proceso de truncamiento puede implicar redondear resultados truncados hacia menos infinito. Esto se logra al no añadir ninguna desviación a los resultados filtrados antes del proceso de truncamiento.

Además, en otro aspecto de la presente invención, las muestras pueden ampliarse a escala hasta una segunda profundidad de bits que es constante para las muestras, con independencia de las profundidades de bits originales de las muestras. La segunda profundidad de bits puede establecerse igual a un número de bits disponibles para representar la muestra, y se establece un factor de ajuste a escala igual al número de bits disponibles para representar la muestra menos la profundidad de bits original de la muestra.

Además, en otro aspecto de la presente invención, las muestras ajustadas a escala se restan con una desviación.

Breve descripción de los dibujos

La figura 1 es un diagrama de bloques que muestra una arquitectura de hardware a modo de ejemplo en la cual puede implementarse la presente invención.

La figura 2 es un diagrama de bloques que muestra una vista general de un codificador de vídeo al cual puede aplicarse la presente invención.

La figura 3 es un diagrama de bloques que muestra una vista general de un decodificador de vídeo al cual puede aplicarse la presente invención.

La figura 4 es un diagrama de bloques que muestra los módulos funcionales de un codificador.

La figura 5 es un diagrama de flujo que muestra un proceso de codificación realizado por el codificador de vídeo.

La figura 6 es un diagrama de flujo que muestra los procesos de interpolación de muestras fraccionaria según una realización de la presente invención.

La figura 7 es una ilustración del filtrado para la interpolación de muestras fraccionaria.

La figura 8 es un diagrama de bloques que muestra los módulos funcionales de un decodificador según una realización de la presente invención.

La figura 9 es un diagrama de flujo que muestra un proceso de decodificación realizado por el decodificador de vídeo según una realización de la presente invención.

Descripción detallada de los dibujos y las realizaciones actualmente preferidas

La figura 1 muestra una arquitectura de hardware a modo de ejemplo de un ordenador 100 en el cual puede implementarse la presente invención. Obsérvese que la arquitectura de hardware mostrada en la figura 1 puede ser común tanto en un codificador de vídeo como en un decodificador de vídeo que implementan las realizaciones de la presente invención. El ordenador 100 incluye un procesador 101, una memoria 102, un dispositivo de almacenamiento 105 y uno o más dispositivos de entrada y/o salida (I/O) 106 (o periféricos) que están acoplados en comunicación mediante una interfaz local 107. La interfaz local 107 puede ser, por ejemplo, pero no se limita a, uno o más buses u otras conexiones por cable o inalámbricas, tal como se conoce en la técnica.

El procesador 101 es un dispositivo de hardware para ejecutar software, particularmente el que está almacenado en la memoria 102. El procesador 101 puede ser cualquier procesador fabricado a medida o disponible en el mercado, una unidad de procesamiento central (CPU), un procesador auxiliar de entre varios procesadores asociados con el ordenador 100, un microprocesador basado en semiconductor (en forma de un microchip o conjunto de chips), o en general cualquier dispositivo para ejecutar instrucciones de software.

La memoria 102 comprende un medio legible por ordenador, que puede incluir uno cualquiera o una combinación de elementos de memoria volátil (por ejemplo, memoria de acceso aleatorio (RAM, tal como DRAM, SRAM, SDRAM, etc.)) y elementos de memoria no volátil (por ejemplo, ROM, disco duro, cinta, CDROM, etc.). Además, la memoria 102 puede incorporar medios de almacenamiento electrónicos, magnéticos, ópticos y/o de otros tipos. Un medio legible por ordenador puede ser cualquier medio que pueda almacenar, comunicar, propagar o transportar el programa para su uso por o en relación con el sistema, aparato o dispositivo de ejecución de instrucciones. Obsérvese que la memoria 102 puede tener una arquitectura distribuida, en la que diversos componentes están situados alejados unos de otros, pero a la que puede acceder el procesador 101.

El software 103 en la memoria 102 puede incluir uno o más programas independientes, cada uno de los cuales contiene un listado ordenado de instrucciones ejecutables para implementar funciones lógicas del ordenador 100, tal como se describe a continuación. En el ejemplo de la figura 1, el software 103 en la memoria 102 define la funcionalidad de codificación de vídeo o de decodificación de vídeo del ordenador 100 según la presente invención. Además, aunque no se requiere, es posible que la memoria 102 contenga un sistema operativo (O/S) 104. El sistema operativo 104 controla esencialmente la ejecución de programas informáticos y proporciona planificación, control de entrada-salida, gestión de archivos y datos, gestión de memoria y control de comunicación y servicios relacionados.

El dispositivo de almacenamiento 105 del ordenador 100 puede ser uno de muchos tipos diferentes de dispositivo de almacenamiento, incluyendo un dispositivo de almacenamiento estacionario o dispositivo de almacenamiento portátil. Como ejemplo, el dispositivo de almacenamiento 105 puede ser una cinta magnética, un disco, una memoria *flash*, una memoria volátil o un dispositivo de almacenamiento diferente. Además, el dispositivo de almacenamiento 105 puede ser una tarjeta de memoria digital segura o cualquier otro dispositivo de almacenamiento extraíble 105.

Los dispositivos de I/O 106 pueden incluir dispositivos de entrada, por ejemplo, pero sin limitarse a, una pantalla táctil, un teclado, un ratón, un escáner, un micrófono u otro dispositivo de entrada. Además, los dispositivos de I/O 106 también pueden incluir dispositivos de salida, por ejemplo, pero sin limitarse a, una pantalla u otros dispositivos de salida. Los dispositivos de I/O 106 pueden incluir además dispositivos que se comuniquen tanto a través de entradas como a través de salidas, por ejemplo, pero sin limitarse a, un modulador/demodulador (por ejemplo, un módem; para acceder a otro dispositivo, sistema o red), un transceptor de radiofrecuencia (RF), inalámbrico u otro, una interfaz telefónica, un puente, un enrutador u otros dispositivos que funcionan como entrada y como salida.

Como bien conocen los expertos habituales en la técnica, la compresión de vídeo se logra extrayendo información redundante en una secuencia de vídeo. Existen muchas normas de codificación de vídeo diferentes, ejemplos de las cuales incluyen MPEG-1, MPEG-2, MPEG-4, H.261, H.263 y H.264/AVC. Obsérvese que no se pretende que la presente invención esté limitada en cuanto a la aplicación de cualquier norma de codificación de vídeo específica. Sin embargo, la siguiente descripción de la presente invención se proporciona usando el ejemplo de la norma H.264/AVC, que se incorpora mediante referencia en el presente documento. La norma H.264/AVC es la norma de codificación de vídeo más reciente y logra una mejora de rendimiento significativa con respecto a las normas de codificación anteriores tales como MPEG-1, MPEG-2, H.261 y H.263.

En la norma H.264/AVC, cada trama o imagen de un vídeo puede descomponerse en varios segmentos. Los segmentos se dividen entonces en bloques de 16x16 píxeles denominados macrobloques, que entonces pueden dividirse adicionalmente en bloques de 8x16, 16x8, 8x8, 4x8, 8x4, hasta de 4x4 píxeles. Existen cinco tipos de segmentos soportados por la norma H.264/AVC. En los segmentos I, todos los macrobloques se codifican usando intra-predicción. En los segmentos P, pueden codificarse macrobloques usando intra o inter-predicción. Los segmentos P permiten que se use solamente una señal de predicción con compensación de movimiento (MCP) por cada macrobloque. En los segmentos B, pueden codificarse macrobloques usando intra o inter-predicción. Pueden usarse dos señales de MCP por cada predicción. Los segmentos SP permiten que se conmuten segmentos P entre flujos de vídeo diferentes de manera eficaz. Un segmento SI es una coincidencia exacta para un segmento SP para acceso aleatorio o recuperación de errores, mientras se usa solamente intra-predicción.

La figura 2 muestra una vista general de un codificador de vídeo al cual puede aplicarse la presente invención. Los bloques mostrados en la figura representan módulos funcionales realizados por el procesador 101 que ejecuta el software 103 en la memoria 102. Se alimenta una imagen de trama de vídeo 200 a un codificador de vídeo 201. El codificador de vídeo trata la imagen 200 en unidades de macrobloques 200A. Cada macrobloque contiene varios píxeles de imagen 200. En cada macrobloque, se realiza una transformación en coeficientes de transformada seguida por una cuantificación en niveles de coeficiente de transformada. Además, se usa intra-predicción o inter-predicción, para no realizar las etapas de codificación directamente en los datos de píxel sino en las diferencias entre valores de píxel y sus predicciones, logrando por tanto valores pequeños que se comprimen con mayor facilidad.

Para cada segmento, el codificador 201 genera varios elementos de sintaxis, que forman una versión codificada de los macrobloques del segmento respectivo. Todos los elementos de datos residuales en los elementos de sintaxis, que están relacionados con la codificación de coeficientes de transformada, tales como los niveles de coeficiente de transformada o un mapa de significación que indica niveles de coeficiente de transformada omitidos, se denominan elementos de sintaxis de datos residuales. Aparte de estos elementos de sintaxis de datos residuales, los elementos de sintaxis generados por el codificador 201 contienen elementos de sintaxis de información de control que contienen información de control sobre cómo se ha codificado cada macrobloque y cómo debe decodificarse, respectivamente. Dicho de otro modo, los elementos de sintaxis pueden dividirse en dos categorías. La primera categoría, los elementos de sintaxis de información de control, contiene los elementos relacionados con un tipo de macrobloque, tipo de submacrobloque e información acerca de modos de predicción de tipo tanto espacial como temporal, así como información de control basada en segmentos y basada en macrobloques, por ejemplo. En la segunda categoría, se combinan todos los elementos de datos residuales, tales como un mapa de significación que indica las ubicaciones de todos los coeficientes significativos en el interior de un bloque de coeficientes de transformada cuantificados y los valores de los coeficientes significativos, que se indican en unidades de niveles correspondientes a las etapas de cuantificación, y pasan a ser elementos de sintaxis de datos residuales.

El codificador 201 comprende un codificador de entropía que codifica elementos de sintaxis y genera palabras de código aritméticas para cada segmento. Al generar las palabras de código aritméticas para un segmento, el codificador de entropía aprovecha dependencias estadísticas entre los valores de datos de elementos de sintaxis en el flujo de bits de señal de vídeo. El codificador 201 emite una señal de vídeo codificada para un segmento de imagen 200 a un decodificador de vídeo 301 mostrado en la figura 3.

La figura 3 muestra una vista general de un decodificador de vídeo al que puede aplicarse la presente invención. De manera similar, los bloques mostrados en la figura representan módulos funcionales realizados por el procesador 101 que ejecuta el software 103 en la memoria 102. El decodificador de vídeo 301 recibe la señal de vídeo codificada y en primer lugar decodifica por entropía la señal para dar de nuevo los elementos de sintaxis. El decodificador 301 usa los elementos de sintaxis con el fin de reconstruir, macrobloque a macrobloque y después segmento tras segmento, las muestras de imagen 300A de píxeles en la imagen 300.

La figura 4 muestra los módulos funcionales del codificador de vídeo 201, que se indica de manera general con el número de referencia 400 en la figura 4. Estos módulos funcionales se realizan por el procesador 101 que ejecuta el software 103 en la memoria 102. Una imagen de vídeo de entrada es una trama o un campo de una imagen de vídeo natural (no comprimida) definida por puntos de muestra que representan componentes de colores originales, tales como crominancia ("croma") y luminancia ("luma"). Son posibles otros componentes tales como tono, saturación y valor. La imagen de vídeo de entrada se divide en macrobloques que representan cada uno un área de imagen cuadrada consistente en 16x16 píxeles del componente de luma del color de imagen. La imagen de vídeo de entrada también se divide en macrobloques que representan cada uno 8x8 píxeles de cada uno de los dos componentes de croma del color de imagen. En el funcionamiento de codificador general, pueden predecirse de manera temporal o espacial macrobloques introducidos usando inter o intra-predicción. Sin embargo, se supone, para el propósito de discusión, que los macrobloques 400 son o bien todos macrobloques de tipo de segmento P o bien todos macrobloques de tipo de segmento B, o bien una mezcla de ambos, y solamente se someten a inter-predicción.

El codificador de vídeo 400 mostrado en la figura 4 solamente transmite valores residuales o diferenciales entre una imagen actual y una predicción de la imagen actual. Un restador 410 obtiene estos valores residuales recibiendo un bloque de imagen actual que va a codificarse como señal de entrada IS y restando del mismo un bloque de imagen de predicción indicado por una señal de predicción PS. Tal como se ilustra en la figura 4, el codificador de vídeo 400 incluye una unidad de transformada/cuantificación 420 para emitir coeficientes de transformada cuantificados QC. La unidad de transformada/cuantificación 420 transforma el valor residual procedente del dominio espacial en el dominio de frecuencia y cuantifica el valor residual transformado. El codificador de vídeo 400 también incluye una unidad de codificación por entropía 490 para realizar codificación por entropía (codificación de longitud variable) de los coeficientes de transformada cuantificados QC y emitir el flujo de bits BS.

El codificador de vídeo 400 incluye además una unidad de cuantificación/transformada inversa 430 y un sumador 435. La unidad de cuantificación/transformada inversa 430 transforma los coeficientes de transformada cuantificados QC para dar de nuevo una señal residual RS, a la que el sumador 435 le suma la señal de predicción PS para emitir una imagen decodificada. Un filtro de desbloqueo 437 retira artefactos de bloque de la imagen decodificada y almacena la imagen decodificada limpia en una memoria de referencia 440.

El codificador de vídeo 400 incluye además una unidad de predicción/compensación de movimiento 460 y una unidad de estimación de movimiento 470. La unidad de predicción/compensación de movimiento 460 lee una imagen de referencia de la memoria de referencia 440 y aplica interpolación para generar muestras fraccionarias en las posiciones de un cuarto de píxel. La unidad de estimación de movimiento 470 también lee la imagen de referencia de la memoria de referencia 440 y estima un vector de movimiento MV para la imagen actual comparando la imagen actual y la imagen de referencia. Se emite el vector de movimiento MV a la unidad de predicción/compensación de movimiento 460 y a la unidad de codificación por entropía 490. El bloque puede dividirse en particiones que tengan 4x4, 4x8, 8x4, 8x8, 8x16, 16x8 o 16x16 píxeles. La unidad de estimación de movimiento 470 realiza la estimación de movimiento en las particiones respectivas en el bloque y emite vectores de movimiento MV para las particiones respectivas. La unidad de predicción/compensación de movimiento 460 desplaza las particiones respectivas en la imagen de referencia las cantidades y en las direcciones indicadas por los vectores de movimiento para derivar la imagen de predicción, que se proporciona como señal predictiva IS al restador 410 y al sumador 435.

La figura 5 es un diagrama de flujo que muestra un alto nivel de proceso de codificación realizado por el codificador de vídeo 400. En la etapa 502, la unidad de predicción/compensación de movimiento 460 genera un bloque de predicción. En la etapa 504, el restador 410 resta la imagen de predicción a la imagen actual para obtener un valor residual. En la etapa 506, la unidad de transformada/cuantificación 420 codifica el valor residual.

La figura 6 es un diagrama de flujo que muestra los procesos de compensación de movimiento realizados por la unidad de predicción/compensación de movimiento 460 para generar una señal de predicción según una realización de la presente invención. En la etapa 602, la unidad de predicción/compensación de movimiento 460 lee muestras de referencia de la memoria de referencia 440. La memoria de referencia 440 almacena múltiples imágenes de referencia. Tal como se comenta en la sección 8.4.2 y en sus subsecciones de la norma H.264/AVC, la ubicación de las muestras

de referencia que van a recuperarse de una imagen de referencia almacenada en la memoria de referencia 440 puede determinarse con la posición de una zona que va a predecirse dentro de la imagen actual, de un vector de movimiento MV de la zona y de un índice de trama de referencia.

En la etapa 604, las muestras de referencia recuperadas se amplían a escala para aumentar la precisión de las muestras de referencia. El factor de ajuste a escala puede determinarse con la profundidad de bits de las muestras de referencia y con el número de bits disponibles para representar la muestra de referencia. El número de bits disponibles para representar la muestra de referencia puede determinarse de tal manera que, si 16 es el número de bits disponibles máximo para la representación de la muestra de referencia, el número de bits disponibles para representar la muestra puede limitarse a 14 para prevenir un flujo excesivo o insuficiente de datos, lo que puede ocurrir durante la compensación de movimiento. Si el número de bits disponibles para representar la muestra es de 14, el factor de ajuste a escala es de 6 para la muestra de referencia que tiene una anchura de 8 bits, y las muestras de referencia recuperadas se amplían a escala desplazando sus bits 6 bits hacia la izquierda. El factor de ajuste a escala debe ser de 4 para la muestra de referencia que tiene una anchura de 10 bits, y las muestras de referencia recuperadas se amplían a escala desplazando sus bits 4 bits hacia la izquierda. Más en general, el factor de ajuste a escala puede expresarse por "el número de bits disponibles para representar la muestra de referencia" menos "la profundidad de bits de la muestra de referencia". El proceso de ampliación a escala realizado en la etapa 604 es ventajoso porque convierte muestras de referencia almacenadas en la memoria de referencia en una forma independiente de sus profundidades de bits y los procesos que siguen el proceso de ampliación a escala pasarán a ser independientes de las profundidades de bits originales de los datos de vídeo.

Opcionalmente, puede restarse una desviación al valor ampliado a escala de la muestra de referencia. Por ejemplo, si se usa una representación de 14 bits con signo para representar las muestras de referencia ampliadas a escala, el valor de la desviación que va a restarse puede ser de 8192 ($=2^{13}$), lo que lleva las muestras de referencia ampliadas a escala dentro del intervalo de -8192 a 8192. Sumar esta desviación resulta beneficioso para reducir el número de bits requeridos para representar una muestra. Puesto que el proceso de interpolación descrito a continuación funciona con valores de muestra con signo, por tanto se prefiere una representación con signo de todos los valores de muestra. En el ejemplo anterior en el que se amplían a escala muestras hasta representaciones de 14 bits, si no se resta la desviación, tendría que usarse una representación con signo de 15 bits que es más costosa que una representación con signo de 14 bits obtenida a partir de la resta de la desviación. Obsérvese que las muestras de referencia pueden ampliarse a escala y desviarse cuando estén almacenadas en la memoria de referencia. Si las muestras de referencia se almacenan en la memoria de referencia después de su ampliación a escala y desviación, la etapa 604 no es necesaria.

Entonces se aumenta la resolución de las muestras de referencia ampliadas a escala usando interpolación de muestras fraccionaria para generar muestras fraccionarias en posiciones de un cuarto de píxel. La figura 7 es una ilustración del filtrado para la compensación de movimiento de muestras fraccionarias. En la figura 7, las letras mayúsculas A_{ij} indican las muestras de referencia ampliadas a escala. Estas muestras se denominan muestras de número entero. Las otras muestras mostradas por las letras minúsculas, tales como $a_{0,0}$, $b_{0,0}$, $c_{0,0}$, $d_{0,0}$,... y $r_{0,0}$, son muestras fraccionarias en posiciones de un cuarto de píxel. En la etapa 606, se deriva la muestra de un cuarto de píxel $a_{0,0}$, $b_{0,0}$ o $c_{0,0}$ mediante las siguientes ecuaciones de interpolación:

$$a_{0,0} = (-A_{-2,0} + 4*A_{-1,0} - 10*A_{0,0} + 58*A_{1,0} + 17*A_{2,0} - 5*A_{3,0} + A_{4,0}) >> 6$$

$$b_{0,0} = (-A_{-1,0} + 4*A_{-2,0} - 11*A_{-1,0} + 40*A_{0,0} + 40*A_{1,0} - 11*A_{2,0} + 4*A_{3,0} - A_{4,0}) >> 6$$

$$c_{0,0} = (A_{-2,0} - 5*A_{-1,0} + 17*A_{0,0} + 58*A_{1,0} - 10*A_{2,0} + 4*A_{3,0} - A_{4,0}) >> 6$$

La parte fraccionaria del componente horizontal del vector de movimiento determina qué muestra se deriva. Por ejemplo si la parte fraccionaria es $\frac{1}{4}$, se deriva $a_{0,0}$.

Cada una de las operaciones de interpolación incluye aplicar un filtro FIR de 8 derivaciones a las muestras de número entero más cercanas y truncando el resultado filtrado para desviar la ganancia de filtro. Una suma de los coeficientes de cada filtro se establece igual a 64 ($=2^6$). Por tanto, la operación de truncamiento incluye dividir el resultado filtrado entre 64 o desplazar 6 hacia la derecha el resultado filtrado. Obsérvese que, en las operaciones de interpolación anteriores, los resultados truncados se redondean hacia menos infinito. Añadiendo una desviación a los resultados filtrados, los resultados truncados pueden redondearse hacia el número entero más cercano. Sin embargo, redondear hacia menos infinito es ventajoso porque no requiere sumar la desviación a los resultados filtrados. Opcionalmente, pueden proporcionarse diferentes tipos de filtros de los cuales se aplica selectivamente un filtro, dependiendo de un desplazamiento fraccionario especificado por el componente horizontal de un vector de movimiento. Esta etapa puede suprimirse en el caso en el que el desplazamiento fraccionario horizontal sea cero. Obsérvese también que existe una operación de desplazamiento de 6 bits hacia la derecha en la etapa 606 y una operación de desplazamiento de 6 bits hacia la izquierda en la etapa 604 en el ejemplo con entradas de 8 bits. Por tanto, las etapas 604 y 606 pueden combinarse de tal manera que se retiran las operaciones de desplazamiento hacia la izquierda y de desplazamiento

hacia la derecha sin perturbar la salida.

En la etapa 608, la muestra fraccionaria $d_{0,0}$, $h_{0,0}$, $n_{0,0}$, $e_{0,0}$, $i_{0,0}$, $p_{0,0}$, $f_{0,0}$, $j_{0,0}$, $q_{0,0}$, $g_{0,0}$, $k_{0,0}$ o $r_{0,0}$ se deriva mediante las siguientes ecuaciones de interpolación:

5

$$\begin{aligned}d_{0,0} &= (-A_{0,-3} + 4*A_{0,-2} - 10*A_{0,-1} + 58*A_{0,0} + 17*A_{0,1} - 5*A_{0,2} + A_{0,3}) >> 6 \\h_{0,0} &= (-A_{0,-3} + 4*A_{0,-2} - 11*A_{0,-1} + 40*A_{0,0} + 40*A_{0,1} - 11*A_{0,2} + 4*A_{0,3} - A_{0,4}) >> 6 \\n_{0,0} &= (A_{0,-2} - 5*A_{0,-1} + 17*A_{0,0} + 58*A_{0,1} - 10*A_{0,2} + 4*A_{0,3} - A_{0,4}) >> 6 \\e_{0,0} &= (-a_{0,-3} + 4*a_{0,-2} - 10*a_{0,-1} + 58*a_{0,0} + 17*a_{0,1} - 5*a_{0,2} + a_{0,3}) >> 6 \\i_{0,0} &= (-a_{0,-3} + 4*a_{0,-2} - 11*a_{0,-1} + 40*a_{0,0} + 40*a_{0,1} - 11*a_{0,2} + 4*a_{0,3} - a_{0,4}) >> 6 \\p_{0,0} &= (a_{0,-2} - 5*a_{0,-1} + 17*a_{0,0} + 58*a_{0,1} - 10*a_{0,2} + 4*a_{0,3} - a_{0,4}) >> 6 \\f_{0,0} &= (-b_{0,-3} + 4*b_{0,-2} - 10*b_{0,-1} + 58*b_{0,0} + 17*b_{0,1} - 5*b_{0,2} + b_{0,3}) >> 6 \\j_{0,0} &= (-b_{0,-3} + 4*b_{0,-2} - 11*b_{0,-1} + 40*b_{0,0} + 40*b_{0,1} - 11*b_{0,2} + 4*b_{0,3} - b_{0,4}) >> 6 \\q_{0,0} &= (b_{0,-2} - 5*b_{0,-1} + 17*b_{0,0} + 58*b_{0,1} - 10*b_{0,2} + 4*b_{0,3} - b_{0,4}) >> 6 \\g_{0,0} &= (-c_{0,-3} + 4*c_{0,-2} - 10*c_{0,-1} + 58*c_{0,0} + 17*c_{0,1} - 5*c_{0,2} + c_{0,3}) >> 6 \\k_{0,0} &= (-c_{0,-3} + 4*c_{0,-2} - 11*c_{0,-1} + 40*c_{0,0} + 40*c_{0,1} - 11*c_{0,2} + 4*c_{0,3} - c_{0,4}) >> 6 \\r_{0,0} &= (c_{0,-2} - 5*c_{0,-1} + 17*c_{0,0} + 58*c_{0,1} - 10*c_{0,2} + 4*c_{0,3} - c_{0,4}) >> 6\end{aligned}$$

Las partes fraccionarias de los componentes horizontal y vertical del vector de movimiento determinan qué muestra se deriva. Por ejemplo si la parte fraccionaria horizontal es $\frac{1}{4}$ y la parte fraccionaria vertical es $\frac{3}{4}$, se deriva $p_{0,0}$.

10

Las operaciones de interpolación anteriores incluyen aplicar filtros FIR de 8 derivaciones de manera vertical con respecto a las muestras $A_{0,i}$, $a_{0,i}$, $b_{0,i}$ y $c_{0,i}$, donde $i = -3..4$, y truncar los resultados filtrados con un factor de 6. Obsérvese que, en la presente invención, se truncan todos los resultados filtrados y no se usa ningún resultado filtrado como valor intermedio sin truncan para calcular muestras fraccionarias a partir del mismo. Por tanto, en la presente invención, la precisión de los valores de muestras es constante a lo largo de la totalidad de la interpolación de muestras fraccionaria. Ni la etapa 606 ni la etapa 608 perturban la precisión en la representación de las muestras.

15

En la realización anterior de interpolación de muestras fraccionaria, los conjuntos de filtros usados en las operaciones de filtrado vertical y horizontal son todos filtros de 8 derivaciones. Obsérvese que pueden usarse filtros con diferentes números de derivaciones para la interpolación de movimiento fraccionaria. Por ejemplo, es deseable usar filtros con menos derivaciones para tamaños de bloque más pequeños. El uso de filtros con menos derivaciones reduce el número de muestras que van a recuperarse de la memoria de referencia, mediante lo cual puede reducirse el ancho de banda de memoria. El uso de filtros con menos derivaciones también reduce el número de multiplicaciones en las operaciones de filtrado y mejora la eficacia de compensación de movimiento. La norma H.264/AVC permite dividir un macrobloque o submacrobloque en una o más particiones con el propósito de compensación de movimiento. En la norma H.264/AVC, un macrobloque que tenga un tamaño de 16×16 puede dividirse en una o más particiones que tengan cada una un tamaño de 4×4 , 4×8 , 8×4 , 8×8 , 8×16 o 16×8 . Por consiguiente, pueden prepararse y aplicarse selectivamente filtros con un número diferente de derivaciones según la anchura de una partición. Por ejemplo, puede aplicarse un filtro de 8 derivaciones a particiones que tengan una anchura igual a o mayor de 8, y puede aplicarse un filtro de 4 derivaciones a particiones que tengan una anchura menor de 8. En una realización, tal filtro de 4 derivaciones tiene los coeficientes $[-4 \ 36 \ 36 \ -4]$. El borrador E603 del JCT-VC ha introducido el concepto de una unidad de codificación (CU). Una CU que tenga un tamaño de $N \times N$ puede dividirse en una o más particiones que tengan un tamaño de $N \times N$, $N \times N/2$ o $N/2 \times N$. Pueden aplicarse selectivamente filtros con diferentes números de derivaciones tanto si la altura o anchura de una CU es N como si es $N/2$.

35

En la realización anterior de compensación de movimiento de muestra fraccionaria, la interpolación de muestras horizontal fraccionaria (etapa 606) precede a la interpolación de muestras vertical fraccionaria (etapa 608). Obsérvese que puede intercambiarse el orden siempre y cuando se acuerde con antelación el orden entre un codificador y un decodificador. En otra realización, puede indicarse el orden de un codificador a un decodificador. Cuando las muestras de referencia tengan una anchura de 8 bits, el resultado del cálculo será el mismo independientemente del orden de interpolación horizontal y vertical. Si es el caso, no se necesita ningún acuerdo previo entre un codificador y un

40

decodificador acerca del orden.

En la compensación de movimiento comentada en el Borrador E603 del JCT-VC, pueden derivarse las mismas muestras fraccionarias tanto si se derivan los valores intermedios verticalmente y se aplican horizontalmente como si se derivan horizontalmente y se aplican verticalmente. Esto es beneficioso en el sentido de que no es necesario ningún acuerdo previo entre un codificador y un decodificador acerca de si los valores intermedios deben derivarse verticalmente y aplicarse horizontalmente o derivarse horizontalmente y aplicarse verticalmente. Sin embargo, los valores intermedios tienen mayor precisión que en la presente invención y el coste de aplicar un segundo filtro FIR a estos valores intermedios es mayor que el de la presente invención. En general, el coste aumenta en proporción a la precisión. Este coste puede ser significativamente más alto en el software en el que la precisión supera un tamaño de palabra habitual tal como 16 bits. En tal caso la precisión se extiende adicionalmente para coincidir con el siguiente tamaño de palabra habitual. Por ejemplo, los valores intermedios con 17 bits de precisión se extenderán hasta 32 bits. La precisión más alta de valores intermedios tal como se comenta en el borrador E603 del JCT-VC tiene un impacto en el coste significativo, sin proporcionar un beneficio significativo en cuanto a exactitud de la señal de predicción.

En la realización de la presente invención descrita anteriormente, es necesario fijar el orden de las interpolaciones horizontal y vertical entre un codificador y un decodificador. Sin embargo, fijar el orden de las interpolaciones horizontal y vertical no provocará ningún coste al codificador y al decodificador. Más bien, fijando el orden, la compensación de movimiento puede funcionar con una precisión inferior y, sin embargo, lo suficientemente alta, para reducir el coste de operaciones tales como multiplicaciones. Además, fijando el orden, la precisión de cálculo pasa a ser constante a lo largo de la totalidad de la interpolación de muestras fraccionaria. Por tanto, el cálculo de filtrado puede simplificarse dado que todas las multiplicaciones entre muestras y coeficientes pueden realizarse dentro de intervalos de bits fijos, por ejemplo, un intervalo de 16 bits con signo para muestras y un intervalo de 8 bits con signo para coeficientes de filtro. Puesto que todas las multiplicaciones se realizan dentro de intervalos de bits fijos, las operaciones de filtrado pueden implementarse eficazmente en paralelo en arquitecturas SIMD.

Volviendo a la figura 6, se aplica una transformación a las muestras derivadas en la etapa 610. Se describe un ejemplo de tal transformación como un proceso de predicción de muestra ponderada en la sección 8.4.2.3 y en sus subsecciones de la norma H.264/AVC. El proceso de predicción de muestra ponderada combina dos conjuntos de muestras mediante una interpolación lineal. La transformación puede mejorar la eficacia de codificación para escenas que contengan efectos de desvanecimiento. El valor de las muestras combinadas que resulta de la transformación puede recortarse para encajar dentro de un intervalo específico, por ejemplo un intervalo de 16 bits con signo. Obsérvese que puede suprimirse la etapa 610 cuando no se necesite ninguna transformación de este tipo.

Entonces se comprueba en la etapa 612 si es necesario generar otra predicción para compensar el movimiento. Por ejemplo, cuando vaya a realizarse un cálculo de promedio bidireccional, es necesario generar dos predicciones para calcular el promedio de las mismas. Si se determina que es necesaria otra predicción, el proceso regresa a la etapa 602. En la etapa 614, se calcula el promedio de dos predicciones para generar una predicción bidireccional. Si no se ordena un cálculo de promedio de manera bidireccional, se omite la etapa 614.

En la etapa 616, se reduce a escala una predicción generada y se recorta hasta un intervalo válido. La operación de reducción a escala realizada en la etapa 616 deshace la operación de ampliación a escala realizada en la etapa 604 y desplaza hacia la derecha la predicción generada el mismo número de bits que el usado para desplazar hacia la izquierda las muestras en la etapa 604. Es preferible sumar una desviación de redondeo antes de la operación de desplazamiento hacia la derecha para redondear el valor reducido a escala hacia el número entero más cercano. Si se resta la desviación de una muestra en la etapa 604, se suma la misma desviación a la predicción. El intervalo válido al que se recorta la predicción se define normalmente como $\{0...255\}$, cuando las muestras de referencia almacenadas tienen una anchura de 8 bits. En general, el intervalo válido puede definirse como $\{0...2^M-1\}$ para muestras de referencia que tengan una anchura de M bits.

En una implementación práctica de la presente invención, no es necesario realizar independientemente las etapas tal como se muestra en la figura 6, y algunas etapas pueden combinarse. Por ejemplo, pueden combinarse las operaciones de desplazamiento realizadas independientemente en las etapas de la figura 6. Un ejemplo es que pueden combinarse la operación de desplazamiento hacia la derecha incluida en el cálculo de promedio bidireccional realizado en la etapa 614 y la operación de desplazamiento hacia la derecha incluida en la reducción a escala realizada en la etapa 616. Además, pueden implementarse múltiples funciones o circuitos para circunstancias en las que pueden simplificarse los procesos, por ejemplo, en las que no hay ningún desplazamiento fraccionario y por tanto no se necesita filtrado FIR.

La figura 8 muestra los módulos funcionales del decodificador de vídeo 301, que se indica de manera general con el número de referencia 800 en la figura 8. Estos módulos funcionales se realizan por el procesador 101 que ejecuta el software 103 en la memoria 102. En la figura 8, una unidad de decodificación por entropía 810 obtiene un flujo de bits BS emitido por el aparato de codificación de imágenes en movimiento 400 y realiza el procesamiento de decodificación por entropía del flujo de bits. La unidad de decodificación por entropía 810 emite entonces los coeficientes de transformada cuantificados QC a la unidad de transformada inversa/cuantificación inversa 820. La unidad de decodificación por entropía 810 también emite el vector de movimiento MV a una unidad de predicción/compensación

de movimiento 860. La unidad de transformada inversa/cuantificación inversa 820 realiza el procesamiento de decodificación de cuantificación inversa y transformada de DCT inversa en los coeficientes de transformada cuantificados QC. La unidad de transformada inversa/cuantificación inversa 820 emite entonces la señal residual RS.

5 Un sumador 830 suma la señal residual RS a la señal de predicción PS obtenida de la unidad de compensación de movimiento 860 y emite una imagen decodificada a un filtro de desbloqueo 840. El filtro de desbloqueo 840 elimina artefactos de bloque de la imagen decodificada y la almacena en una memoria de referencia 850. Se recupera la imagen decodificada de la memoria 850 y se emite como señal de salida OS.

10 La unidad de predicción/compensación de movimiento 860 recupera muestras en una imagen de referencia almacenada en la memoria de referencia 850 y realiza la interpolación de muestras fraccionaria tal como se muestra en la figura 7. Esta unidad de predicción/compensación de movimiento 860 genera una predicción basándose en el vector de movimiento MV y en la imagen de referencia en la que se ha realizado la interpolación de muestras fraccionaria. De esta manera, la unidad de predicción/compensación de movimiento 860 emite la señal de predicción PS al sumador 830.

15 La figura 9 es un diagrama de flujo que muestra un alto nivel de proceso de decodificación realizado por el decodificador de vídeo 800. En la etapa 902, la unidad de predicción/compensación de movimiento 860 genera un bloque de predicción. En la etapa 904, la unidad de transformada inversa/cuantificación inversa 820 decodifica los coeficientes de transformada cuantificados QC y emite la señal residual RS. En la etapa 906, el sumador 830 suma la señal residual RS a la señal de predicción PS.

20 Aunque muchas alteraciones y modificaciones de la presente invención resultarán, sin duda, evidentes para un experto habitual en la técnica tras la lectura de la descripción anterior, debe entenderse que no se pretende en modo alguno que ninguna realización particular mostrada y descrita a modo ilustrativo se considere limitativa. Por tanto, no se pretende que las referencias a detalles de diversas realizaciones limiten el alcance de las reivindicaciones, que mencionan en ellas mismas sólo aquellas características que se consideran esenciales para la invención.

25 A continuación se describirán realizaciones adicionales preferidas:

30 La realización 1 es un método de decodificación de vídeo para compensación de movimiento realizada con una predicción inter-tramas, comprendiendo el método etapas ejecutables mediante ordenador ejecutadas mediante un procesador de un decodificador de vídeo para implementar: recuperar muestras de una memoria de imágenes de referencia; aplicar una primera interpolación de muestras fraccionaria a las muestras recuperadas en una primera
35 dirección para generar un primer conjunto de muestras fraccionarias; y aplicar una segunda interpolación de muestras fraccionaria al primer conjunto de muestras fraccionarias en una segunda dirección para generar un segundo conjunto de muestras fraccionarias, en el que el primer conjunto de muestras fraccionarias se representa con una precisión independiente de una precisión de representación de las muestras recuperadas de la memoria de imágenes de referencia. Las muestras de referencia se representan con una primera profundidad de bits y se convierten desde la
40 primera profundidad de bits hasta una segunda profundidad de bits. La segunda profundidad de bits se establece igual a un número de bits disponibles para representar las muestras fraccionarias. Las muestras fraccionarias del primer y el segundo conjunto de muestras fraccionarias se representan con la segunda profundidad de bits. Se genera una muestra de predicción bidireccional calculando el promedio de dos de los segundos conjuntos de muestras fraccionarias generados, siendo los dos segundos conjuntos diferentes entre sí. Se recorta un valor de la muestra de
45 predicción bidireccional para representarse con la primera profundidad de bits de las muestras de referencia.

La realización 2 es el método según la realización 1, en el que la interpolación de muestras fraccionaria aplica un filtro FIR de 8 derivaciones que tiene uno de los tres conjuntos siguientes de coeficientes para generar una muestra de un cuarto de píxel:

50 [-1, 4, -10, 58, 17, -5, 1, 0];
[-1, 4, -11, 40, 40, -11, 4, -1]; y
55 [0, 1, -5, 17, 58, -10, 4, -1].

La realización 3 es el método según la realización 2, en el que la interpolación de muestras fraccionaria comprende truncar cada uno de los resultados filtrados de modo que las muestras fraccionarias tengan una precisión deseada.

60 La realización 4 es el método según la realización 3, en el que la precisión deseada es constante a lo largo del primer y el segundo procesos de interpolación fraccionaria.

La realización 5 es el método según la realización 3, en el que truncar cada uno de los resultados filtrados comprende redondear resultados truncados hacia menos infinito.

65 La realización 6 es el método según la realización 1, que comprende además ajustar a escala las muestras

recuperadas hasta una segunda profundidad de bits que es constante para las muestras, con independencia de las profundidades de bits originales de las muestras, en el que la segunda profundidad de bits se establece igual a un número de bits disponibles para representar la muestra, y se establece un factor de ajuste a escala igual al número de bits disponibles para representar la muestra menos la profundidad de bits original de la muestra.

5 La realización 7 es el método según la realización 1, que comprende además restar una desviación de las muestras ajustadas a escala.

10 La realización 8 y las realizaciones 9-14 derivadas adicionalmente a continuación no son realizaciones de la invención, ya que están fuera del alcance de las reivindicaciones adjuntas y se proporcionan en el presente documento con propósitos ilustrativos solamente. La realización 8 es un método de codificación de vídeo para compensación de movimiento realizada con una predicción inter-tramas, comprendiendo el método etapas ejecutables mediante ordenador ejecutadas mediante un procesador de un codificador de vídeo para implementar:

15 recuperar muestras de una memoria de imágenes de referencia; aplicar una primera interpolación de muestras fraccionaria a las muestras recuperadas en una primera dirección para generar un primer conjunto de muestras fraccionarias; y aplicar una segunda interpolación de muestras fraccionaria al primer conjunto de muestras fraccionarias en una segunda dirección para generar un segundo conjunto de muestras fraccionarias, en el que el primer conjunto de muestras fraccionarias se representa con una precisión independiente de una precisión de representación de las muestras recuperadas de la memoria de imágenes de referencia. Las muestras de referencia se representan con una primera profundidad de bits y se convierten desde la primera profundidad de bits hasta una segunda profundidad de bits. La segunda profundidad de bits se establece igual a un número de bits disponibles para representar las muestras fraccionarias. Las muestras fraccionarias del primer y el segundo conjunto de muestras fraccionarias se representan con la segunda profundidad de bits. Se genera una muestra de predicción bidireccional calculando el promedio de dos de los segundos conjuntos de muestras fraccionarias generados, siendo los dos segundos conjuntos diferentes entre sí. Se recorta un valor de la muestra de predicción bidireccional para representarse con la primera profundidad de bits de las muestras de referencia.

30 La realización 9 es el método según la realización 8, en el que la interpolación de muestras fraccionaria aplica un filtro FIR de 8 derivaciones que tiene uno de los tres conjuntos siguientes de coeficientes para generar una muestra de un cuarto de píxel:

[-1, 4, -10, 58, 17, -5, 1, 0];

35 [-1, 4, -11, 40, 40, -11, 4, -1]; y

[0, 1, -5, 17, 58, -10, 4, -1].

40 La realización 10 es el método según la realización 9, en el que la interpolación de muestras fraccionaria comprende truncar cada uno de los resultados filtrados de modo que las muestras fraccionarias tengan una precisión deseada.

La realización 11 es el método según la realización 10, en el que la precisión deseada es constante a lo largo del primer y el segundo procesos de interpolación fraccionaria.

45 La realización 12 es el método según la realización 10, en el que truncar cada uno de los resultados filtrados comprende redondear resultados truncados hacia menos infinito.

50 La realización 13 es el método según la realización 8, que comprende además ajustar a escala las muestras recuperadas hasta una segunda profundidad de bits que es constante para las muestras con independencia de sus profundidades de bits originales, en el que la profundidad de bits unificada se establece igual a un número de bits disponibles para representar la muestra, y se establece un factor de ajuste a escala igual al número de bits disponibles para representar la muestra menos la profundidad de bits original de la muestra.

55 La realización 14 es el método según la realización 13, que comprende además restar una desviación de las muestras ajustadas a escala.

REIVINDICACIONES

1. Método de decodificación de vídeo para compensación de movimiento realizada con una predicción inter-tramas, comprendiendo el método etapas ejecutables mediante ordenador ejecutadas mediante un procesador (101) de un decodificador de vídeo (301) para implementar:
5 recuperar muestras de referencia de una memoria de imágenes de referencia (440), en el que las muestras de referencia se representan con una primera profundidad de bits;
10 aplicar (604, 606) una primera interpolación de muestras fraccionaria basándose en las muestras de referencia en una primera dirección para generar un primer conjunto de muestras fraccionarias, en el que la primera interpolación de muestras fraccionaria comprende convertir las muestras de referencia desde la primera profundidad de bits hasta una segunda profundidad de bits; y
15 aplicar (608) una segunda interpolación de muestras fraccionaria al primer conjunto de muestras fraccionarias en una segunda dirección para generar un segundo conjunto de muestras fraccionarias,
en el que las muestras fraccionarias del primer conjunto de muestras fraccionarias se representan con la segunda profundidad de bits que es independiente de la primera profundidad de bits de las muestras de
20 referencia,
en el que las muestras fraccionarias del segundo conjunto de muestras fraccionarias se representan con la segunda profundidad de bits,
25 en el que la segunda profundidad de bits se establece igual a un número de bits disponibles para representar las muestras fraccionarias,
en el que se genera una muestra de predicción bidireccional calculando el promedio de dos segundos conjuntos de muestras fraccionarias generados mediante la segunda interpolación de muestras fraccionaria,
30 siendo los dos segundos conjuntos diferentes entre sí,
en el que se recorta un valor de la muestra de predicción bidireccional para representarse con la primera profundidad de bits de las muestras de referencia.
35

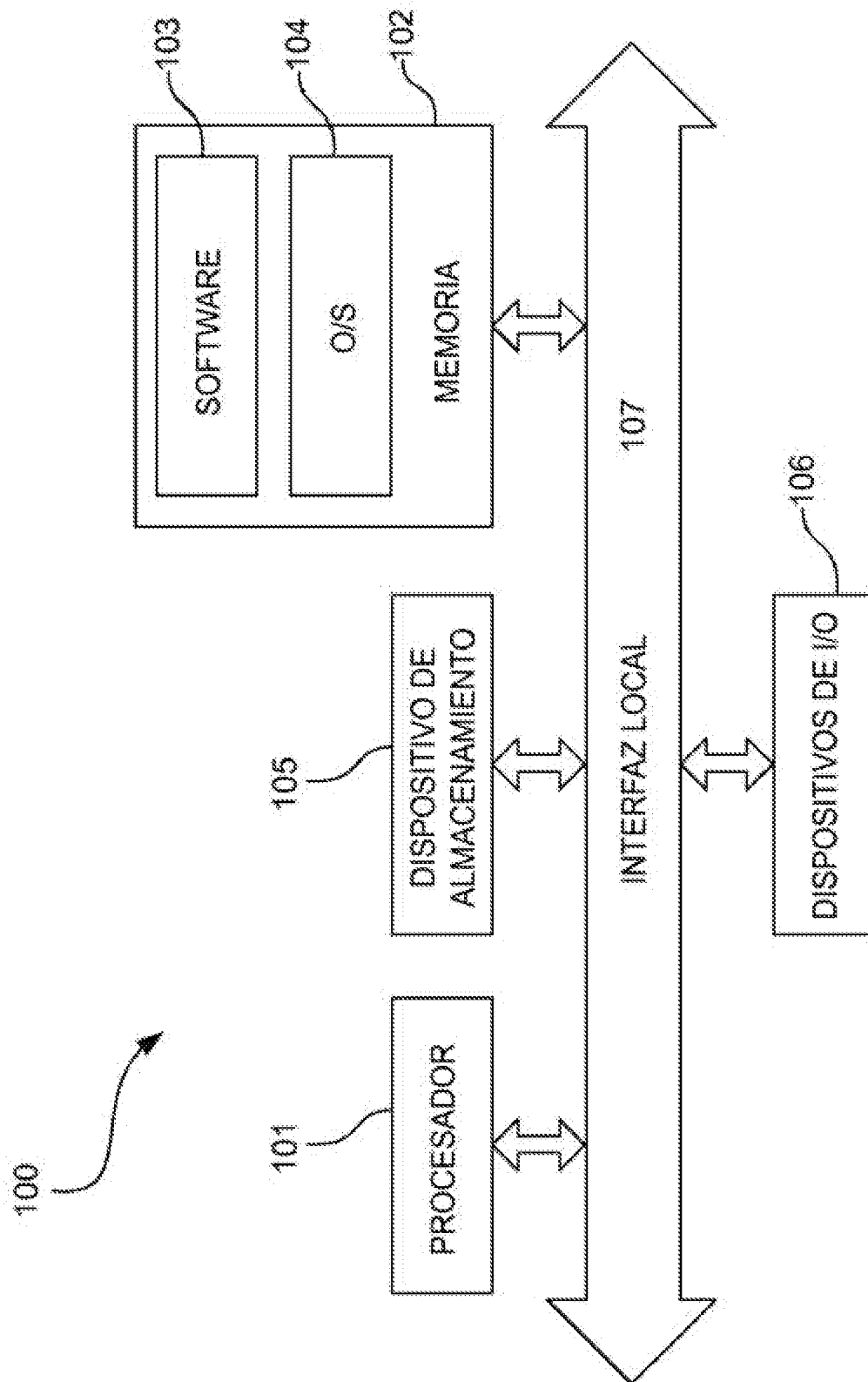


Fig. 1

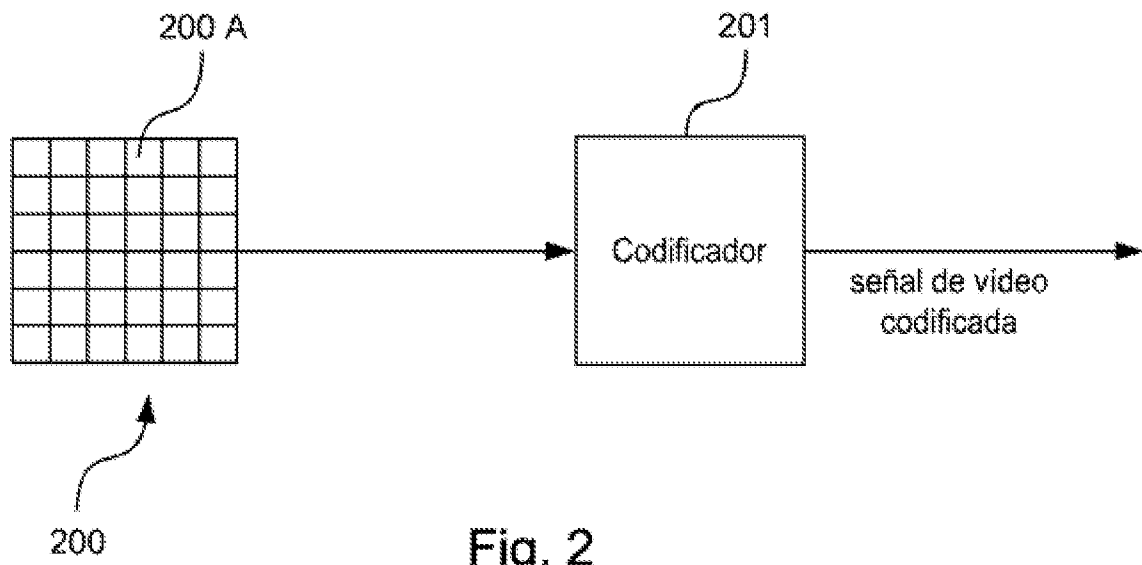


Fig. 2

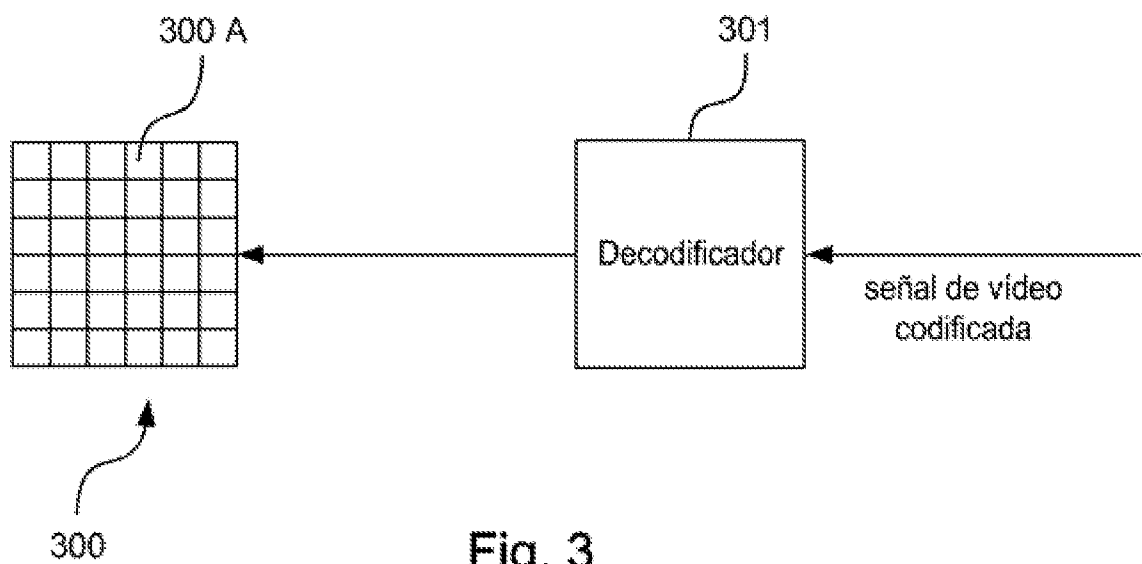


Fig. 3

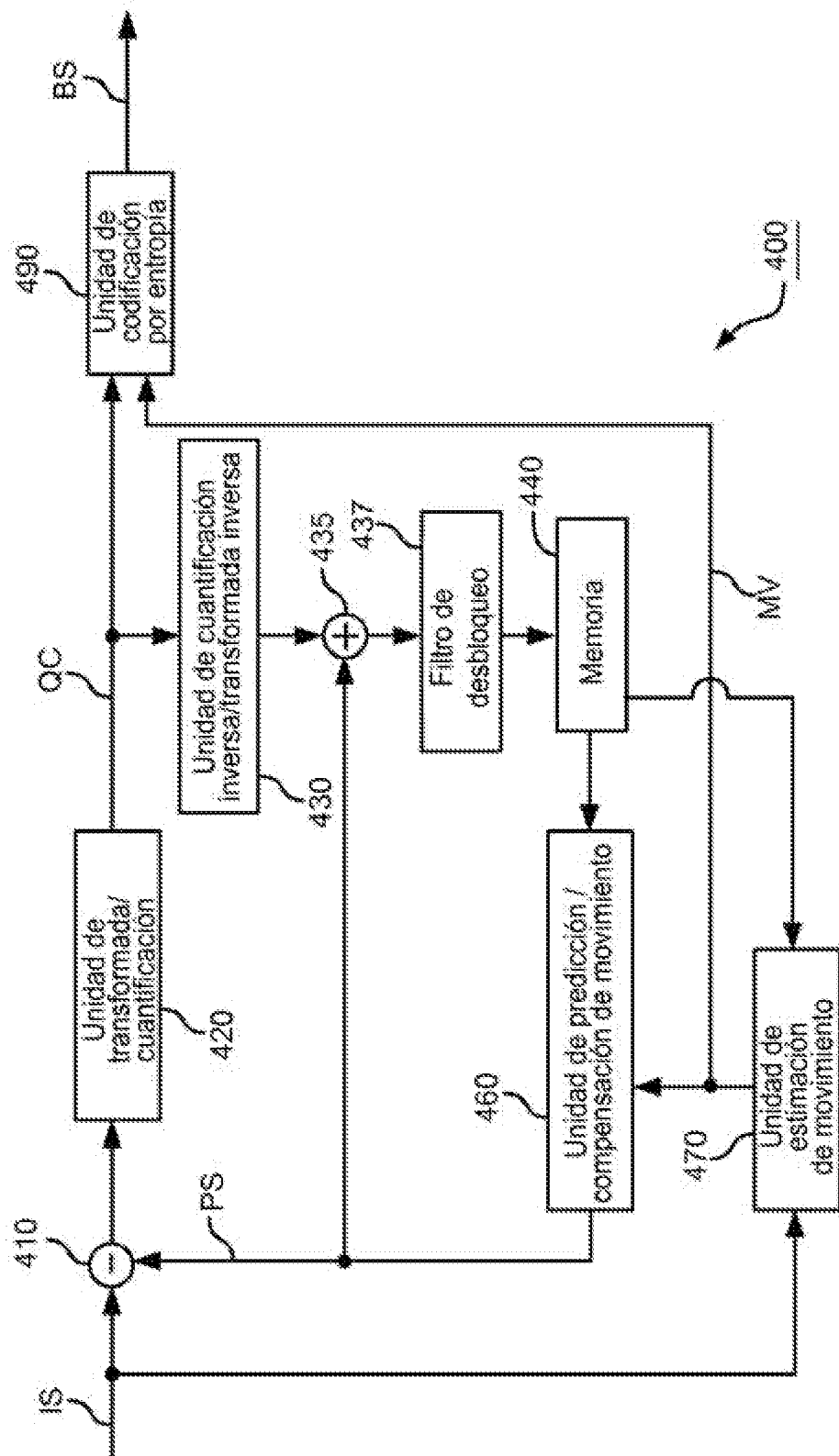


FIG. 4

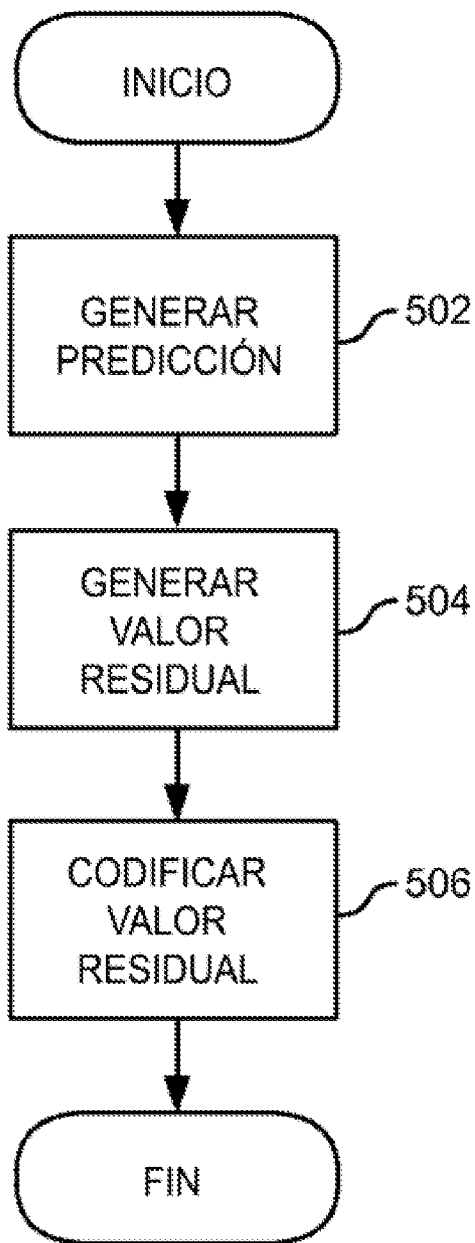


Fig. 5

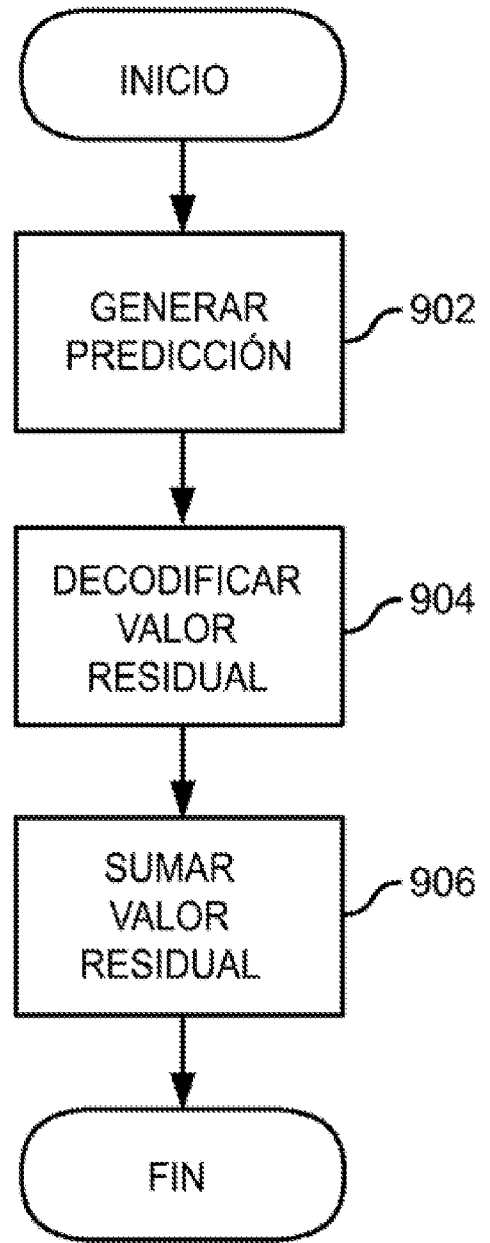


Fig. 9

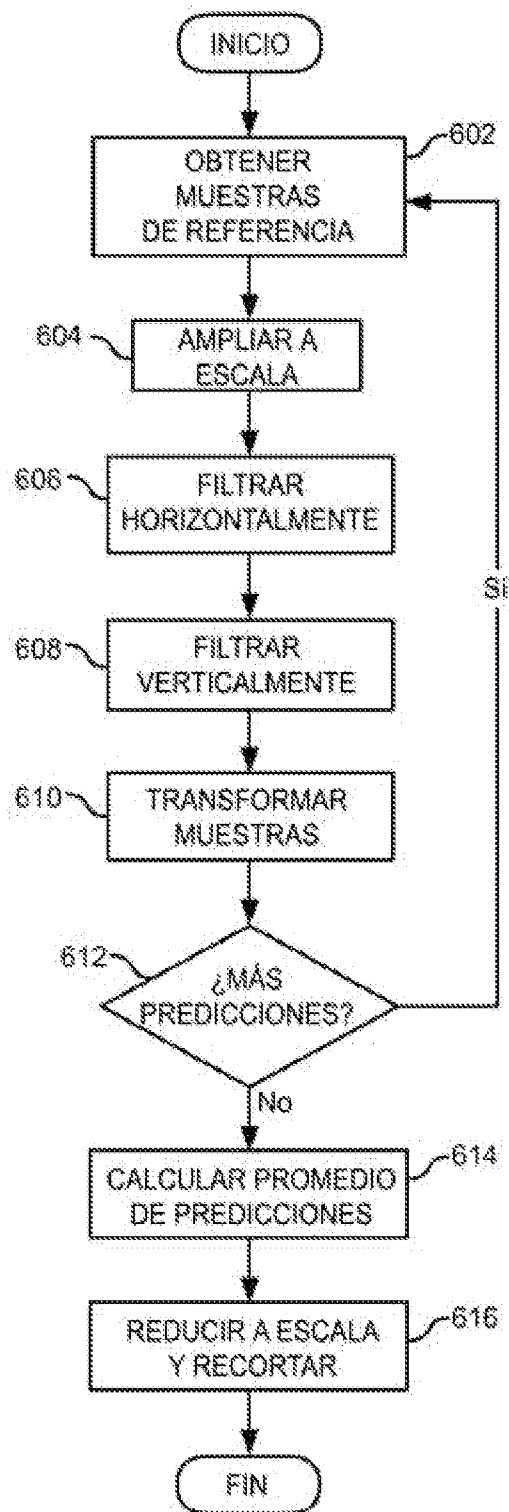


Fig. 6

$A_{-1,-1}$				$A_{0,-1}$	$a_{0,-1}$	$b_{0,-1}$	$c_{0,-1}$	$A_{1,-1}$				$A_{2,-1}$
$A_{-1,0}$				$A_{0,0}$	$a_{0,0}$	$b_{0,0}$	$c_{0,0}$	$A_{1,0}$				$A_{2,0}$
$d_{-1,0}$				$d_{0,0}$	$e_{0,0}$	$f_{0,0}$	$g_{0,0}$	$d_{1,0}$				$d_{2,0}$
$h_{-1,0}$				$h_{0,0}$	$i_{0,0}$	$j_{0,0}$	$k_{0,0}$	$h_{1,0}$				$h_{2,0}$
$n_{-1,0}$				$n_{0,0}$	$p_{0,0}$	$q_{0,0}$	$r_{0,0}$	$n_{1,0}$				$n_{2,0}$
$A_{-1,1}$				$A_{0,1}$	$a_{0,1}$	$b_{0,1}$	$c_{0,1}$	$A_{1,1}$				$A_{2,1}$
$A_{-1,2}$				$A_{0,2}$	$a_{0,2}$	$b_{0,2}$	$c_{0,2}$	$A_{1,2}$				$A_{2,2}$

Fig. 7

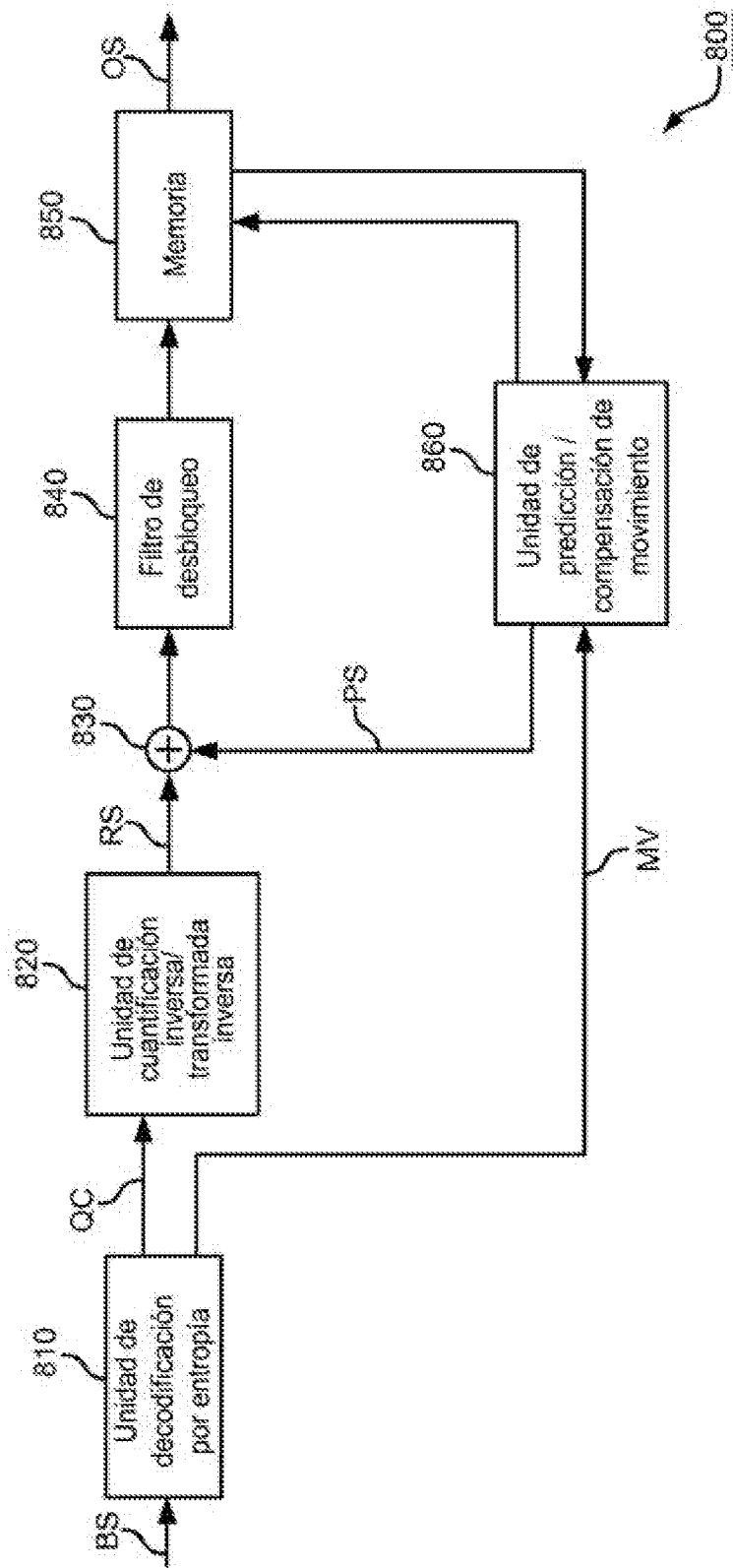


Fig. 8