(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 728 097**

(51) Int. CI.:

H04N 19/593 (2014.01)
H04N 19/105 (2014.01)
H04N 19/176 (2014.01)
H04N 19/186 (2014.01)
G06T 9/00 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(96) Fecha de presentación y número de la solicitud europea: **24.05.2006 E 15201105 (2)**

(97) Fecha y número de publicación de la concesión europea: **27.02.2019 EP 3023940**

(54) Título: **Procesamiento de imágenes basado en peso**

(30) Prioridad:

27.05.2005 SE 0501260
01.07.2005 WO PCT/SE2005/001070

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
22.10.2019

(73) Titular/es:

TELEFONAKTIEBOLAGET LM ERICSSON (PUBL) (100.0%)
164 83 Stockholm, SE

(72) Inventor/es:

STRÖM, JACOB

(74) Agente/Representante:

ELZABURU, S.L.P

DESCRIPCIÓN

Procesamiento de imágenes basado en peso

Campo técnico

La presente invención se refiere en general al procesamiento de imágenes, y en particular a métodos y sistemas para codificar y descodificar imágenes.

Antecedentes

La presentación y representación (en inglés, "rendering") de imágenes y gráficos en sistemas de procesamiento de datos y terminales de usuario tales como ordenadores, y en particular en terminales móviles, han aumentado enormemente en los últimos años. Por ejemplo, los gráficos e imágenes tridimensionales (3D) tienen una serie de atractivas aplicaciones en este tipo de terminales, entre ellas juegos, mapas en 3D y mensajería, protectores de pantalla e interfaces hombre-máquina.

Un proceso de representación de gráficos en 3D comprende típicamente tres sub-etapas. En pocas palabras, una primera etapa, la etapa de aplicación, crea varios triángulos. En una segunda etapa, la etapa de la geometría, se transforman las esquinas de estos triángulos, se proyectan y se iluminan. En una tercera etapa, la etapa de barrido (en inglés, "rastering"), se pueden "pegar" en los triángulos imágenes, denominadas frecuentemente texturas, que aumentan el realismo de la imagen representada. La tercera etapa también realiza normalmente una clasificación utilizando una memoria intermedia z (en inglés, "z-buffer").

Sin embargo, la representación de imágenes y texturas, y en particular de imágenes en 3D y gráficos, es una tarea computacionalmente cara en términos de ancho de banda de memoria y potencia de procesamiento requeridos para los sistemas gráficos. Por ejemplo, las texturas son costosas tanto en términos de memoria, puesto que se requiere ubicar o almacenar las texturas en memoria "en el chip", que es más rápida, como en términos de ancho de banda de memoria, porque puede ser necesario acceder a una textura varias veces para dibujar un solo píxel.

Para reducir los requisitos de ancho de banda y de potencia de procesamiento, típicamente se emplea un método o sistema de codificación de imagen (textura). Este sistema de codificación debe permitir un uso más eficaz de la costosa memoria en el chip y un menor ancho de banda de memoria durante la representación y, por lo tanto, un menor consumo de energía y/o una representación más rápida.

Delp y Mitchell [1] desarrollaron un esquema simple, denominado codificación de truncamiento de bloques (BTC, por sus siglas en inglés) para la compresión de imágenes. Aunque sus aplicaciones no fueron en sí la compresión de texturas, algunos de los otros sistemas descritos en esta sección se basan en sus ideas. Su esquema comprimía imágenes en escala de grises, considerando un bloque de 4x4 píxeles cada vez. Para un bloque de este tipo, se almacenaban dos valores de escala de grises de 8 bits, y cada pixel del bloque utilizaba después un solo bit para indicar una de estas escalas de grises. Esto daba lugar a 2 bits por pixel (bpp).

Campbell *et al.* [2] presentaron un desarrollo sencillo de la BTC, denominado compresión de células de color (CCC). En lugar de utilizar un valor de escala de grises de 8 bits, se emplea el valor de 8 bits como un índice en una paleta de colores. Esto permite comprimir texturas de color a 2 bpp. Sin embargo, es necesaria una búsqueda de memoria en la paleta, y el tamaño de la paleta está limitado.

El método S3TC de compresión de texturas, de Iourcha *et al.* [3] es probablemente el esquema más popular en la actualidad. Se utiliza en DirectX y también existen extensiones del mismo en OpenGL. Su funcionamiento se puede considerar un desarrollo adicional de la CCC. El tamaño del bloque para la S3TC es de 4x4 píxeles, que se comprimen a 64 bits. Se almacenan dos colores de base como 16 bits cada uno, y cada píxel almacena un índice de dos bits en una paleta local de colores que se compone de los dos colores de base y dos colores adicionales entre los colores de base. Esto significa que todos los colores se encuentran en una línea dentro del espacio RGB. La tasa de compresión de S3TC es 4 bpp. Una desventaja del S3TC es que solo se pueden utilizar cuatro colores por bloque.

Fenney [4] introduce un esquema radicalmente distinto que se utiliza en la plataforma MBX de *hardware* para gráficos en teléfonos móviles. Este esquema utiliza dos imágenes de baja resolución, y durante la descompresión se incrementa bilinealmente la escala de las mismas. Cada píxel también almacena un factor de mezcla entre estas dos imágenes de escala ampliada. Se describen la compresión a 4 bpp y a 2 bpp. Para cada bloque se utilizan 64 bits. La principal desventaja del esquema de Fenney, que hace que sea menos atractivo en implementaciones reales, es que la información de bloques de imagen vecinos es necesaria durante la descompresión, lo que complica gravemente la descompresión.

PACKMAN es un método de compresión de texturas desarrollado recientemente por Ström y Akenine-Möller [5], y más ampliamente divulgado en Ström y Akenine-Möller, PACKMAN: Texture Compression for Mobile Phones, In Sketches program at ACM SIGGRAPH (2004). Codifica un bloque de 2x4 téxeles (píxeles) a 32 bits. Solo se utiliza un color por bloque, pero en cada píxel se puede modificar la intensidad de este color. El principal objetivo de

PACKMAN era conseguir la mínima complejidad de descompresión. En PACKMAN la crominancia está muy cuantificada, lo que puede acarrear artefactos de bloque.

Para mejorar PACKMAN, Ström y Akenine-Möller han desarrollado un método mejorado de compresión denominado iPACKMAN/Ericsson Texture Compression (ETC) [6,7]. En iPACKMAN/ETC se codifican en común dos bloques 2x4 de imagen, lo que permite la codificación diferencial de los colores. Esto ha hecho posible disponer de una cuantificación más fina de los colores, lo que produce un aumento de la calidad en aproximadamente 3 dB. Por tanto, este método de compresión ha superado en términos de calidad a la S3TC y actualmente constituye el método/sistema de compresión de texturas de mayor calidad públicamente conocido.

Aún existe la necesidad de mejorar la compresión de imágenes y, en particular, en términos de compresión y descompresión de bloques de imagen problemáticos que presentan determinadas características de color que los esquemas de procesamiento de imágenes de la técnica anterior no pueden gestionar de manera eficaz con alta calidad. Estos bloques de imagen problemáticos incluyen bloques que presentan transiciones con variación lenta entre dos o más colores.

Compendio

La presente invención supera estos y otros inconvenientes de los arreglos de la técnica anterior.

Es un objeto general de la presente invención proporcionar un sistema de procesamiento de imágenes que pueda gestionar eficazmente bloques de imagen que presenten transiciones con variación lenta entre dos o más colores.

Este y otros objetos se logran mediante la invención tal como se define por las reivindicaciones adjuntas.

En pocas palabras, la presente invención implica un procesamiento de imágenes en forma de codificación (compresión) de una imagen y descodificación (descompresión) de una imagen codificada (comprimida).

Según la invención, se descompone una imagen a codificar en una serie de bloques de imagen que comprenden múltiples elementos de imagen (píxeles, téxeles o vóxeles). Un bloque de imagen comprende preferiblemente dieciséis elementos de imagen y tiene un tamaño de 2^mx2^n elementos de imagen, en donde m y n son preferiblemente 2. Cada elemento de imagen de un bloque se caracteriza por una propiedad de elemento de imagen, preferiblemente un color, por ejemplo, un color RGB (rojo, azul, verde, por sus siglas en inglés) de 24 bits. Posteriormente se codifican los bloques de imagen.

En esta codificación de bloques (con pérdida) se asignan pesos de color a al menos un subconjunto de los elementos de imagen del bloque de imagen. Después se determinan al menos dos claves de color que son representaciones de al menos dos valores de color, basándose al menos parcialmente en los pesos de color asignados. La representación codificada o comprimida generada del bloque de imagen comprende las al menos dos claves de color, que se pueden considerar como valores de color cuantificados. En consecuencia, los colores originales de los elementos de imagen del bloque de imagen serán representados por representaciones de color derivables de los al menos dos valores de color, obtenibles a su vez de las al menos dos claves de color. Además, las representaciones de color de los elementos de imagen en el al menos un subconjunto se pueden derivar de combinaciones de los al menos dos valores de color ponderados mediante los pesos de color asignados.

Esta manera de representar bloques de imagen gestiona de manera eficaz transiciones con variación suave de al menos dos colores dentro de un bloque de imagen, y transiciones y pendientes de color de este tipo que se extiendan a bloques vecinos. Tras leer la siguiente descripción de las realizaciones de la invención se apreciarán otras ventajas ofrecidas por la presente invención.

Durante la decodificación o descompresión, se determinan los al menos dos valores de color basándose en las al menos dos claves de color. Después se asignan pesos de color a un elemento de imagen a descodificar. Por último, se calcula una representación de color a utilizar para este elemento de imagen basándose en los pesos de color proporcionados y los al menos dos valores de color determinados.

La presente invención también enseña sistemas para codificar imágenes y bloques de imagen, sistemas para descodificar imágenes codificadas y bloques de imagen y terminales de usuario que albergan tales sistemas.

Breve descripción de los dibujos

La invención, junto con objetos y ventajas adicionales de la misma, se puede entender mejor haciendo referencia a la siguiente descripción tomada junto con los dibujos adjuntos, en los cuales:

la Figura 1 es un diagrama de flujo que ilustra un método para comprimir/codificar una imagen y un bloque de imagen según la presente invención;

la Figura 2 es una ilustración de un ejemplo de un bloque de imagen según la presente invención;

la Figura 3 es un dibujo que ilustra esquemáticamente la asignación de pesos de color según una realización de la

presente invención;

la Figura 4 es una ilustración de una representación comprimida de un bloque de imagen según una realización de la presente invención;

la Figura 5 es un dibujo que ilustra esquemáticamente la asignación de pesos de color según otra realización de la presente invención;

la Figura 6 es una ilustración de una representación comprimida de un bloque de imagen según otra realización de la presente invención;

la Figura 7 es un dibujo que ilustra esquemáticamente la asignación de pesos de color según una realización adicional de la presente invención;

la Figura 8 es un dibujo que ilustra esquemáticamente la asignación de pesos de color según otra realización más de la presente invención;

la Figura 9 es un diagrama de flujo que ilustra con más detalle una realización del paso determinante de la Figura 1;

la Figura 10 es un diagrama de flujo de pasos adicionales del método de codificación/compresión de imágenes de la Figura 1 según una implementación multimodal de la presente invención;

la Figura 11 es un diagrama de flujo que ilustra con más detalle realizaciones de pasos de compresión de la Figura 10;

la Figura 12A es un diagrama que ilustra la distribución de colores de elementos de imagen de un bloque de imagen que ventajosamente se puede comprimir según un modo de la implementación multimodal;

la Figura 12B es un diagrama que ilustra representaciones de color generadas según un modo de la implementación multimodal y adecuadas para representar los colores de los elementos de imagen ilustrados en la Figura 12A;

la Figura 13A es un diagrama que ilustra la distribución de colores de elementos de imagen de otro bloque de imagen que ventajosamente se puede comprimir según otro modo de la implementación multimodal;

la Figura 13B es un diagrama que ilustra representaciones de color generadas según otro modo de la implementación multimodal y adecuadas para representar los colores de los elementos de imagen ilustrados en la Figura 13A;

las Figuras 14 a 17 son ilustraciones de representaciones comprimidas de un bloque de imagen según la implementación multimodal;

la Figura 18 es un diagrama de flujo de un método para descodificar/descomprimir una imagen comprimida y el bloque de imagen según la presente invención;

la Figura 19 es un diagrama de flujo que ilustra pasos adicionales del método de descodificación/descompresión de la Figura 20 para una implementación multimodal;

la Figura 20 es un diagrama de flujo que ilustra con más detalle realizaciones del paso de descompresión de la Figura 19;

la Figura 21 es un diagrama de flujo que ilustra con más detalle otra realización del paso de descompresión de la Figura 19;

la Figura 22 ilustra esquemáticamente un ejemplo de un terminal de usuario con un codificador y descodificador de imágenes según la presente invención;

la Figura 23 es un diagrama de bloques que ilustra esquemáticamente una realización de un codificador de imágenes según la presente invención;

la Figura 24 es un diagrama de bloques que ilustra esquemáticamente una realización de un codificador de bloques según la presente invención;

la Figura 25 es un diagrama de bloques que ilustra esquemáticamente otra realización de un codificador de bloques según la presente invención;

la Figura 26 es un diagrama de bloques que ilustra esquemáticamente una realización de un descodificador de imágenes según la presente invención;

la Figura 27 es un diagrama de bloques que ilustra esquemáticamente una realización de un descodificador de bloques según la presente invención; y

la Figura 28 es un diagrama de bloques que ilustra esquemáticamente otra realización de un descodificador de bloques según la presente invención.

Descripción detallada

En los dibujos se utilizarán las mismas referencias para elementos correspondientes o similares.

La presente invención se refiere al procesamiento de imágenes y gráfico, y en particular a la codificación o compresión de imágenes y bloques de imagen y a la descodificación o descompresión de imágenes y bloques de imagen codificados (comprimidos).

En general, y según la invención, durante la codificación de imágenes se descompone o divide una imagen en una serie de bloques de imagen. Cada uno de tales bloques de imagen comprende entonces múltiples elementos de imagen que tienen, entre otras cosas, un color determinado. Se codifican o comprimen los bloques de imagen para generar una representación codificada/comprimida de la imagen.

Cuando posteriormente hay que representar, por ejemplo, mostrar en una pantalla, una imagen o gráfico primitivos codificados, se identifican y descodifican los elementos de imagen relevantes de los bloques de imagen codificados. Después se utilizan estos elementos de imagen descodificados para generar una representación descodificada de la imagen o gráficos originales primitivos.

La presente invención está bien adaptada para el uso con gráficos en tres dimensiones (3D), tales como juegos, mapas en 3D y escenas en 3D, mensajes, por ejemplo mensajes animados, protectores de pantalla, interfaces hombre-máquina (MMI, por sus siglas en inglés), etc., pero no se limita a esto. Así, también se podría emplear la invención para codificar otros tipos de imágenes o gráficos, por ejemplo imágenes en una dimensión (1D), en dos dimensiones (2D) o en 3D.

En el procesamiento de gráficos en 3D, típicamente se crean varios triángulos y se determinan las correspondientes coordenadas en pantalla de las esquinas de estos triángulos. A cada triángulo se le asigna ("se pega") una imagen (o parte de una imagen), o una denominada "textura". Sin embargo, la gestión de texturas es costosa para un sistema gráfico, tanto en términos de memoria utilizada para el almacenamiento de texturas como en términos de ancho de banda de memoria durante los accesos a memoria, cuando se recuperan texturas desde la memoria. Esto constituye un problema, en particular, para los clientes ligeros, por ejemplo las unidades y teléfonos móviles, que tienen una capacidad de memoria y ancho de banda limitados. En consecuencia, a menudo se emplea un esquema de codificación de texturas o de imágenes. En un esquema de este tipo, típicamente se descompone o se divide una textura en una serie de bloques de imagen que comprenden múltiples téxeles. A continuación se codifican los bloques de imagen y se almacenan en una memoria. Téngase en cuenta que el tamaño de un bloque de imagen codificado (una versión codificada de un bloque de imagen) es más pequeño que el tamaño correspondiente de la versión no codificada del bloque de imagen.

En la presente invención, la expresión "elemento de imagen" se refiere a un elemento de un bloque de imagen o representación codificada de un bloque de imagen. Este bloque de imagen, a su vez, corresponde a una parte de una imagen o textura. Por tanto, según la invención un elemento de imagen podría ser un téxel (elemento de textura) de una textura (1D, 2D, 3D), un píxel de una imagen (1D o 2D) o un vóxel (elemento de volumen) de una imagen en 3D. En general, un elemento de imagen se caracteriza por determinadas propiedades de elemento de imagen, tales como un valor de color. Además, el término "imagen" se utiliza en lo que sigue para indicar cualquier imagen o textura en 1D, 2D o 3D que se pueda codificar y descodificar por medio de la presente invención, incluidos, pero sin imitación, fotos, texturas de tipo juego, texto, dibujos, etc.

La presente invención proporciona un procesamiento de imágenes que es particularmente adecuado para comprimir y descomprimir imágenes y bloques de imagen con transiciones de variación lenta entre al menos dos colores. En los esquemas de la técnica anterior, discutidos en la sección de antecedentes, mediante claves de color (S3TC) o una o varias claves de color y uno o varios modificadores de intensidad/color (PACKMAN e iPACKMAN/ETC) se forma en el espacio de color una paleta de colores que comprende típicamente cuatro valores de color. Así, cada elemento de imagen tiene un índice de color asociado con uno de los colores de la paleta de colores. Con una solución de este tipo, generalmente es difícil procesar elementos de imagen que presenten transiciones de color con variación lenta.

En claro contraste con estos esquemas de la técnica anterior, la presente invención asigna diferentes pesos de color a elementos de imagen de un bloque de imagen. Después se determinan los colores a utilizar para el bloque de imagen basándose, al menos en parte, en los pesos de color asignados. Esto significa que los colores originales de los elementos de imagen se representarán por representaciones de color derivables de combinaciones de los colores determinados, ponderadas mediante los pesos de color asignados. Potencialmente, esto permite la utilización de representaciones de color únicas, dependiendo de los pesos de color asignados, para cada elemento de imagen del bloque, lo que a su vez significa una paleta de colores mucho mayor. Además, los pesos de color se pueden configurar de manera que también se pueden representar con alta calidad de imagen bloques problemáticos que presenten transiciones de color con variación lenta.

Compresión

La Figura 1 ilustra un método (con pérdidas) para codificar una imagen según la presente invención. En un primer paso S1, se descompone o divide la imagen en una serie de bloques de imagen. Cada uno de tales bloques de imagen comprende entonces múltiples elementos de imagen. En una realización preferida de la invención, un bloque de imagen comprende dieciséis elementos de imagen (píxeles, téxeles o vóxeles) y tiene un tamaño de 2^mx2^n elementos de imagen, donde m=4-n y n=0, 1, 2, 3, 4. Más preferiblemente, m y n son ambos 2. También sería posible utilizar un bloque de imagen de tamaño 2^m x 2^n o 2^m x 2^n x 2^p elementos de imagen, donde m, n, p son cero o números enteros positivos con la condición de que no todos los m, n, p puedan ser cero simultáneamente. La Figura 2 ilustra esquemáticamente un ejemplo de un bloque 600 de imagen con dieciséis elementos 610 de imagen según la presente invención. En una realización alternativa de la presente invención, se descompone la imagen en una serie de sub-bloques de imagen, que preferiblemente tienen un tamaño de 2x4 o 4x2 elementos de imagen. En este caso, se podrían gestionar juntos dos de tales sub-bloques durante la compresión para formar un bloque 4x4 600 como se ilustra en la Figura 2. Volviendo a la Figura 1, en el paso S1 preferiblemente se descompone el bloque completo de imagen en bloques de imagen (no solapantes). Sin embargo, en algunas aplicaciones solamente se codifica una parte de una imagen y, por tanto, solamente se descompone en bloques de imagen esta parte.

Los pasos siguientes S2 y S4 realizan una codificación o compresión de los bloques de imagen. En primer lugar, en el paso S2, se asignan pesos de color a al menos un subconjunto de los elementos de imagen del bloque de imagen, lo que se ilustra esquemáticamente por la línea L1. Los pesos de color se determinan preferiblemente basándose en la posición relativa que los elementos de imagen del al menos un subconjunto tienen en el bloque de imagen. Estos pesos de color se utilizarán durante la descompresión para ponderar distintos colores que se determinen para el bloque de imagen, con el fin de generar representaciones de color utilizadas para representar los colores originales ("verdaderos") de los elementos de imagen. Por ejemplo, supóngase que se determinan dos colores (C_0 y C_1) para el bloque actual de imagen. Los pesos de color asignados en este paso S2 pueden ser entonces w_0^{xy} y w_1^{xy} para el elemento de imagen que tiene la posición (x,y) en el bloque de imagen. Durante la compresión, la representación de imagen de este elemento de imagen será $w_0^{xy}C_0 + w_1^{xy}C_1$, es decir, una combinación, en este caso una combinación lineal, ponderada de los dos colores.

Como es bien conocido en la técnica, un color comprende típicamente múltiples componentes de color, muy frecuentemente tres componentes de color, dependiendo del espacio de color propietario utilizado. Por ejemplo, los colores pueden ser de colores RGB (rojo, verde, azul), colores en el espacio YUV o el espacio YCrCb, o cualquier otro espacio de color propietario utilizado en el procesamiento y gestión de imágenes y gráficos. En ese caso, los múltiples pesos de color asignados en el paso S2 podrían considerarse como un vector de pesos de color

$\mathbf{w}^{xy} = \begin{bmatrix} w_{R0}^{xy} & w_{G0}^{xy} & w_{B0}^{xy} \\ w_{R1}^{xy} & w_{G1}^{xy} & w_{B1}^{xy} \end{bmatrix}$. En este caso, los elementos de componente individual en un vector de pesos se podrían fijar de forma individual o al menos podrían ser iguales. En el caso en que $w_{R0}^{xy} = w_{B0}^{xy} = w_{G0}^{xy}$, el vector de pesos solo comprende dos pesos $\begin{bmatrix} w_0^{xy} \\ w_1^{xy} \end{bmatrix}$ por elemento de imagen en este ejemplo ilustrativo.

Los pesos de color se asignan preferiblemente para cada elemento de imagen en al menos un subconjunto de los elementos de imagen de los bloques, que se representa por la línea L1.

En una primera realización, el bloque de imagen comprende N elementos de imagen, N es un número entero mayor que uno, y el subconjunto comprende M elementos de imagen, en donde 0≤M<N. Esto significa que no se asignan pesos de color para el o los N-M elementos de imagen restantes. En ese caso, el color original de este o estos elementos restantes de imagen se representa por una de las claves de color a determinar para el bloque de imagen. Sin embargo, esto corresponde básicamente a fijar en 1 todos los elementos de componente de uno de los vectores de pesos de color y fijar en 0 todos los elementos de componente del otro u otros vectores de peso de color.

Por lo tanto, en otra implementación preferida de la presente invención, se asignan pesos de color a todos los elementos de imagen del bloque, básicamente mediante la repetición del paso S2 para cada elemento de imagen. En esta realización, al menos uno de los pesos de color asignados a al menos un elemento de imagen es preferiblemente distinto de 0, 1 y -1.

En un siguiente paso S3, se determinan al menos dos claves de color para el bloque de imagen basándose en, o utilizando, los pesos de color asignados. Estas al menos dos claves de color son representaciones de al menos dos valores de color. Como se ha indicado antes, los valores de color podrían ser colores RGB (rojo, verde, azul), colores en el espacio YUV o en el espacio YCrCb, o cualquier otro espacio de color propietario utilizado en el procesamiento y gestión de imágenes y gráficos.

Las claves de color están preferiblemente en el mismo formato (espacio) de color que la imagen. Sin embargo, en algunos casos puede ser útil convertir la imagen a un espacio de color distinto, es decir, tener las claves de color en un primer espacio de color y la imagen original en un segundo espacio de color distinto.

En un primer arreglo descrito en la presente memoria, en este paso S3 se determinan dos claves de color basándose en los pesos de color asignados. Sin embargo, en una implementación preferida de la presente invención se utilizan tres claves de color. Incluso a veces se determinan en lugar de ello cuatro o más claves de color basándose en los pesos de color. Estas múltiples claves representan entonces tres, cuatro o más valores de color. Conforme a la presente invención es posible, basando la determinación de claves de color en los pesos de color asignados, determinar claves de color que den lugar a una alta calidad de imagen y permitan la generación de representaciones de color con transiciones de color de variación lenta.

En un siguiente paso S4, los colores originales de los múltiples elementos de imagen del bloque se representan por representaciones de color derivables de los al menos dos valores de color, que a su vez se representan por las al menos dos claves de color determinadas en el paso S3. Además, las representaciones de color de los elementos de imagen en el al menos un subconjunto, es decir, aquellos elementos de imagen para los que se asignaron pesos de color en el paso S2, se pueden derivar de combinaciones de los al menos dos valores de color ponderados mediante los pesos de color asignados.

Si el al menos un subconjunto comprende un primer subconjunto del elemento de imagen del bloque, las representaciones de color de estos elementos de imagen se pueden derivar de combinaciones de los al menos dos valores de color ponderados mediante los pesos de color asignados. Sin embargo, las representaciones de color de elementos de imagen en un segundo subconjunto restante de los elementos de imagen del bloque se seleccionan directamente de los valores de color y por lo tanto no constituyen combinaciones de múltiples valores de color.

Preferiblemente, se repiten los pasos S2 a S4 para todos los bloques de imagen proporcionados durante la descomposición del paso S1 (lo que se ilustra esquemáticamente por la línea L2). El resultado es, entonces, una secuencia o archivo de bloques de imagen codificados. Se podrían ordenar los bloques de imagen codificados (representaciones codificadas de los bloques de imagen) en un archivo, de izquierda a derecha y de arriba a abajo en el mismo orden en el que se desglosaron en el paso S1 de descomposición en bloques. Después, el método termina.

Se podría enviar la imagen codificada a una memoria para almacenarse allí hasta una ulterior representación, por ejemplo, visualización, de la imagen. Además, se podría enviar a un transmisor la imagen codificada, en forma de una señal de representaciones de bloque codificadas, para transmitirla (de modo inalámbrico o por cable) a otra unidad.

La Figura 3 es una ilustración esquemática de un bloque 600 de imagen en una imagen o textura a comprimir según la presente invención. Aquí se han de determinar cuatro claves de color para el bloque 600 de imagen. Cada una de las cuatro claves de color representa un respectivo valor de color, cuyos componentes de rojo se denotan con R_0, R_1, R_2 y R_3. Esto permite una ampliación bilineal de escala en la que, sin embargo, se almacenan como una representación comprimida del bloque de imagen todos los colores necesarios para la ampliación bilineal de escala. Los elementos de imagen 610 que forman las esquinas del bloque 600 de imagen tienen pesos de color de solo unos y ceros. La Tabla 1 ilustra los pesos color asignados a los elementos 610 de imagen en el bloque según este arreglo.

Tabla 1 – pesos de color

Posición (x, y)	Color 0	Color 1	Color 2	Color 3
(0,0)	1	0	0	0
(1,0)	2/3	1/3	0	0
(2,0)	1/3	2/3	0	0
(3,0)	0	1	0	0
(0,1)	2/3	0	1/3	0
(1,1)	4/9	2/9	2/9	1/9
(2,1)	2/9	4/9	1/9	2/9
(3,1)	0	2/3	0	1/3
(0,2)	1/3	0	2/3	0
(1,2)	2/9	1/9	4/9	2/9
(2,2)	1/9	2/9	2/9	4/9

Posición (x, y)	Color 0	Color 1	Color 2	Color 3
(3,2)	0	1/3	0	2/3
(0,3)	0	0	1	0
(1,3)	0	0	2/3	1/3
(2,3)	0	0	1/3	2/3
(3,3)	0	0	0	1

Esto significa que las representaciones de color del bloque de imagen ilustrado en la Figura 3, y que tienen asignados pesos de color conforme a la Tabla 1 anterior, tendrán componentes de rojo conforme a la Tabla 2 a continuación.

Tabla 2 – combinaciones de color

R_0	$\frac{2}{3}R_0+\frac{1}{3}R_1$	$\frac{1}{3}R_0+\frac{2}{3}R_1$	R_1
$\frac{2}{3}R_0+\frac{1}{3}R_2$	$\frac{4}{9}R_0+\frac{2}{9}R_1+\frac{2}{9}R_2+\frac{1}{9}R_3$	$\frac{2}{9}R_0+\frac{4}{9}R_1+\frac{1}{9}R_2+\frac{2}{9}R_3$	$\frac{2}{3}R_1+\frac{1}{3}R_3$
$\frac{1}{3}R_0+\frac{2}{3}R_2$	$\frac{2}{9}R_0+\frac{1}{9}R_1+\frac{4}{9}R_2+\frac{2}{9}R_3$	$\frac{1}{9}R_0+\frac{2}{9}R_1+\frac{2}{9}R_2+\frac{4}{9}R_3$	$\frac{1}{3}R_1+\frac{2}{3}R_3$
R_2	$\frac{2}{3}R_2+\frac{1}{3}R_3$	$\frac{1}{3}R_2+\frac{2}{3}R_3$	R_3

Preferiblemente, los componentes de azul y verde se gestionan de la misma manera, es decir, básicamente intercambiando R_z por B_z o G_z, en donde z=0, 1, 2, 3.

Como puede verse en la Tabla 2, los componentes del color rojo (verde y azul) de doce de los elementos de imagen se pueden derivar como combinaciones lineales ponderadas de al menos los valores de color (R_0, R_1, R_2, R_3) representados por las cuatro claves de color. Sin embargo, los componentes del color rojo (verde y azul) de los cuatro elementos de imagen de esquina se derivan cada uno directamente de uno de los valores de color.

Cuando uno se mueve a lo largo de la primera fila (de R_0 a R_1), la tercera fila (de R_2 a R_3), la primera columna (de R_0 a R_2) y la tercera columna (de R_1 a R_3) de la Tabla 2, el componente del color rojo de las representaciones de color de los elementos de imagen en estas filas y columnas cambia monótonamente (salvo que los dos valores extremos sean iguales). Por ejemplo, si $R_1>R_0$, el valor del componente de rojo aumenta monótonamente a lo largo de la primera fila, es decir, cuando se va del elemento de imagen (0,0) al elemento de imagen (0,3). Análogamente, si $R_2<R_0$, el valor del componente de rojo disminuye monótonamente a lo largo de la primera columna (del elemento de imagen (0,0) al (3,0)). Si se utilizan los mismos pesos de color también para los componentes del color verde y/o el azul, también estos cambiarán monótonamente para estas filas y columnas. A una fila o columna se le denomina "matriz unidimensional de elementos de imagen". Esto significa que al menos un componente de color de las representaciones de color cambia monótonamente a lo largo de al menos una matriz unidimensional de elementos de imagen. Esto permite que transiciones de color suaves, y por lo tanto bloques de imagen que tengan tales distribuciones de color, se puedan representar con una elevada calidad de imagen.

La Figura 4 ilustra una representación comprimida 700 del bloque de imagen ilustrado en la Figura 3 que ha sido comprimido. La representación 700 (bloque de imagen codificado o comprimido) comprende una primera clave 710 de color, una segunda clave 720 de color, una tercera clave 730 de color y una cuarta clave 740 de color. Téngase en cuenta que el orden respectivo de las claves 710, 720, 730, 740 del bloque 700 de imagen codificado puede ser distinto del ilustrado en la Figura.

En este caso, los pesos de color asignados a los elementos de imagen del bloque están predefinidos y se utilizarán para todos los bloques de imagen de la imagen comprimida. Esto significa que, si se comprimen según este esquema todos los bloques de imagen de una imagen, se utilizarán para todos los bloques de imagen los pesos de color enumerados en la Tabla 1. Sin embargo, téngase en cuenta que las cuatro claves de color pueden ser distintas para bloques distintos, lo que origina en último término distintas representaciones de color para los bloques de imagen.

Sin embargo, se prevé que se pueda incluir, como parte de la representación 700 del bloque comprimido, información acerca de los pesos de color asignados a elementos de imagen del bloque de imagen. Por ejemplo,

supóngase que existen múltiples conjuntos de pesos que se pueden utilizar para los bloques de imagen. Cada uno de tales conjuntos de pesos comprende entonces pesos de color asignados a elementos de imagen de al menos un subconjunto de los elementos del bloque. Un primero de tales conjuntos de pesos puede incluir los pesos de color indicados en la Tabla 1 precedente. Entonces, un segundo conjunto puede tener diferentes pesos para al menos uno de estos elementos de imagen. En tal caso, la representación 700 de bloque comprimida comprende preferiblemente un índice de conjunto de pesos o clave de pesos que represente el conjunto de pesos de color utilizado para el bloque de imagen actual. Se puede utilizar un bit único (0_{bin} y 1_{bin}) como clave de pesos para discriminar entre dos conjuntos de pesos, mientras que se requieren dos o más bits si existen más de dos conjuntos. En una solución de este tipo, se pueden comprimir distintos bloques de imagen de una imagen utilizando distintas distribuciones de pesos de color.

Si se asignan 64 bits para el bloque 700 de imagen comprimido, se pueden utilizar 16 bits por cada clave 710, 720, 730, 740 de color y cada clave 710, 720, 730, 740 de color puede estar en forma de RGB565. Preferiblemente, se invierten más bits en los componentes de verde, ya que el componente de verde aporta una contribución desproporcionada a la percepción de la intensidad.

Sin embargo, el arreglo precedente se utiliza preferiblemente como un modo auxiliar al esquema iPACKMAN/ETC antes mencionado. En tal caso, solo están disponibles 57 bits para codificar las cuatro claves 710, 720, 730, 740 de color (los siete bits restantes se utilizarán como índice de modo para discriminar entre este modo auxiliar, iPACKMAN/ETC y otros modos auxiliares). Hay que codificar cuatro claves 710, 720, 730, 740 de color, con tres componentes cada una, utilizando solo 57 bits, y una posible solución podría ser utilizar un formato RGB554, lo que daría como resultado 56 bits en total. El bit restante se puede utilizar como clave de peso o bien para reforzar uno de los componentes de color de una de las claves 710, 720, 730, 740.

El arreglo presentado en las Figuras 3 y 4 y en la Tabla 1 tiene el inconveniente de que la resolución de color obtenible para las claves 710, 720, 730, 740 de color es bastante baja, especialmente cuando se emplean como un complemento a iPACKMAN/ETC y cada clave 710, 720, 730, 740 está en el formato RGB554. En este caso, en particular la baja resolución del componente de azul dará lugar a artefactos.

En cambio, en una implementación preferida de la presente invención se utiliza preferiblemente un plano, en lugar de la función bilineal, para aproximarse a la superficie de color y proporcionar los pesos de color. Esto se ilustra esquemáticamente en la Figura 5. En comparación con la Figura 3 y la forma de realización bilineal antes descrita, por cada bloque 600 de imagen se determinan solo tres claves de color. Esto significa que tres valores de color están representados por estas claves, y en la Figura 5 se han ilustrado los componentes de rojo (R_0, R_H, R_V) de estos valores de color.

Las representaciones de color y los pesos de color de los elementos 610 de imagen del bloque 600 de imagen se pueden calcular utilizando las siguientes fórmulas:

$$R(x,y) = \frac{x}{3}(R_H - R_0) + \frac{y}{3}(R_V - R_0) + R_0 \quad w_0^{xy} = 1 - \frac{x}{3} - \frac{y}{3}$$

$$G(x,y) = \frac{x}{3}(G_H - G_0) + \frac{y}{3}(G_V - G_0) + G_0 \quad w_H^{xy} = \frac{x}{3}$$

$$B(x,y) = \frac{x}{3}(B_H - B_0) + \frac{y}{3}(B_V - B_0) + B_0 \quad w_V^{xy} = \frac{y}{3}$$

Esto significa que las representaciones de color de los elementos 610 de imagen en las posiciones (0,0), (0,3) y (3,0) se pueden seleccionar directamente de los valores de color representados por las tres claves, lo que da como resultado (R_0, G_0, B_0) para el elemento de imagen (0,0), (R_H, G_H, B_H) para el elemento de imagen (3,0) y (R_V, G_V, B_V) para el elemento de imagen (0,3) en este ejemplo ilustrativo. Esto corresponde a los siguientes pesos de color para estos elementos de imagen: w^{00} = [1 0 0], w^{30} = [0 1 0] y w^{03} = [0 0 1].

Los pesos de color asignados a los bloques de imagen comprimidos según esta realización de la presente invención se distribuyen entre los elementos de imagen según la Tabla 3 a continuación.

Tabla 3 - pesos de color

Posición (x, y)	Color 0	Color h	Color V
(0,0)	1	0	0
(1,0)	2/3	1/3	0
(2,0)	1/3	2/3	0
(3,0)	0	1	0

Posición (x, y)	Color 0	Color h	Color V
(0,1)	2/3	0	1/3
(1,1)	1/3	1/3	1/3
(2,1)	0	2/3	1/3
(3,1)	-1/3	1	1/3
(0,2)	1/3	0	2/3
(1,2)	0	1/3	2/3
(2,2)	-1/3	2/3	2/3
(3,2)	-2/3	1	2/3
(0,3)	0	0	1
(1,3)	-1/3	1/3	1
(2,3)	-2/3	2/3	1
(3,3)	-1	1	1

Esto significa que las representaciones de color del bloque de imagen ilustrado en la Figura 5 y que tiene asignados pesos de color conforme a la Tabla 3 precedente tendrán componentes de rojo según la Tabla 4 a continuación.

Tabla 4 - representaciones de color

R_0	$\frac{2}{3}R_0+\frac{1}{3}R_H$	$\frac{1}{3}R_0+\frac{2}{3}R_H$	R_H
$\frac{2}{3}R_0+\frac{1}{3}R_V$	$\frac{1}{3}R_0+\frac{1}{3}R_H+\frac{1}{3}R_V$	$\frac{2}{3}R_H+\frac{1}{3}R_V$	$-\frac{1}{3}R_0+R_H+\frac{1}{3}R_V$
$\frac{1}{3}R_0+\frac{2}{3}R_V$	$\frac{1}{3}R_H+\frac{2}{3}R_V$	$-\frac{1}{3}R_0+\frac{2}{3}R_H+\frac{2}{3}R_V$	$-\frac{2}{3}R_0+R_H+\frac{2}{3}R_V$
R_V	$-\frac{1}{3}R_0+\frac{1}{3}R_H+R_V$	$-\frac{2}{3}R_0+\frac{2}{3}R_H+R_V$	$-R_0+R_V+R_H$

Preferiblemente, los componentes de azul y verde se gestionan de la misma manera, es decir, básicamente intercambiando R_Z por B_Z o G_Z, donde z=0, 1, 2, 3.

También en este caso los componentes de color aumentan o disminuyen monótonamente cuando uno se mueve a lo largo de la primera fila y la primera columna de la Tabla 4.

La Figura 6 ilustra una representación comprimida 700 del bloque de imagen ilustrado en la Figura 5 que ha sido comprimido según esta realización de la presente invención. La representación 700 (bloque de imagen codificado o comprimido) comprende una primera clave 710 de color, una segunda clave 720 de color y una tercera clave 730 de color. Téngase en cuenta que el orden respectivo de las claves 710, 720, 730 del bloque 700 de imagen codificado puede ser distinto del ilustrado en la Figura.

Las claves 710, 720, 730 de color se pueden representar ahora mediante la cuantificación de color RGB676, dando como resultado un tamaño total de 57 bits para las tres claves. Esto significa que esta realización se puede utilizar ventajosamente como complemento y modo auxiliar a iPACKMAN/ETC como se ha descrito más arriba. RGB676 permite una resolución mayor en comparación con la realización anterior (RGB554). Como alternativa, si se utiliza la presente invención como un esquema autónomo, las claves 710, 720, 730 de color se pueden representar como RGB777 y el bit restante se puede utilizar, por ejemplo, como clave de peso o bien para reforzar un componente de color de una de las claves 710, 720, 730.

En lugar de emplear tres claves 710, 720, 730 de color que permiten el cálculo de un valor de color directamente a partir de una clave 710, 720, 730, se pueden utilizar las denominadas claves diferenciales. En ese caso, se puede determinar la primera clave 710 de color como se ha descrito más arriba, es decir, comprendiendo tres componentes de color cuantificados, por ejemplo en forma RGB777. El primer valor de color (R_0, G_0, B_0) se puede obtener entonces directamente a partir de esta primera clave 710 mediante la expansión de los componentes de color

cuantificados. Las otras dos claves 720, 730 codifican, en cambio, una distancia en el espacio de color desde este primer valor de color. Así, las dos claves 720, 730 podrían representar $dR_HdG_HdB_H666$ y $dR_VdG_VdB_V666$, donde cada uno de los componentes dW_Z, W=R, G, B y z=H, V, representan un número en el intervalo [-31, 32]. Los otros dos valores de color se obtienen después según las siguientes fórmulas:

$$R_{H/V}=R_0 + dR_{H/V}$$

$$G_{H/V}=G_0 + dG_{H/V}$$

$$B_{H/V}=B_0 + dB_{H/V}$$

Las realizaciones descritas en relación con las Figuras 3 y 5 presentan, sin embargo, algunas desventajas. En primer lugar, varios de los pesos de color asignados implican la división por tres o potencias de tres ($9=3^2$), que es bastante caro de realizar cuando se implementa la descompresión en *hardware*. Otro inconveniente de estos patrones y asignaciones de pesos de color es que es difícil crear gradientes de pendiente constante. Por ejemplo, si los R_H, G_H, B_H de un bloque se han fijado en R_0, G_0, B_0 en el bloque anterior, habrá dos elementos de imagen con una misma representación de color uno a continuación de otro, interrumpiendo la pendiente. Por lo tanto, aunque se puede representar con alta calidad una transición suave de color dentro de un bloque dado, es más difícil codificar una transición suave de color de este tipo a lo largo de bloques de imagen vecinos.

La Figura 7 ilustra una realización de la presente invención en donde la asignación de pesos de color se lleva a cabo de manera que resuelve los dos inconvenientes antes mencionados. En esta realización, los pesos de color se seleccionan preferiblemente de manera que solo un único elemento 610 de imagen del bloque 600 tiene pesos de color constituidos por un 1 y el resto 0. Esto significa que este único elemento 610 de imagen, preferiblemente un elemento de imagen de esquina, tiene su color original representado por un valor de color derivado de solo una de las tres claves de color.

La forma de colocar los valores de color que se ilustra en la Figura 7 da lugar a las siguientes fórmulas para calcular las representaciones de color y pesos de color de los elementos 610 de imagen en el bloque 600 de imagen:

$$R(x,y)=\frac{x}{4}(R_H-R_0)+\frac{y}{4}(R_V-R_0)+R_0 \quad w_0^{xy}=1-\frac{x}{4}-\frac{y}{4}$$

$$G(x,y)=\frac{x}{4}(G_H-G_0)+\frac{y}{4}(G_V-G_0)+G_0 \quad w_H^{xy}=\frac{x}{4}$$

$$B(x,y)=\frac{x}{4}(B_H-B_0)+\frac{y}{4}(B_V-B_0)+B_0 \quad w_V^{xy}=\frac{y}{4}$$

Los pesos de color asignados a bloques de imagen comprimidos según esta realización de la presente invención se distribuyen entre los elementos de imagen conforme a la Tabla 5 a continuación.

Tabla 5 - pesos de color

Posición (x, y)	Color 0	Color h	Color V
(0,0)	1	0	0
(1,0)	3/4	1/4	0
(2,0)	1/2	1/2	0
(3,0)	1/4	3/4	0
(0,1)	3/4	0	1/4
(1,1)	1/2	1/4	1/4
(2,1)	1/4	1/2	1/4
(3,1)	0	3/4	1/4
(0,2)	1/2	0	1/2
(1,2)	1/4	1/4	1/2
(2,2)	0	1/2	1/2

Posición (x, y)	Color 0	Color h	Color V
(3,2)	-1/4	3/4	1/2
(0,3)	1/4	0	3/4
(1,3)	0	1/4	3/4
(2,3)	-1/4	1/2	3/4
(3,3)	-1/2	3/4	3/4

Esto significa que las representaciones de color del bloque de imagen ilustrado en la Figura 7, y que tiene asignados pesos de color conforme a la Tabla 5 precedente, tendrán componentes de rojo conforme a la Tabla 6 a continuación.

Tabla 6 - representaciones de color

R_0	$\frac{3}{4}R_0+\frac{1}{4}R_H$	$\frac{1}{2}R_0+\frac{1}{2}R_H$	$\frac{1}{4}R_0+\frac{3}{4}R_H$
$\frac{3}{4}R_0+\frac{1}{4}R_V$	$\frac{1}{2}R_0+\frac{1}{4}R_H+\frac{1}{4}R_V$	$\frac{1}{4}R_0+\frac{1}{2}R_H+\frac{1}{4}R_V$	$\frac{3}{4}R_H+\frac{1}{4}R_V$
$\frac{1}{2}R_0+\frac{1}{2}R_V$	$\frac{1}{4}R_0+\frac{1}{4}R_H+\frac{1}{2}R_V$	$\frac{1}{2}R_H+\frac{1}{2}R_V$	$-\frac{1}{4}R_0+\frac{3}{4}R_H+\frac{1}{2}R_V$
$\frac{1}{4}R_0+\frac{3}{4}R_V$	$\frac{1}{4}R_H+\frac{3}{4}R_V$	$-\frac{1}{4}R_0+\frac{1}{2}R_H+\frac{3}{4}R_V$	$-\frac{1}{2}R_0+\frac{3}{4}R_V+\frac{3}{4}R_H$

Preferiblemente, los componentes de azul y verde se gestionan de la misma manera, es decir, básicamente intercambiando R_Z por B_Z o G_Z, donde z=0, 1, 2, 3.

También en este caso, los componentes de color aumentan o disminuyen monótonamente cuando uno se mueve a lo largo de la primera fila y la primera columna de la Tabla 6.

La representación de bloque comprimida para esta realización se ilustra en la Figura 6, y la discusión con respecto a esa Figura también se aplica a esta realización.

En esta realización, los pesos de color implican la división por dos o cuatro, cuya implementación en *hardware* es trivial. Además, la realización permite la continuidad de la pendiente de color (transiciones de color) más allá de los límites de bloque. Esto es posible fijando la R_H (o R_V) de un bloque en la R_0 del bloque precedente, lo que se traduce en una pendiente perfecta. Una ventaja adicional de esta realización reside en que la precisión de las representaciones de color que se pueden derivar de las claves de color y los pesos de color aumenta aún más, ya que el color puede variar en escalones de 1/4 en lugar de 1/3, con cada elemento de imagen.

Se pueden obtener realizaciones adicionales basadas en el mismo tema que la Figura 7 rotando las posiciones de R_0, R_H y R_V un cuarto de vuelta, media vuelta o tres cuartos de vuelta, respectivamente.

En general, en un contexto más amplio los pesos de color asignados a los elementos de imagen según la presente invención se pueden seleccionar de manera que los valores de color representados por las claves de color podrían verse como ubicados todos dentro del bloque de imagen (véanse las Figuras 3 y 5), ubicados algunos dentro del bloque de imagen y ubicados algunos fuera del bloque (véase la Figura 7), o ubicados todos fuera del bloque. La posición relativa de los valores de color en relación con el bloque define la ecuación (plana) utilizada para obtener los pesos de color para los elementos de imagen. En una realización muy preferida de la presente invención, no todos los valores de color están ubicados en una misma línea (columna/fila) del bloque.

La presente invención prevé que se puedan seleccionar los pesos de color para representar posiciones de los valores de color en el bloque 600 de manera que los valores de color o puntos no estén ubicados en el centro de elementos 610 de imagen, lo que se ilustra esquemáticamente en la Figura 8.

Las fórmulas para el plano serán entonces:

$$R(x,y)=2x(R_H-R_0)-2y(R_V-R_0)+R_0-3R_H+3R_V$$

$$G(x,y) = 2x(G_H - G_0) - 2y(G_V - G_0) + G_0 - 3G_H + 3G_V$$

$$B(x,y) = 2x(B_H - B_0) - 2y(B_V - B_0) + B_0 - 3B_H + 3B_V$$

y los pesos de color están representados por las ecuaciones:

$$w_0^{xy} = 1 - 2x + 2y$$

$$w_H^{xy} = -3 + 2x$$

$$w_V^{xy} = 3 - 2y$$

Los pesos de color asignados a bloques de imagen comprimidos según esta realización de la presente invención se distribuyen entre los elementos de imagen conforme a la Tabla 7 a continuación.

Tabla 7 - pesos de color

Posición (x, y)	Color 0	Color H	Color V
(0,0)	1	-3	3
(1,0)	-1	-1	3
(2,0)	-3	1	3
(3,0)	-5	3	3
(0,1)	3	-3	1
(1,1)	1	-1	1
(2,1)	-1	1	1
(3,1)	-3	3	1
(0,2)	5	-3	-1
(1,2)	3	-1	-1
(2,2)	1	1	-1
(3,2)	-1	3	-1
(0,3)	7	-3	-3
(1,3)	5	-1	-3
(2,3)	3	1	-3
(3,3)	1	3	-3

Por lo tanto, la presente invención implica asignar pesos de color a algunos o a todos los elementos de imagen del bloque y, después, determinar claves de color basándose en los pesos, en donde las representaciones de color de los elementos de imagen se pueden derivar de combinaciones ponderadas de valores de color representados por las claves, utilizando los pesos asignados a los elementos de imagen. Por supuesto, este concepto se puede extender a distinto número de múltiples claves de color y distintas asignaciones de pesos de color. En una implementación preferida, al menos un peso de al menos un elemento de imagen es distinto de 0, 1 y -1. Por lo tanto, el empleo de tres claves de color significa que se pueden asignar pesos de color de manera que sea posible una interpolación plana de valores de color. Si, en cambio, se emplean cuatro o cinco claves para un bloque de imagen, se pueden utilizar la interpolación bilineal y la gaussiana, respectivamente. En la implementación preferida, se fijan los pesos de color para reflejar que al menos uno de los valores de color representados por claves de color se puede considerar ubicado fuera del bloque de imagen. Esto permite una buena aproximación de un color que varía muy lentamente - más lentamente que si se considerasen todos los valores de color como lugares dentro del bloque (compárense las Figuras 5 y 7).

La Figura 9 es un diagrama de flujo que ilustra con más detalle una realización del paso de determinación de la Figura 1. El método continúa desde el paso S2 de la Figura 1. En el paso S10 se seleccionan claves de color candidatas que son representaciones de valores de color candidatas. Estas claves candidatas se pueden

seleccionar al azar, o bien pueden ser las menores o mayores claves de color candidatas posibles, por ejemplo una secuencia de 19 0_{bin} (la menor clave posible, que representa el valor de color (0, 0, 0)), o bien una secuencia de 19 1_{bin} (la mayor clave posible, que representa el valor de color (255, 255, 255)). Los tres componentes de color R, G, B se pueden determinar por separado, es decir, básicamente ejecutando tres procesos paralelos o subsiguientes. Por tanto, la discusión que sigue se limita a solamente un componente de color.

En un siguiente paso S11, utilizando los pesos de color asignados se calculan los componentes de representación del color rojo para los elementos de imagen del bloque obtenido con esta selección de componentes de claves de color candidatas. Esto equivale a reemplazar R_0, R_H y R_V (R_0, R_1, R_2, R_3 y R_4) en los bloques de imagen discutidos más arriba con los componentes de rojo del valor de color candidato. El error de representar el componente de rojo de los elementos de imagen con estos componentes de representación de color candidatos se calcula después en el paso S11.

$$\varepsilon^2 = \sum_{y=0}^{3}\sum_{x=0}^{3}\left(\begin{bmatrix} w_0^{xy} & w_H^{xy} & w_V^{xy} \end{bmatrix}\begin{bmatrix} R_0^c \\ R_H^c \\ R_V^c \end{bmatrix} - \overline{R}^{xy}\right)^2$$

En la fórmula de error precedente, R_0^c, R_H^c, R_V^c representan el componente de rojo de tres valores de color candidatos representados por los tres componentes de clave de color candidatos seleccionados y $\overline{R}^{xy}$ es el color rojo original del elemento de imagen en la posición (x, y) dentro del bloque. En la fórmula anterior, los pesos de color y el color original son específicos para el elemento de imagen, mientras que los valores de color candidatos son los mismos, dentro de una selección dada de componentes de clave candidatos, para todos los elementos de imagen del bloque. Después, en el paso S12 se almacena en una memoria de errores el valor de error calculado, junto con una nota acerca de las claves de color candidatas seleccionadas.

Se repiten los pasos S10 a S12 para diferentes selecciones de componentes de claves de color candidatos y, si el valor de error estimado es menor que el valor de error almacenado en la memoria de errores, el nuevo valor de error reemplaza al valor almacenado y los componentes de clave candidatos utilizados para el cálculo de este nuevo valor de error más pequeño reemplazan igualmente a los componentes de clave candidatos previamente almacenados.

Se realiza este procedimiento para todos los posibles 18 bits de los componentes de rojo candidatos, lo que origina 2^{18} pasos. Además, se realiza el procedimiento para los componentes del color verde y el azul, lo que implica en total $2^{18}+2^{19}+2^{18}$ operaciones. En el paso S13 se seleccionan los respectivos componentes de rojo/verde/azul de claves que originan los valores de error más pequeños, y se utilizan como claves de color según la presente invención. El método continúa entonces al paso S4 de la Figura 1.

En otra realización, se adopta una aproximación de mínimos cuadrados en lugar de una búsqueda exhaustiva. Esto se puede representar en forma de matriz según lo siguiente:

$$\begin{bmatrix} \overline{R}^{00} \\ \overline{R}^{10} \\ \\ \overline{R}^{33} \end{bmatrix} = \begin{bmatrix} w_0^{00} & w_H^{00} & w_V^{00} \\ w_0^{10} & w_H^{10} & w_V^{10} \\ & & \\ w_0^{33} & w_H^{33} & w_V^{33} \end{bmatrix}\begin{bmatrix} R_0 \\ R_H \\ R_V \end{bmatrix}$$

Esto se puede escribir también de la forma $\overline{y} = A\overline{x}$, donde $\overline{y}$ es un vector que comprende los 16 componentes de rojo originales de los elementos de imagen del bloque, A es una matriz que comprende los 3x16=48 pesos de color asignados a los elementos de imagen y $\overline{x}$ es un vector que comprende los componentes de rojo a determinar y cuantificar en componentes de rojo de las claves de color. Por lo tanto, se debe determinar el vector $\overline{x}$. Esto se puede realizar según la siguiente ecuación:

$$\overline{x} = \left(A^T A\right)^{-1} A^T \overline{y}$$

Se lleva a cabo también el mismo procedimiento para los componentes del color verde y el azul.

La presente invención prevé que se pueden utilizar otras técnicas, además de la búsqueda exhaustiva y los mínimos cuadrados, para determinar las al menos dos claves por bloque de imagen según la presente invención.

Como se ha discutido en lo que antecede, la presente invención se utiliza ventajosamente como un complemento o modo auxiliar al esquema de compresión iPACKMAN/ETC. En tal caso, la presente invención se utilizará para comprimir y descomprimir bloques de imagen que tengan transiciones de color de variación lenta y, en particular, cuando exista un gradiente de color que se extiende a lo largo de varios bloques vecinos. Para otros bloques de imagen, se puede utilizar en su lugar iPACKMAN/ETC u otro modo auxiliar.

La Figura 10 ilustra esquemáticamente una implementación de este tipo. El método continúa desde el paso S1 de la Figura 1. Después se procesa el bloque de imagen a comprimir proporcionado, conforme a diferentes esquemas, típicamente en paralelo. En otras palabras, en el paso S20 se comprime el bloque de imagen según un primer modo de compresión para generar una primera representación de bloque comprimida. En un siguiente paso S24 se calcula un valor de error representativo de representar el bloque de imagen con este primer bloque comprimido. Además, se comprime adicionalmente el mismo bloque de imagen según al menos otro modo de compresión, en la Figura otros tres modos. Por tanto, en S21, S22 y S23 se utilizan un segundo, tercer y cuarto modos de compresión para comprimir el bloque con el fin de generar respectivamente una segunda, tercera y cuarta representaciones de bloque comprimidas. En los pasos S25, S26 y S27 se estiman valores de error representativos de representar el bloque con la segunda, tercera o cuarta representación de bloque comprimidas. Ahora están disponibles cuatro (o, en realizaciones alternativas, dos, tres o más de cuatro) representaciones comprimidas distintas. En el siguiente paso S28 se selecciona una de estas cuatro representaciones comprimidas y se utiliza como versión comprimida del bloque actual. Este paso de selección se realiza basándose en los valores de error calculados en los pasos S24-S27. Así, en el paso S28 se seleccionará la representación comprimida asociada con el menor valor de error. En un siguiente paso S29, se proporciona un representante de índice de modo del modo de compresión utilizado para generar la representación comprimida seleccionada, y se incluye en el bloque de imagen comprimido, es decir, en la secuencia de bits que representa el bloque de imagen. Preferiblemente, se realiza este procedimiento para cada bloque de imagen que ha de comprimirse. Después, el método termina.

Esto significa que cada bloque de imagen de una imagen se analiza y se comprime preferiblemente de forma individual, lo que en la mayoría de las aplicaciones típicas (dependiendo de la imagen real a codificar), da como resultado un mosaico de bloques de imagen comprimidos según los diferentes modos. Es decir, se comprime según el primer modo un primer conjunto de bloques de imagen, se comprime según el segundo modo un segundo conjunto de bloques, se comprime según el tercer modo un tercer conjunto de bloques y se comprime según el cuarto modo un cuarto conjunto de bloques. Para otras aplicaciones solo se utilizarán, para los distintos bloques de imagen, uno, dos o tres de los modos.

En esta realización, se genera un bloque candidato comprimido por cada modo de compresión disponible. Sin embargo, en otra realización la selección del modo de compresión a utilizar para un bloque dado se realiza antes de las compresiones reales del bloque. En ese caso, se realiza un primer paso de análisis en donde se investigan y se analizan los colores originales de los elementos de imagen del bloque y, en particular, su distribución en el espacio de color. La selección del modo de compresión se realiza basándose en este análisis.

Esta realización es posible porque, como se discutirá con mayor detalle más adelante, los diferentes modos de compresión disponibles son particularmente adecuados y eficaces para determinados tipos de bloque. Por ejemplo, el esquema de la presente invención es eficaz en la gestión de bloques con transiciones de color de variación lenta. El esquema iPACKMAN/ETC es muy eficaz para gestionar bloques de imagen en donde los colores de los elementos de imagen tienen crominancia bastante similar pero distinta luminancia. Un tercer esquema posible podría ser THUMB [8], que también se puede utilizar como modo auxiliar a iPACKMAN/ETC. Este esquema tiene dos modos o, así denominados, patrones, que están adaptados para gestionar bloques de imagen que tengan dos crominancias (colores) distintas.

Esta realización tiene la ventaja de que solo se necesita generar uno y no cuatro bloques candidatos comprimidos, aunque al coste de un análisis de bloque y de color y del riesgo de seleccionar un esquema de compresión no óptimo.

En la Figura 10, el primer paso S20 de compresión puede representar los pasos S2 a S4 de la Figura 1, es decir, la asignación de pesos de color, la determinación de claves de color y la representación de los colores originales de los elementos de imagen.

La Figura 11 es un diagrama de flujo que ilustra distintas realizaciones de los otros pasos S21, S22 y S23 de compresión de la Figura 10 según los esquemas iPACKMAN/ETC y THUMB.

Comenzando con THUMB, el procedimiento continúa desde el paso S1 de la Figura 1. En un siguiente paso S30, se determinan una primera y una segunda claves de color. La primera clave de color es una representación de un primer valor de color y la segunda clave de color es, análogamente, una representación de un segundo valor de color. El primer y segundo valores de color están ubicados en una primera línea en el espacio de color, preferiblemente el espacio RGB. Esta primera línea tiene también una primera dirección. En un siguiente paso S31, se proporciona una clave de modificador de color. Esta clave de modificador es una representación de al menos un modificador de color aplicable para modificar el primer valor de color a lo largo de una segunda línea que tiene una segunda dirección en el espacio de color. Al modificar el primer valor de color con el al menos un modificador de

color, se obtienen a lo largo de la segunda línea múltiples representaciones de color. En esta realización, la segunda dirección es diferente de la primera dirección, es decir, la primera línea y la segunda línea no son paralelas.

En el paso S32 se selecciona un índice de color, asociado a una representación de color, de entre i) las múltiples representaciones de color a lo largo de la segunda línea e ii) al menos una representación de color basada en el segundo valor de color. Este paso de selección de índice se realiza preferiblemente para cada elemento de imagen del bloque, lo que se ilustra esquemáticamente por la línea L4.

La resultante representación de bloque comprimida de este modo comprenderá, por tanto, la primera y la segunda claves de color, la clave de modificador de color y una secuencia de índices de color.

En realidad, el esquema THUMB antes descrito puede ejecutarse, a su vez, conforma a dos modos o patrones, dependiendo de cómo estén distribuidos en el espacio de color los colores de los elementos de imagen. Esto significa que THUMB ocupa dos de los cuatro modos distintos de la Figura 10. Las Figuras 12 - 14B describen con más detalle el uso de THUMB. En la Figura 12A se representan en el espacio de color los (16) colores originales de elementos de imagen de un bloque a comprimir. De la Figura se desprende que los colores están ubicados en dos grupos 2, 4, conteniendo cada uno múltiples colores. Tal distribución de color se gestiona ventajosamente con el patrón/modo H de THUMB, que se ilustra en la Figura 12B.

En la Figura 12B, el primer valor 10 de color representado por la primera clave de color y el segundo valor 20 de color representado por la segunda clave de color están ubicados en una primera línea 40 que tiene una primera dirección 45. Del primer valor 10 de color se pueden derivar dos representaciones 30, 32 de color utilizando un modificador de color representado por la clave de modificador de color. Estas dos representaciones 30, 32 de color y el primer valor 10 de color se ubican en una segunda línea 12 que tiene una segunda dirección 15, segunda dirección 15 que es diferente de la primera dirección 45. En este patrón/modo H se utiliza análogamente un modificador de color representado por la clave de modificador de color para modificar el segundo valor 20 de color, con el fin de obtener dos representaciones 31, 33 de color. Estas dos representaciones 31, 33 de color y el segundo valor 20 de color están ubicados a lo largo de una tercera línea 22, que tiene una tercera dirección 25. En el ejemplo ilustrado, la segunda 15 y tercera 25 direcciones son paralelas.

Después se asocian los índices de color seleccionados para los elementos de imagen con una de las cuatro representaciones 30, 31, 32, 33 de color.

La Figura 13A es un diagrama correspondiente de una distribución original de color gestionada eficazmente mediante un patrón/modo T de THUMB. En la Figura, los colores están ubicados en dos grupos 2, 4, de manera similar a la Figura 12A. Sin embargo, al contrario que en la Figura 12A, uno de los grupos 4 tiene una forma general circular en lugar de elíptica.

La Figura 13B ilustra cómo maneja THUMB una situación de este tipo. El primer valor 10 y el segundo valor 20 de color están ubicados en la primera línea 40 que tiene la primera dirección 45. El primer valor 10 de color es modificado por un modificador de color para generar una primera 30 y una segunda 32 representaciones de color ubicadas en la segunda línea 12 que tiene la segunda dirección 15. En este patrón/modo, las representaciones de color disponibles para los elementos de imagen son la primera representación 30 de color y la segunda representación 32 de color, el primer valor de color 10 y el segundo valor 20 de color. Cada índice de color seleccionado para los elementos de imagen está asociado con una de estas cuatro representaciones posibles.

Si, por el contrario, en el paso S22 de la Figura 10 se utiliza el esquema iPACKMAN/ETC, en el paso S30 se determina una primera clave de color como una representación de un primer valor de color. En este paso S30, también se determina una segunda clave de color. Sin embargo, esta segunda clave es una representación de un color diferencial. Entonces se puede obtener un segundo valor de color como una suma del primer valor de color y el color diferencial. A los elementos de imagen de un primer sub-bloque (2 x 4 o 4 x 2 elementos de imagen) se les asigna el primer valor de color, mientras que a los elementos de imagen de un segundo sub-bloque (2 x 4 o 4 x 2 elementos de imagen) se les asigna el segundo valor de color.

En el paso S31 se proporciona una clave de intensidad, en donde la clave de intensidad es una representación de un conjunto de múltiples modificadores de intensidad. Estos modificadores de intensidad se pueden aplicar para modificar la intensidad del primer o segundo valores de color, con el fin de generar múltiples representaciones de color. En una implementación preferida, la clave de intensidad es un índice tabular a una tabla de intensidades que comprende múltiples conjuntos de modificadores, en donde los conjuntos de modificadores tienen distintos modificadores de intensidad. En el paso S32, se selecciona un índice de intensidad para cada elemento de imagen del bloque, en donde el índice de intensidad está asociado con un modificador de intensidad procedente del conjunto de modificadores de intensidad representado por la clave de intensidad.

La Figura 14 ilustra esquemáticamente una posible representación 700 de bloque comprimida para el modo diferencial iPACKMAN/ETC. El bloque comprimido 700 incluye la primera clave 710 de color que incluye tres componentes de color, rojo 712, verde 714 y azul 716, preferiblemente representado cada uno por cinco bits. Análogamente, la segunda clave 720 de color o color diferencial incluye tres componentes, rojo 722, verde 724 y azul 726, preferiblemente representado cada uno por tres bits. El bloque comprimido 700 incluye, además, dos

claves 750A, 750B de intensidad, una para cada sub-bloque 2x4/4x2, cada una preferiblemente de 3 bits. También se incluye en el bloque comprimido 700 una secuencia 760 de índices de intensidad, preferiblemente un índice de intensidad de 2 bits por cada elemento de imagen del bloque, lo que da como resultado 32 bits. Un "flipbit" (bit de inversión) 770 define si los dos sub-bloques del bloque de imagen son dos bloques 2x4 o dos bloques 4x2, es decir, si están colocados verticalmente (flipbit = 0_{bin}) u horizontalmente (flipbit = 1_{bin}). iPACKMAN/ETC comprende dos modos del tipo denominado "por defecto", uno de los cuales se ha descrito y expuesto en la presente memoria. Un "diffbit" (bit de diferencia) 780 discrimina entre estos dos modos predeterminados. En las Figuras 14 a 17, este diffbit 780 es igual a, y se ha fijado en, 1_{bin} (o 0_{bin}). Téngase en cuenta que el orden respectivo de las claves 710, 720, 750A, 750B, la secuencia 760 de índices, el flipbit 770 y el diffbit 780 del bloque de imagen codificado 700 puede ser distinto del ilustrado en la Figura. El tamaño total del bloque comprimido son 64 bits.

En el modo iPACKMAN/ETC antes mencionado, los componentes 712, 714, 716 de color de la primera clave 710 de color comprenden cada uno preferiblemente 5 bits, que representan básicamente cualquier valor en el intervalo de 0 a 31 (de 00000_{bin} a 11111_{bin}). Los componentes 722, 724, 726 de la segunda clave 720 comprenden cada uno preferiblemente 3 bits, que se utilizan para representar un valor en el intervalo de -4 a +3. Los componentes de color del segundo valor de color se pueden obtener sumando los componentes 712, 714, 716; 722, 724, 726 de las dos claves:

Componente rojo = R + dR

Componente verde = G + dG

Componente azul = B + dB

Dado que estos componentes de color representan información de intensidad, se les permite tomar los valores de 0 (ninguna intensidad) a 31 (máxima intensidad). Esto significa que el codificador que trabaje en este modo de iPACKMAN/ETC nunca utilizará combinaciones de bits de la primera clave 710 de color y la segunda clave 720 de color que den lugar a que las sumas R + dR, G + dG, B + dB se desborden, es decir, sean <0 o >31. Esto permite introducir tres modos auxiliares que se pueden utilizar para complementar iPACKMAN/ETC.

En el primer modo auxiliar, el componente del color rojo se desborda, es decir, R + dR es menor que cero o mayor que 31. Esto sucede si los tres primeros bits del componente 712 de rojo de la primera clave 710 de color son iguales y distintos del primer bit del componente 722 de rojo de la segunda clave 720 de color.

En la Figura 15 se emplea este principio para utilizar el patrón/modo H de THUMB como un modo auxiliar para iPACKMAN/ETC. Así, en este modo no se pueden seleccionar libremente cuatro bits 790 ya que los componentes de rojo deben desbordarse, lo que sucede si estos cuatro bits 790 son iguales a 1110_{bin} o 0001_{bin}. La representación 700 de bloque comprimida tiene preferiblemente un total de 64 bits, de los cuales cuatro se han utilizado según lo anterior. Los 60 bits restantes se dividen preferiblemente entre las partes anexas según lo siguiente; se asignan 4 bits a cada componente de color 712, 714, 716; 722, 724, 726 de la primera 710 y segunda 720 claves de color. La clave 750 de modificador de color comprende tres bits, el diffbit 780 un bit (que tiene el mismo valor que en la Figura 14) y la secuencia 760 de índices de color preferiblemente 32 bits.

Se puede obtener un segundo modo auxiliar si se desborda el componente de verde, es decir, G + dG es menor que cero o mayor que 31, y no se deja que el componente de rojo se desborde, es decir, $0 \leq R + dR \leq 31$. En este modo, el primer bit del componente 712 de rojo de la primera clave de color de la Figura 14 se fija distinto del segundo o tercer bit del componente 712 de rojo. Además, los tres primeros bits del componente 714 de verde de la primera clave 710 de color deben ser iguales y distintos del primer bit del componente 724 de verde de la segunda clave 720 de color.

En la Figura 16 no se pueden fijar libremente el bit0 (corresponde al primer bit del componente de rojo en la primera clave de color de la Figura 14), del bit8 al bit10 (corresponden a los tres primeros bits del componente de verde en la primera clave de color de la Figura 14) y el bit13 (corresponde al primer bit del componente de verde en la segunda clave de color de la Figura 14), representados colectivamente por 790 en la Figura. Por lo tanto, quedan 59 bits para uso en las otras partes del bloque comprimido 700. La división de bits entre las claves 710, 720 de color, la clave 750 de modificador de color, el diffbit 780 y la secuencia 760 de índices de color es preferiblemente la misma que para la Figura 15 excepto que la clave 750 de modificador de color comprende en este caso solamente dos bits en lugar de tres.

Está disponible un tercer modo auxiliar que utiliza el esquema de la presente invención si se desborda el componente de azul, es decir, B + dB es menor que cero o mayor que 31, pero no se deja que los componentes de rojo y de verde se desborden. Esto significa que el primer bit del componente 712 de rojo y 714 de verde de la primera clave 710 de color de la Figura 14 debe ser distinto del segundo o tercer componente de los componentes 712 de rojo y 714 de verde. Además, el componente de azul se desborda, es decir, B + dB es menor que cero o mayor que 31.

En la Figura 17, por lo tanto, no se pueden fijar libremente el bit0 (corresponde al primer bit del componente de rojo en la primera clave de color de la Figura 14), el bit8 (corresponde al primer bit del componente de verde en la

primera clave de color de la Figura 14), del bit16 al bit18 (corresponden a los tres primeros bits del componente de azul en la primera clave de color de la Figura 14) y el bit21 (corresponde al primer bit del componente de azul en la segunda clave de color de la Figura 14), denominados colectivamente 790. A los componentes 712, 722, 732 de rojo y 716, 726, 736 de azul de las tres claves 710, 720, 730 de color se les asignan preferiblemente 6 bits a cada uno, mientras que los correspondientes componentes 714, 724, 734 de verde comprenden 7 bits cada uno y el diffbit 780 es un bit. Esto suma, en total, 64 bits.

Si son posibles las cuatro representaciones comprimidas distintas ilustradas en las Figuras 14-17, un índice de modo utilizado para discriminar entre los cuatro modos incluye preferiblemente posiciones definidas en las secuencias de bits. Estas posiciones de bit incluyen los tres primeros bits del componente de rojo, de verde y de azul de la primera clave de color, y el primer bit del componente de rojo, de verde y de azul de la segunda clave de color de la Figura 14. Además, preferiblemente se utiliza el diffbit para discriminar entre el otro modo iPACKMAN/ETC disponible. Téngase en cuenta, sin embargo, que algunos de los bits de este índice de modo se pueden utilizar para codificar el bloque comprimido en los distintos modos.

Descompresión

La Figura 18 ilustra un diagrama de flujo de un método para descodificar una imagen codificada o versión codificada de una imagen original según la presente invención. La imagen codificada comprende básicamente varias representaciones codificadas de bloques de imagen. Estas representaciones de bloques codificadas se generan preferiblemente mediante el método de codificación de imagen descrito más arriba.

El método comienza por lo general identificando el bloque o bloques de imagen codificada que hay que descodificar. Podría ocurrir que hubiera que descodificar todos los bloques de imagen codificados de una imagen codificada, para generar una representación descodificada de la imagen original. Como alternativa, solo hay que acceder a una parte de la imagen original. En consecuencia, solamente se tienen que descodificar un número seleccionado de bloques de imagen (o, más exactamente, una cantidad seleccionada de elementos de imagen de ciertos bloques de imagen).

Una vez se han identificado (la representación de) el bloque o (las representaciones de) los bloques de imagen codificados correctos, el paso S40 determina al menos dos valores de color basados en las al menos dos claves de color de las representaciones de bloque comprimidas. En una implementación preferida, este paso de determinación implica expandir el color cuantificado de la clave de color, por ejemplo RGB676, a, preferiblemente, 24 bits (RGB888). Si la segunda clave de color comprende componentes de color diferenciales, estos componentes se añaden preferiblemente a componentes correspondientes de la primera clave de color antes de la expansión, para generar un segundo valor de color. En una implementación preferida de la presente invención, la representación de bloque comprimida comprende tres claves de color. Esto significa que en el paso S40 se determinan tres valores de color mediante la extensión de cada una de las claves.

Los siguientes pasos S41 y S42 se realizan para cada elemento de imagen que ha de descodificarse, lo que se ilustra esquemáticamente por la línea L5. En el paso S41 se proporcionan los pesos de color asignados al elemento de imagen a descodificar. Los pesos de color son preferiblemente pesos predefinidos, que dependen de la posición real del elemento de imagen en los bloques:

$$w_0^{xy} = f_0(x, y)$$

$$w_H^{xy} = f_H(x, y)$$

$$w_V^{xy} = f_V(x, y)$$

Por lo tanto, todos los bloques de imagen de la imagen comprimida según la presente invención tienen preferiblemente la misma asignación de pesos de color, de manera que un elemento de imagen en, por ejemplo, la posición (2,1) de un primer bloque tendrá los mismos pesos de color que un elemento de imagen en la posición (2,1) de un segundo bloque.

Sin embargo, la presente invención prevé que podrían existir alternativas en la asignación de pesos que se lleva a cabo bloque por bloque. En tal caso, la representación comprimida comprende preferiblemente una clave de peso. Esto significa que la determinación de pesos en el paso S41 se realiza entonces basándose en la clave de peso, es decir, el conjunto de pesos de color utilizado para el bloque actual se identifica con base en la clave de peso.

En un siguiente paso S42, la representación de color utilizada para representar el color original del elemento de imagen a descodificar se calcula basándose en los pesos de color proporcionados y los al menos dos valores de color determinados. En una implementación preferida, la representación de color se calcula como combinación (lineal) ponderada, mediante los pesos proporcionados, de los valores de color determinados. En este contexto, en el cálculo se utilizan preferiblemente todos los valores de color, pero podría ser posible utilizar solamente un subconjunto de los mismos.

Se podrían realizar los pasos S41 y S42 para varios elementos de imagen del bloque de imagen (lo que se ilustra esquemáticamente mediante la línea L5). La invención prevé que, en algunas aplicaciones, solamente se descodifique un único elemento de imagen a partir de un bloque de imagen específico, se descodifiquen múltiples elementos de imagen de un bloque de imagen específico y/o se descodifiquen todos los elementos de imagen de un bloque específico.

Después se repiten, preferiblemente, los pasos S40 a S42 para todos los bloques de imagen que comprenden elementos de imagen que deban descodificarse (lo que se ilustra esquemáticamente mediante la línea L6). Esto significa que el bucle de pasos S40 a S42 podría realizarse una vez, pero más frecuentemente varias veces para diferentes bloques de imagen codificados y/o varias veces para un bloque de imagen codificado específico.

En el paso opcional S43 se genera una representación descodificada de la imagen original, o una parte de la misma, basándose en los elementos y bloques de imagen descodificados. Después, el método termina.

La Figura 19 es un diagrama de flujo de una implementación multimodal de la descodificación/descompresión de imágenes y bloques de la presente invención. El método comienza en el paso S50 donde, basándose en un índice de modo, se selecciona un modo de descompresión para uso en el bloque actual. Si son posibles las cuatro representaciones comprimidas distintas ilustradas en las Figuras 14 a 17, el índice de modo incluye el diffbit, los tres primeros bits del componente de rojo, de verde y de azul de la primera clave de color y el primer bit del componente de rojo, de verde y de azul de la segunda clave de color de la Figura 14. Así, el descodificador investiga estas posiciones de bit en la secuencia de bits que constituye la representación de bloque comprimida y selecciona el modo de descompresión basándose en los bits investigados. En una implementación preferida, se selecciona un primer modo de compresión si el componente del color azul se desborda, pero no el del rojo y el del verde. Se seleccionan un segundo y un tercer modo si se desborda el componente de rojo o se desborda el componente de verde, pero no el del rojo. Si ninguno de los componentes se desborda, en el paso S50 se selecciona el cuarto modo.

Si en el paso S50 se selecciona el primer modo, el procedimiento continúa al paso S51, donde se descomprime el bloque según este modo. Esto corresponde a realizar los pasos S40-S42 ilustrados en la Figura 18. Si, en cambio, se selecciona un segundo, tercer o cuarto modo, el procedimiento continúa al paso S52, S53 o S54.

La Figura 20 ilustra la descompresión realizada según el modo THUMB. Comenzando en el paso S60, se determina un primer valor de color basándose en la primera clave de color. En el paso S61 se determina un segundo valor de color basándose en la segunda clave de color. Estos dos valores de color están ubicados en una primera línea que tiene una primera dirección en el espacio de color (véanse las Figuras 12B y 13B). Las determinaciones de color de los pasos S60 y S61 implican preferiblemente expandir la secuencia de bits de las claves para generar los valores de color. Un siguiente paso S62 genera múltiples representaciones de color a lo largo de una segunda línea que tiene una segunda dirección en el espacio de color, modificando el primer valor de color con al menos un modificador de color representado por la clave de modificador de color. Esta segunda dirección es distinta de la primera dirección. El siguiente paso S63 se lleva a cabo para cada elemento de imagen a descodificar, lo que se ilustra esquemáticamente mediante la línea L7. Este paso S63 implica seleccionar, basándose en la secuencia de índices de color y, más exactamente, en el índice de color asignado al elemento de imagen correspondiente, una representación de color de entre i) las múltiples representaciones de color a lo largo de la segunda línea e ii) al menos una representación de color basada en el segundo valor de color. En el patrón/modo H, también se utiliza al menos un modificador de color basado en la clave de modificador, para modificar el segundo valor de color a lo largo de una tercera línea que tiene una tercera dirección (distinta de la primera dirección) con el fin de generar múltiples representaciones de color. Por lo tanto, en este patrón H están disponibles dos conjuntos de múltiples representaciones de color (uno de ellos situado en la segunda línea y el otro situado en la tercera línea) y el índice de color de los elementos de imagen apunta a una de las representaciones de los dos conjuntos. En el patrón/modo T, las múltiples representaciones de color de la segunda línea se complementan con el primer y el segundo valor de color, que también se pueden seleccionar como representaciones de color para los elementos de imagen. El método continúa entonces al paso S43 de la Figura 18.

Si, en cambio, basándose en el índice de modo se selecciona el modo iPACKMAN/ETC, se determina un valor de color basándose en la primera clave de color o la primera y la segunda claves de color del paso S70 de la Figura. 21. Si el elemento de imagen a descodificar está presente en un primer sub-bloque (2x4/4x2), el valor de color se determina basándose en la primera clave de color, preferiblemente mediante la expansión de la secuencia de bits de la clave de RGB555 a RGB888. Si, en cambio, el elemento de imagen está presente en un segundo sub-bloque (2x4/4x2), el valor de color se determina basándose tanto en la primera como en la segunda claves de color, básicamente mediante la suma de los componentes de rojo, los componentes de verde y los componentes de azul de la primera y la segunda claves y la posterior expansión del resultado a RGB888 (o bien, como alternativa, primeramente la expansión de los componentes de claves y después la suma de los mismos). En este modo, el bloque comprimido comprende dos claves de intensidad, una por cada sub-bloque. La clave de intensidad asignada al sub-bloque que comprende el elemento de imagen a descodificar se utiliza en el paso S71 para proporcionar un conjunto de múltiples modificadores de intensidad. Este paso comprende preferiblemente proporcionar, basándose en la clave de intensidad, el conjunto de modificadores a partir de una tabla que comprende múltiples conjuntos de modificadores de este tipo. En el paso S72 se selecciona un modificador de intensidad para uso en el elemento de

imagen, a partir del conjunto de modificadores proporcionados, basándose en el índice de intensidad asignado al elemento de imagen. En el siguiente paso S73, el modificador seleccionado modifica en intensidad el valor de color determinado, con el fin de generar una representación de color para el elemento de imagen. Preferiblemente, se repiten los pasos S70 a S73 para todos los elementos de imagen del bloque que han de descodificarse. El método continúa después al paso S43 de la Figura 18.

Ejemplos de descompresión

A continuación se ofrecen ejemplos de descompresión utilizando un diseño de secuencia de bits como se ilustra en las Figuras 14 a 17.

iPACKMAN/ETC

El bloque de imagen comprimido está representado por la siguiente secuencia de bits:

10110 010 11010 110 00100 000 101 110 1 1

10 01 11 00 01 01 10 11 10 00 11 00 01 01 00 01

Primeramente se investigan los bit0-bit2, bit6, bit8-10, bit13, bit16-18, bit21 y el diffbit 780 para determinar el modo de descompresión a utilizar para este bloque de imagen. Dado que ninguno de los componentes de color se desborda y el diffbit 780 está fijado en 1, se debe seleccionar el modo por defecto diferencial de iPACKMAN/ETC.

En primer lugar, se expanden a RGB888 los componentes 712, 714, 716 de color de la primera clave 710 de color, para generar el primer valor de color:

Rojo: $10110_{bin} \Rightarrow 10110101_{bin} = 181$

Verde: $11010_{bin} \Rightarrow 11010110_{bin} = 214$

Azul: $00100_{bin} \Rightarrow 00100001_{bin} = 33$

A estos componentes se les suman los componentes diferenciales 722, 724, 726 de la segunda clave 720 de color para obtener el segundo valor de color:

Rojo: $010_{bin} \leftrightarrow 2 \Rightarrow 181+2=183$

Verde: $110_{bin} \leftrightarrow 2 \Rightarrow 214-2=212$

Azul: $000_{bin} \leftrightarrow 0 \Rightarrow 33+2=33$

El flipbit se fija en 1_{bin}, lo que implica que se asigna el primer valor de color a los ocho elementos de imagen de las dos filas superiores del bloque 4x4, mientras que se utiliza el segundo valor de color para los ocho elementos de imagen de las dos filas inferiores.

Las dos claves 750A, 750B de intensidad apuntan a una tabla de intensidades, ejemplificada por la Tabla 8 a continuación:

Tabla 8 - Tabla de intensidades

Clave de intensidad	11_{bin}	10_{bin}	00_{bin}	01_{bin}
000_{bin}	-8	-2	2	8
001_{bin}	-12	-4	4	12
010_{bin}	-31	-6	6	31
011_{bin}	-34	-12	12	34
100_{bin}	-50	-8	8	50
101_{bin}	-57	-19	19	57
110_{bin}	-80	-28	28	80
111_{bin}	-127	-42	42	127

La primera clave 750A de intensidad aplicable para elementos de imagen en el primer sub-bloque 2x4 es 101_{bin}, que representa los modificadores de intensidad -57, -19, 19, 57. En cambio, la segunda clave 750B de intensidad representa los modificadores de intensidad -80, -28, 28, 80.

El primer elemento de la imagen en la posición (0,0) tendrá la siguiente representación de color:

(181, 214, 33) + (-19, -19, -19) = (162, 195, 14)

En consecuencia, la representación de color del último elemento de imagen (en la posición (3,3)) se calcula de la manera siguiente:

(183, 212, 33) + (80, 80, 80) = (255, 255, 113)

después de restringir los valores calculados de componentes de color entre el valor mínimo permitido de 0 y el valor máximo de 255.

Después se prosigue este procedimiento para el resto de los elementos de imagen de los bloques de imagen.

Patrón H de THUMB

El bloque de imagen comprimido se representa por la siguiente secuencia de bits:

111 10 0 10 1101 0110 0010 0000 1011 101 1

10 01 11 00 01 01 10 11 10 00 11 00 01 01 00 01

En este caso, los bits bit0-bit2 son todos iguales y diferentes de bit5, lo que significa que el componente rojo se desborda y se debe utilizar un primer modo auxiliar, es decir el patrón H de THUMB. La secuencia de bits presentada más arriba tiene la disposición que se ilustra en la Figura 15.

El primer y segundo valores de color se generan mediante la expansión de los componentes 712, 714, 716; 722, 724, 726 de las dos claves 710, 720 de color:

Rojo 0: $1010_{bin} \Rightarrow 10101010_{bin}$=170
Verde 0: $1101_{bin} \Rightarrow 11011101_{bin}$=221
Azul 0: $0110_{bin} \Rightarrow 01100110_{bin}$=102

Rojo 1: $0010_{bin} \Rightarrow 00100010_{bin}$=34
Verde 1: $0000_{bin} \Rightarrow 00000000_{bin}$=0
Azul 1: $1011_{bin} \Rightarrow 10111011_{bin}$=187

Por lo tanto, el primer valor de color es (170, 221, 102) y el segundo valor es (34, 0, 187).

La clave 750 de modificador 101_{bin}=5 implica que hay que desplazar hacia la izquierda el número 1_{bin} cinco veces para obtener 100000_{bin}=32. Se utiliza este valor para modificar los dos valores de color con el fin de obtener cuatro representaciones de color:

C0: (170, 221, 102) - (32, 32, 32) = (138, 189, 70)

C1: (170, 221, 102) + (32, 32, 32) = (202, 253, 134)

C2: (34, 0, 187) - (32, 32, 32) = (2, 0, 155)

C3: (34, 0, 187) + (32, 32, 32) = (66, 32, 219)

El primer elemento de imagen tiene el índice de color 10_{bin}, lo que implica que para este elemento de imagen se utiliza la representación C2 de color. Se prosigue este procedimiento para el resto de elementos de imagen (índice $00_{bin}\leftrightarrow$C0, $01_{bin}\leftrightarrow$C2, $10_{bin}\leftrightarrow$C2 y $11_{bin}\leftrightarrow$ C3).

Patrón T de THUMB

El bloque de imagen comprimido se representa por la siguiente secuencia de bits:

1 0110 010 000 1 0 1 100 0100 0001 0111 01 1

10 01 11 00 01 01 10 11 10 00 11 00 01 01 00 01

En este caso, el componente verde desborda, ya que bit8-bit10 son iguales y distintos de bit12. Además, el componente de rojo no desborda, porque bit0 es distinto de bit1. Esto significa que se debe seleccionar un segundo modo de descompresión auxiliar en forma de patrón T de THUMB y que la secuencia de bits se interpreta con la disposición de la Figura 16.

Los dos valores de color se calculan del mismo modo que antes para el patrón P:

Rojo 0: $0110_{bin} \Rightarrow 01100110_{bin}$=102
Verde 0: $0101_{bin} \Rightarrow 01010101_{bin}$=85
Azul 0: $0100_{bin} \Rightarrow 01000100_{bin}$=68

Rojo 1: $0100_{bin} \Rightarrow 01000100_{bin}$=68
Verde 1: $0001_{bin} \Rightarrow 00010001_{bin}$=17
Azul 1: $0111_{bin} \Rightarrow 01110111_{bin}$=119

En este caso el modificador de color 750 incluye solamente dos bits 01_{bin}=1, lo que implica que hay que desplazar hacia la izquierda una posición el número 1_{bin} para obtener 10_{bin}=2. Utilizando este valor modificador se calculan dos de las cuatro posibles representaciones de color, mientras que las otras dos representaciones son iguales a los dos valores de color:

C0: (102, 85, 68)

C1: (102, 85, 68) - (2, 2, 2) = (100, 83, 66)

C2: (102, 85, 68) + (2, 2, 2) = (104, 87, 70)

C3: (68, 17, 119)

El primer elemento de imagen tiene el índice de color 10_{bin}, que corresponde a C2. Después se repite el procedimiento para el resto de los elementos de imagen del bloque.

PLANO

El bloque de imagen comprimido se representa por la siguiente secuencia de bits:

1 011001 0 0 101011 0 000 00 1 010 111001 1

1001110 001011 011100 0110001 010001

En este ejemplo, el componente azul se desborda ya que bit16-bit18 son iguales y diferentes de bit21. Además, bit0 es distinto de bit1 (el rojo no se desborda) y bit8 es distinto de bit9 (el verde no se desborda). En consecuencia, se debe utilizar un cuarto modo de descompresión tal como se define en la presente invención, PLANO.

En este ejemplo, se calculan tres valores de color mediante la expansión a RGB888 de los componentes 712, 714, 716; 722, 724, 726; 732, 734, 736 de color de las tres claves 710, 720, 730.

R_0: $011001_{bin} \Rightarrow 01100101_{bin}$=101

G_0: $0101011_{bin} \Rightarrow 01010110_{bin}$=86

B_0: $000010_{bin} \Rightarrow 00001000_{bin}$=8

R_H: $111001_{bin} \Rightarrow 11100111_{bin}$=231

G_H: $1001110_{bin} \Rightarrow 10011101_{bin}$=157

B_H: $001011_{bin} \Rightarrow 00101100_{bin}$= 44

R_V: $011100_{bin} \Rightarrow 01110001_{bin}$=113

G_V: $0110001_{bin} \Rightarrow 01100010_{bin}$=98

B_V: $010001_{bin} \Rightarrow 01000101_{bin}$=69

Después se ponderan estos valores de color y se combinan según las enseñanzas de la Tabla 6 precedente. La representación de color para el primer elemento (0,0) de imagen es simplemente el primer valor (101, 86, 8). La representación de color para el elemento (1,0) de imagen es tres cuartas partes del primer valor de color y una cuarta parte del segundo valor de color, es decir, $\frac{3}{4}(101, 86, 8) + \frac{1}{4}(231, 157, 44) = (133, 104, 17)$. Se prosigue este procedimiento para el resto de los elementos de imagen, a fin de proporcionar una representación descodificada del bloque de imagen.

Aspectos de implementación

El esquema de codificación de imagen (codificación de bloques de imagen) y de descodificación de imagen

(descodificación o procesamiento de bloques de imagen) según la presente invención podría proveerse en un sistema de procesamiento de datos en general, por ejemplo en un terminal de usuario u otra unidad configurada para procesar y/o representar imágenes. Un terminal de este tipo podría ser un ordenador, por ejemplo un ordenador personal (PC, por sus siglas en inglés), una consola de juegos o un cliente ligero, por ejemplo un asistente personal digital (PDA), una unidad móvil y un teléfono.

Terminal de usuario

La Figura 22 ilustra un terminal 100 de usuario representado por una unidad móvil. Sin embargo, la invención no está limitada a las unidades móviles, y podría implementarse en otros terminales y unidades de procesamiento de datos, tales como ordenadores PC y consolas de juegos. En la Figura solo se ilustran medios y elementos de la unidad móvil 100 directamente implicados en la presente invención.

La unidad móvil 100 comprende una unidad (central) 200 de procesamiento (CPU, por sus siglas en inglés) para procesar datos, entre ellos datos de imagen, dentro de la unidad móvil 100. Se proporciona un sistema gráfico 130 en la unidad móvil 100 para gestionar datos de imagen y datos gráficos. En particular, el sistema gráfico 130 está adaptado para representar o visualizar imágenes en una pantalla conectada 120 u otra unidad de visualización. La unidad móvil 100 también comprende un almacenamiento o memoria 140 para almacenar datos en el mismo. En esta memoria 140 se pueden almacenar datos de imagen, en particular datos de imagen codificados (bloques de imagen codificados) según la presente invención.

En la unidad móvil 100 se proporciona un codificador 210 de imágenes según la presente invención. Este codificador 210 está configurado para codificar una imagen o textura a una representación codificada de la imagen (o textura). Como se ha discutido antes, tal representación codificada comprende una secuencia o archivo de múltiples bloques de imagen codificados. Este codificador 210 de imágenes puede proporcionarse como *software* que se ejecuta en la CPU 200, tal como se ilustra en la Figura. Como alternativa, o adicionalmente, se puede disponer el codificador 210 en el sistema gráfico 130 o en otro lugar de la unidad móvil 100.

Se puede proporcionar una representación codificada de una imagen procedente del codificador 210 de bloques a la memoria 140 a través de un bus (de memoria) 150, para almacenarla en la misma hasta una representación ulterior de la imagen. Como alternativa, o adicionalmente, se pueden reenviar los datos de imagen codificados a una unidad 110 de entrada y salida (I/O, por sus siglas en inglés) para ser transmitidos (inalámbricamente o por cable) a otros terminales o unidades externos. Esta unidad I/O 110 también puede estar adaptada para recibir datos de imagen desde una unidad externa. Estos datos de imagen podrían ser una imagen que debe ser codificada por el codificador 210 de imágenes o datos codificados de imagen que deben ser descodificados. También sería posible almacenar la representación de imagen codificada en una memoria de texturas dedicada prevista, por ejemplo, en el sistema gráfico 130. Además, también, o como alternativa, se podrían almacenar (temporalmente) partes de la imagen codificada, en una memoria *caché* de texturas, por ejemplo, en el sistema gráfico 130.

Se proporciona en la unidad móvil 100 un descodificador 220 de imágenes según la presente invención para descodificar una imagen codificada con el fin de generar una representación de la imagen descodificada. Esta representación descodificada podría corresponder a toda la imagen original o a una parte de la misma. El descodificador 220 de imágenes proporciona datos de imagen descodificados al sistema gráfico 130, que a su vez normalmente procesa los datos antes de representarlos o presentarlos en la pantalla 120. Se puede disponer el descodificador 220 de imágenes en el sistema gráfico 130, tal como se ilustra en la Figura. Como alternativa, o adicionalmente, se puede proporcionar el descodificador 200 como *software* que se ejecuta en la CPU 200 o en otro lugar de la unidad móvil 100.

La unidad móvil 100 podría estar equipada con un codificador 210 de imágenes y con un descodificador 220 de imágenes, como se ilustra en la Figura. Sin embargo, para algunos terminales 100 podría ser posible incluir solo un codificador 210 de imágenes. En tal caso, se podrían transmitir datos de imagen codificados a otro terminal que realice la descodificación y, posiblemente, la representación de la imagen. En consecuencia, un terminal 100 podría incluir solo un descodificador 220 de imágenes, es decir, ningún codificador. Un terminal 100 de este tipo recibe entonces de otro terminal una señal que comprende datos de imagen codificados y la descodifica para generar una representación de imagen descodificada. Por tanto, se podría transmitir de forma inalámbrica la señal de imagen codificada entre los terminales, utilizando transmisor y receptor de radio. Como alternativa, se podrían emplear otras técnicas para distribuir imágenes y representaciones de imagen codificadas entre terminales según la invención, tales como Bluetooth®, técnicas IR utilizando puertos IR, y transferencia cableada de datos de imagen entre terminales. También se podrían utilizar tarjetas o chips de memoria que se pueden conectar e intercambiar entre los terminales para esta distribución de datos de imagen entre terminales.

Las unidades 110, 130, 200, 210 y 220 de la unidad móvil 100 pueden proporcionarse como *software*, *hardware* o una combinación de ambos.

Codificador de imágenes

La Figura 23 ilustra un diagrama de bloques de una realización de un codificador 210 de imágenes según la presente invención. El codificador 210 comprende típicamente un desintegrador 215 de imágenes para

descomponer o dividir una imagen de entrada en varios bloques de imagen. Preferiblemente, el desintegrador 215 está configurado para descomponer la imagen en bloques de imagen que comprenden dieciséis elementos de imagen (píxeles, téxeles o vóxeles), es decir, que tienen un tamaño general de 4x4 elementos de imagen.

Este desintegrador 215 podría estar adaptado para descomponer distintas imágenes de entrada en bloques de imagen con distintos tamaños. En tal caso, el desintegrador 215 recibe preferiblemente información de entrada, que permite la identificación del formato de bloque de imagen a utilizar para una imagen dada.

Esta realización del codificador 210 de imágenes comprende un único codificador 300 de bloques. Este codificador 300 de bloques codifica el bloque o bloques de imagen recibidos desde el desintegrador de imágenes para generar la representación o representaciones de bloque codificadas. El tamaño global de la representación de bloque es menor que el tamaño correspondiente del bloque de imagen sin codificar. El codificador 300 de bloques está configurado preferiblemente para procesar (codificar) secuencialmente cada bloque de imagen procedente del desintegrador 215.

En una implementación alternativa, el codificador 210 incluye múltiples codificadores 300 de bloques para procesar en paralelo múltiples bloques de imagen procedentes del desintegrador 215 de imágenes, lo que reduce el tiempo total de codificación de imágenes.

Las unidades 215 y 300 del codificador 210 de imágenes pueden proporcionarse como *software*, *hardware* o una combinación de ambos. Las unidades 215 y 300 pueden implementarse juntas en el codificador 210 de imagen. Como alternativa, también es posible una implementación distribuida, siendo proporcionadas algunas de las unidades en otra parte de la unidad móvil.

Codificador de bloques

La Figura 24 ilustra un diagrama de bloques de una realización de un codificador 300 de bloques según la presente invención, tal como el codificador de bloques del codificador de imágenes de la Figura 23. El codificador 300 comprende un asignador 310 de pesos para asignar pesos de color a al menos un subconjunto de los elementos de imagen de un bloque de imagen que ha de comprimirse. En una implementación preferida, el asignador 310 de pesos asigna un peso Z de color por cada elemento de imagen del bloque, donde Z es un número múltiple que es igual al número de claves de color que un cuantificador 320 color determina para el bloque de imagen. En otra realización preferida, el asignador 310 asigna pesos de color a los elementos de imagen de un bloque de manera que los valores de los componentes de color de al menos un componente de color de las representaciones de color utilizadas para representar los colores originales de los elementos de imagen cambien monótonamente a lo largo de una fila o/y columna de elementos de imagen del bloque. La asignación de pesos realizada por el asignador 310 del codificador 300 de bloques de elementos de imagen de un bloque se lleva a cabo preferiblemente basándose en la posición de los elementos de imagen, es decir, las coordenadas relativas de los elementos de imagen en el bloque.

El cuantificador 320 de color del codificador 300 de bloques está configurado para determinar, basándose al menos parcialmente en los pesos de color asignados por el asignador 310, al menos dos claves de color para el bloque de imagen. En una implementación preferida, el cuantificador 320 de color determina tres claves de color, preferiblemente tres claves RGB676.

Las unidades 310 y 320 del codificador 300 de bloques pueden proporcionarse como *software*, *hardware* o una combinación de ambos. Las unidades 310 y 320 se pueden implementar juntas en el codificador 300 de bloques. Como alternativa, también es posible una implementación distribuida, siendo proporcionadas algunas de las unidades en otra parte del codificador de imágenes.

La Figura 25 es un diagrama de bloques esquemático de otra realización de un codificador 300 de bloques según la presente invención. Este codificador 300 de bloques está adaptado para funcionar según diferentes modos de compresión, preferiblemente cuatro modos diferentes. En un primer modo de compresión, el asignador 310 de pesos y el cuantificador 320 de color se hacen funcionar conforme a la discusión precedente en relación con la Figura 24. Por lo tanto, esta da lugar a un bloque de imagen comprimido que comprende tres claves de color y un índice de modo, que se describirá más adelante. En la Figura 17 se ilustra un ejemplo de un bloque comprimido de este tipo.

En el modo de compresión iPACKMAN/ETC, se hace funcionar el cuantificador 320 de color para determinar una primera clave de color que es una representación de un primer valor de color y para determinar una segunda clave de color como una representación de un color diferencial, que se puede sumar al primer valor de color para obtener un segundo valor de color. Se hace funcionar de esta manera un cuantificador 340 de modificadores para determinar al menos una, y preferiblemente dos, claves de intensidad como representación de al menos un conjunto de múltiples modificadores de intensidad utilizados para modificar el primer o segundo valor de color con el fin de obtener representaciones de color. Las claves de intensidad son, preferiblemente, índices de tabla de una tabla 500 de modificadores que comprende múltiples conjuntos de modificadores de este tipo. En el codificador 300 de bloques se proporciona un selector 350 de índice con el fin de determinar, para cada elemento de imagen del bloque, un índice de intensidad asociado con uno de los modificadores de intensidad del conjunto o los conjuntos de modificadores representados por la clave o las claves de intensidad.

En los dos modos de THUMB, el cuantificador 320 de color determina una primera clave de color como una representación de un primer valor de color. Además, el cuantificador 320 determina una segunda clave como representación de un segundo valor de color, donde estos dos valores están ubicados en una primera línea con una primera dirección en el espacio de color. Se hace funcionar de esta manera el cuantificador 340 de modificadores para proporcionar una clave de modificador de color como una representación de al menos un modificador de color aplicable para modificar el primer valor de color a lo largo de una segunda línea que tiene una segunda dirección en el espacio de color. Esta modificación de color da lugar a múltiples representaciones de color a lo largo de la segunda línea. Las segunda y primera direcciones no son paralelas. Después, el selector 350 de índice selecciona, para cada elemento de imagen, un índice de color asociado con una representación de color seleccionada de entre i) las representaciones de color a lo largo de la segunda línea e ii) al menos una representación de color basada en el segundo valor de color.

En una implementación preferida, se determinan múltiples representaciones candidatas comprimidas para un bloque de imagen dado, una representación por cada modo de compresión. Después se implementa un selector 360 de modo para seleccionar la representación candidata que se debe utilizar como representación comprimida para el bloque de imagen. Esta selección se lleva a cabo preferiblemente basándose en una comparación de estimaciones de error, una de tales estimaciones por cada modo de compresión. El selector 360 de modo selecciona preferiblemente el candidato que lleve a un error menor. A continuación, un gestor 370 de índice de modo compila un índice de modo representativo del modo de compresión que da lugar al error más pequeño, es decir, el modo utilizado cuando se genera el candidato seleccionado por el selector 360 de modo. Este índice de modo constituye una parte del bloque de imagen comprimida.

Las unidades 310 a 370 del codificador 300 de bloques pueden proporcionase como *software*, *hardware* o una combinación de ambos. Las unidades 310 a 370 y 500 se pueden implementar juntas en el codificador 300 de bloques. Como alternativa, también es posible una implementación distribuida, siendo proporcionadas algunas de las unidades en otra parte del codificador de imágenes.

Descodificador de imágenes

La Figura 26 ilustra un diagrama de bloques de una realización de un descodificador 220 de imágenes según la presente invención. El descodificador 220 de imágenes comprende preferiblemente un selector 222 de bloques que está adaptado para seleccionar, por ejemplo desde una memoria, el bloque o bloques de imagen codificados que deban ser proporcionados a un descodificador 400 de bloques para su descodificación. El selector 222 de bloques recibe preferiblemente información de entrada asociada con los datos de imagen codificados, por ejemplo desde un encabezado o un motor de representación. Entonces, basándose en la información de entrada, se calcula una dirección de un bloque de imagen codificado que tiene el o los elementos de imagen deseados. Esta dirección calculada depende preferiblemente de las coordenadas del elemento de imagen (píxel, téxel o vóxel) dentro de una imagen. Utilizando la dirección, el selector 222 de bloques identifica el bloque de imagen codificado de la memoria. Después, se toma del almacenamiento este bloque de imagen codificado identificado y se le envía al descodificador 400 de bloques.

El acceso (aleatorio) a elementos de imagen de un bloque de imagen permite ventajosamente la descodificación selectiva solamente de las partes necesarias de una imagen. Además, se puede descodificar la imagen siguiendo cualquier orden en el que se requieran los datos. Por ejemplo, en la asignación de texturas se pueden necesitar solamente partes de la textura, y generalmente se requerirán estas partes en un orden no secuencial. Por lo tanto, la descodificación de imágenes de la presente invención puede aplicarse ventajosamente para procesar solo una parte o sección de una imagen.

A continuación se envía al descodificador 400 de bloques el bloque de imagen codificado seleccionado. Además del bloque de imagen, el descodificador 400 recibe preferiblemente información que especifica los elementos de imagen del bloque que deben descodificarse. La información podría especificar que se debe descodificar todo el bloque de imagen, es decir, todos los elementos de imagen del mismo. No obstante, la información recibida podría identificar solamente un único elemento de imagen o unos pocos de los elementos de imagen que deben descodificarse. El descodificador 400 de bloques genera entonces una representación descodificada del elemento o elementos de imagen del bloque. Esta representación descodificada es preferiblemente un color P-bit, donde P es el número de bits por elemento de imagen en la imagen original, por ejemplo, un color RGB de 24 bits.

Se podría disponer en el descodificador 220 de imágenes un compositor 224 de imágenes opcional. Este compositor recibe los elementos de imagen descodificados desde el descodificador 400 de bloques y los compone para generar un píxel que se puede representar o visualizar en una pantalla. Como alternativa, se podría disponer este compositor 224 de imágenes en el sistema gráfico.

Como alternativa, el descodificador 220 de imágenes comprende múltiples descodificadores 400 de bloques. Al tener acceso a múltiples descodificadores 400 de bloques, el descodificador 220 de imágenes puede procesar (descodificar) en paralelo múltiples bloques de imagen codificados. Estos descodificadores múltiples 400 de bloques permiten un procesamiento paralelo que aumenta el rendimiento de procesamiento y la eficacia del descodificador 220 de imágenes.

Las unidades 222, 224 y 400 del descodificador 220 de imágenes pueden proporcionarse como *software*, *hardware* o una combinación de ambos. Las unidades 222, 224 y 400 se pueden implementar juntas en el descodificador 220 de imágenes. Como alternativa, también es posible una implementación distribuida, siendo proporcionadas algunas de las unidades en otra parte del terminal de usuario.

Descodificador de bloques

La Figura 27 es una ilustración de una realización de un descodificador 400 de bloques según la presente invención. El descodificador 400 de bloques comprende un generador 410 de color que genera al menos dos valores de color basándose en las al menos dos claves de color de la representación de bloque comprimida. Preferiblemente, este generador 410 de color está configurado para expandir o ampliar los componentes cuantificados de color de las claves de color a, preferiblemente, RGB888. En el descodificador 400 de bloques está dispuesto un gestor 420 de pesos con el fin de proporcionar, para cada elemento de imagen que deba descodificarse, pesos de color asignados al elemento o elementos de imagen. En una implementación preferida, los elementos de imagen correspondientes en una posición dada de bloques de imagen diferentes tienen los mismos pesos de color asignados. Por tanto, los pesos de color dependen de las coordenadas o las posiciones de los elementos de imagen en el bloque, pero no cambian para diferentes bloques comprimidos según la presente invención. Por tanto, el gestor 420 de pesos proporciona preferiblemente pesos de color basados en posiciones/coordenadas de elementos de imagen dentro del bloque de imagen.

Al generador 410 de color y al gestor 420 de pesos está conectada un calculador 430 de color que utiliza los pesos de color proporcionados y los valores de color generados para determinar una representación de color para su uso como una representación del color original del elemento de imagen. Preferiblemente, el calculador 430 se implementa para combinar los valores de color procedentes del generador 410 pero ponderados con los pesos de color procedentes del gestor 420 de pesos.

Las unidades 410 a 430 del descodificador 400 de bloques pueden proporcionarse como *software*, *hardware* o una combinación de ambos. Las unidades 410 a 430 se pueden implementar juntas en el descodificador 400 de bloques. Como alternativa, también es posible una implementación distribuida, siendo proporcionadas algunas de las unidades en otra parte del descodificador de imágenes.

La Figura 28 es un diagrama de bloques esquemático de otra realización de un descodificador 400 de bloques según la presente invención adaptado para el funcionamiento multimodal. El descodificador 400 de bloques comprende un selector 460 de modo, que selecciona entre múltiples modos disponibles, preferiblemente cuatro modos, el modo de descompresión a utilizar cuando se descomprime la representación de bloque comprimida en cuestión. Este selector 460 de modo utiliza un índice de modo en el bloque comprimido para seleccionar el modo correcto.

Si el selector 460 selecciona un primer modo de descompresión, se hacen funcionar el generador 410 de color, el gestor 420 de pesos y el calculador 430 de color como se ha descrito en lo que antecede en relación con la Figura 27.

Si, en cambio, el selector 460 selecciona un segundo modo de descompresión que corresponde a iPACKMAN/ETC, el generador 410 de color determina un valor de color basándose en la primera clave de color o basándose en la primera y segunda claves de color (dependiendo de la posición real del elemento de imagen del bloque). En el primer caso, simplemente se expanden los colores del componente de cuantificador a, preferiblemente, RGB888. En el segundo caso, se suman los componentes diferenciales de la segunda clave a los componentes de color derivables de la primera clave, a fin de determinar el valor de color. En el descodificador 400 de bloques se proporciona un gestor 470 de modificadores para proporcionar, basándose en uno de la al menos una clave de intensidad, un conjunto de múltiples modificadores de intensidad, preferiblemente tomados de una tabla 500 de modificadores. Un selector 450 de color selecciona, utilizando un índice de intensidad asociado con el elemento de imagen en cuestión, uno de los modificadores de intensidad de entre el conjunto proporcionado. A continuación, un modificador 440 de color utiliza este modificador de intensidad seleccionado para modificar la intensidad del valor de color con el fin de calcular una representación de color para el elemento de imagen.

Si el selector 460 de modo selecciona los modos THUMB, el generador 410 de color determina un primer valor de color utilizando la primera clave de color, y determina un segundo valor de color basándose en la segunda clave de color. Los dos valores están ubicados en una primera línea que tiene una primera dirección en el espacio de color. El modificador 440 de color genera múltiples representaciones de color a lo largo de una segunda línea que tiene una segunda dirección distinta, modificando el primer valor de color con al menos un modificador de color representada por la clave de modificador de color. A continuación, el selector 450 de color selecciona, basándose en la secuencia de índices de color, una representación de color de entre i) las múltiples representaciones de color a lo largo de la segunda línea e ii) al menos una representación de color basada en el segundo valor de color.

Las unidades 410 a 470 del descodificador 400 de bloques pueden proporcionarse como *software*, *hardware* o una combinación de ambos. Las unidades 410 a 470 y 500 se pueden implementar juntas en el descodificador 400 de bloques. Como alternativa, también es posible una implementación distribuida, siendo proporcionadas algunas de

las unidades en otra parte del descodificador de imágenes.

Un experto en la técnica comprenderá que se pueden efectuar diversas modificaciones y cambios en la presente invención sin apartarse del alcance de la misma, que está definido por las reivindicaciones adjuntas.

Referencias

[1] Delp, Mitchell: Image Compression using Block Truncation Coding. IEEE Transactions on Communications 2, 9 (1979), 1335-1342

[2] Campbell, Defant, Frederiksen, Joyce, Leske, Lindberg, Sandin: Two Bit/Pixel Full Color Encoding. En Proceedings of SIGGRAPH (1986), vol. 22, págs. 215-223

[3] Patente de EE. UU. 5,956,431

[4] S. Fenney, "Texture compression using low-frequency signal modulation", Graphics Hardware 2003, págs. 84-91, julio de 2003

[5] Solicitud internacional WO 2005/059836

[6] Solicitud internacional WO 2006/006915

[7] Strom, Akenine-Möller: iPACKMAN high-quality, low complexity texture compression for mobile phones, Graphics Hardware 05, Los Angeles, EE.UU., junio de 2005

[8] Strom, Pettersson: "Texture compression: THUMB - Two Hues Using Modified Brightness", SIGRAD'05, Lund, Suecia, noviembre de 2005

REIVINDICACIONES

1. Un método para comprimir un bloque (600) de imagen que comprende múltiples elementos (610) de imagen en una representación (700) de bloque comprimida, dicho método comprende los pasos de:

asignar, para cada componente de color de cada elemento (610) de imagen de dicho bloque (600) de imagen, los pesos de color predefinidos $w^{x,y}$ basándose en una posición (x, y) de dicho elemento (610) de imagen en dicho bloque (600) de imagen;

determinar, en base a dichos pesos de color predefinidos $w^{x,y}$ asignados y a los colores originales de dichos elementos (610) de imagen múltiples, al menos dos claves (710, 720, 730, 740) de color que son representaciones de al menos dos valores de color, comprendiendo dicha representación (700) de bloque comprimida dichas al menos dos claves (710, 720, 730, 740) de color; y

representar colores originales de dichos múltiples elementos (610) de imagen mediante representaciones de color que se derivan de dichos al menos dos valores de color, en donde dichas representaciones de color de dichos elementos (610) de imagen se derivan de combinaciones lineales de dichos al menos dos valores de color ponderados mediante dichos pesos de color predefinidos $w^{x,y}$ asignados, y

en donde dicha representación de color de dicho elemento de imagen se obtiene multiplicando cada componente de color de dichos al menos dos valores de color por el correspondiente peso de color predefinido $w^{x,y}$ asignado de dicho componente de color en dicha posición (x, y) de dicho elemento de imagen,

caracterizado por que

dichos pesos de color $w^{x,y}$ están predefinidos para que dichos pesos de color $w^{x,y}$ disminuyan monótonamente a lo largo de cada fila de elementos (610) de imagen en dicho bloque (600) de imagen, o aumenten monótonamente a lo largo de cada fila de elementos (610) de imagen en dicho bloque (600) de imagen; y

dichos pesos de color $w^{x,y}$ están predefinidos para que dichos pesos de color $w^{x,y}$ disminuyan monótonamente a lo largo de cada columna de elementos (610) de imagen en dicho bloque (600) de imagen, o aumenten monótonamente a lo largo de cada columna de elementos (610) de imagen en dicho bloque (600) de imagen.

2. El método según la reivindicación 1, en el que

dicha representación (700) de bloque comprimida comprende una primera clave (710) de color, una segunda clave (720) de color y una tercera clave (730) de color;

dicha primera clave (710) de color ocupa 6 bits para un componente de color (712) rojo, 7 bits para un componente de color (714) verde y 6 bits para un componente de color (716) verde, dicha segunda clave (720) de color ocupa 6 bits para un componente de color (722) rojo, 7 bits para un componente de color (724) verde y 6 bits un componente de color (726) verde, y dicha tercera clave (730) de color ocupa 6 bits para un componente de color (732) rojo, 7 bits para un componente de color (734) verde y 6 bits para un componente de color (736) verde; y

el número de bits disponibles para codificar dichos colores originales de dichos elementos (610) de imagen múltiples es 57.

3. El método según la reivindicación 1, en el que

dicha representación (700) de bloque comprimida comprende una primera clave (710) de color, una segunda clave (720) de color, una tercera clave (730) de color y 7 bits que utilizan un índice de modo; y

dicha representación (700) de bloque comprimida tiene un tamaño total de 64 bits

4. El método según cualquiera de las reivindicaciones 1 a 3, en el que dicho paso de asignación se realiza antes de dicho paso de determinación.

5. El método según cualquiera de las reivindicaciones 1 a 4, en el que dicho paso de determinación comprende determinar, basándose en dichos pesos de color predefinidos $w^{x,y}$ asignados y en dichos colores originales de dichos elementos (610) de imagen múltiples, tres claves (710, 720, 730) de color que son representaciones de tres valores de color.

6. Un método para codificar una imagen, comprendiendo dicho método los pasos de:

- descomponer dicha imagen en bloques (600) de imagen, comprendiendo cada bloque (600) de imagen múltiples elementos (610) de imagen; y

- determinar, para al menos un bloque (600) de imagen, una representación (700) de bloque comprimida por compresión de dicho al menos un bloque (600) de imagen según cualquiera de las reivindicaciones 1 a 5.

7. Un método para descomprimir una representación (700) de bloque comprimida en un bloque (600) de imagen que comprende múltiples elementos (610) de imagen, comprendiendo dicho método los pasos de:

asignar, para cada componente de color de cada elemento (610) de imagen de dicho bloque (600) de imagen, los pesos de color predefinidos $w^{x,y}$ basándose en una posición (x, y) de dicho elemento (610) de imagen en dicho bloque (600) de imagen;

obtener, a partir de dicha representación (700) de bloque comprimida, al menos dos claves (710, 720, 730, 740) de color que son representaciones de al menos dos valores de color; y

obtener una representación de color de un elemento de imagen en la posición (x, y) en dicho bloque (600) de imagen multiplicando cada componente de color de dichos al menos dos valores de color por el peso de color predefinido $w^{x,y}$ asignado correspondiente de dicho componente de color en dicha posición (x, y) de dicho elemento de imagen;

en donde dichas representaciones de color de dichos elementos (610) de imagen se derivan de combinaciones lineales de dichos al menos dos valores de color ponderados mediante dichos pesos de color predefinidos $w^{x,y}$ asignados; y

en donde los colores originales de dichos elementos (610) de imagen múltiples están representados por representaciones de color que se derivan de dichos al menos dos valores de color,

caracterizado por que

dichos pesos de color $w^{x,y}$ están predefinidos para que dichos pesos de color $w^{x,y}$ disminuyan monótonamente a lo largo de cada fila de elementos (610) de imagen en dicho bloque (600) de imagen, o aumenten monótonamente a lo largo de cada fila de elementos (610) de imagen en dicho bloque (600) de imagen; y

dichos pesos de color $w^{x,y}$ están predefinidos para que dichos pesos de color $w^{x,y}$ disminuyan monótonamente a lo largo de cada columna de elementos (610) de imagen en dicho bloque (600) de imagen, o aumenten monótonamente a lo largo de cada columna de elementos (610) de imagen en dicho bloque (600) de imagen.

8. Un sistema (300) para comprimir un bloque (600) de imagen que comprende múltiples elementos (610) de imagen en una representación (700) de bloque comprimida, comprendiendo dicho sistema (300):

un asignador (310) de peso para asignar, para cada componente de color de cada elemento (610) de imagen de dicho bloque (600) de imagen, pesos de color predefinidos $w^{x,y}$ basándose en una posición (x, y) de dicho elemento (610) de imagen en dicho bloque (600) de imagen; y

un cuantificador (320) de color para determinar, basándose en dichos pesos de color predefinidos $w^{x,y}$ asignados y en los colores originales de dichos elementos (610) de imagen múltiples, al menos dos claves (710, 720, 730, 740) de color que son representaciones de al menos dos valores de color, comprendiendo dicha representación (700) de bloque comprimida dichas al menos dos claves (710, 720, 730, 740) de color,

en donde dichos colores originales de dichos elementos (610) de imagen múltiples están representados por representaciones de color que se derivan de dichos al menos dos valores de color, y

en donde dichas representaciones de color de dichos elementos (600) de imagen se derivan de combinaciones lineales de dichos al menos dos valores de color ponderados mediante dichos pesos de color predefinidos $w^{x,y}$ asignados,

caracterizado por que

dichos pesos de color $w^{x,y}$ están predefinidos para que dichos pesos de color $w^{x,y}$ disminuyan monótonamente a lo largo de cada fila de elementos (610) de imagen en dicho bloque (600) de imagen, o aumenten monótonamente a lo largo de cada fila de elementos (610) de imagen en dicho bloque (600) de imagen; y

dichos pesos de color $w^{x,y}$ están predefinidos para que dichos pesos de color $w^{x,y}$ disminuyan monótonamente a lo largo de cada columna de elementos (610) de imagen en dicho bloque (600) de imagen, o aumenten monótonamente a lo largo de cada columna de elementos (610) de imagen en dicho bloque (600) de imagen.

9. El sistema según la reivindicación 8, en el que dicho cuantificador (320) de color está dispuesto para determinar, basándose en dichos pesos de color predefinidos $w^{x,y}$ asignados y en dichos colores originales de dichos elementos (610) de imagen múltiples, tres claves (710, 720, 730) de color que son representaciones de tres valores de color.

10. El sistema de acuerdo con la reivindicación 8 o 9, en el que dicho sistema (300) está adaptado para comprimir dicho bloque (600) de imagen según múltiples modos de compresión para obtener múltiples bloques (700) de imagen candidatos comprimidos, comprendiendo además dicho sistema (300):

- un selector (360) de modo para seleccionar un bloque de imagen candidato comprimido de entre dichos múltiples bloques de imagen candidatos comprimidos como representación (700) de bloque comprimida de dicho bloque (600) de imagen; y

- un gestor (370) de índice de modo para proporcionar un índice de modo (710, 720, 780) asociado con un modo de compresión utilizado para comprimir dicho bloque de imagen candidato comprimido seleccionado, en donde dicho asignador (310) de pesos y dicho cuantificador (320) de color se hacen funcionar según un primer modo de compresión.

11. Un sistema (210) para codificar una imagen, comprendiendo dicho sistema (210):

- un desintegrador (215) de imágenes para descomponer dicha imagen en bloques (600) de imagen, comprendiendo cada bloque (600) de imagen múltiples elementos (610) de imagen; y

- un sistema (300) según cualquiera de las reivindicaciones 8 a 10 para determinar, para al menos un bloque (600) de imagen, una representación (700) de bloque comprimida por compresión de dicho al menos un bloque (600) de imagen.

12. Un sistema (400) para descomprimir una representación (700) de bloque comprimida en un bloque (600) de imagen que comprende múltiples elementos (610) de imagen, comprendiendo dicho sistema (400):

un gestor (420) de pesos para asignar, para cada componente de color de cada elemento (610) de imagen de dicho bloque (600) de imagen, pesos de color predefinidos $w^{x,y}$ basándose en una posición (x, y) de dicho elemento (610) de imagen en dicho bloque (600) de imagen;

un generador (410) de color para obtener, a partir de dicha representación (700) de bloque comprimida, al menos dos claves (710, 720, 730, 740) de color que son representaciones de al menos dos valores de color; y

un calculador (430) de color para obtener una representación de color de un elemento de imagen en la posición (x, y) en dicho bloque (600) de imagen multiplicando cada componente de color de dichos al menos dos valores de color por el correspondiente peso predefinido $w^{x,y}$ asignado de dicho componente de color en dicha posición (x, y) de dicho elemento de imagen;

en donde dichas representaciones de color de dichos elementos (610) de imagen se derivan de combinaciones lineales de dichos al menos dos valores de color ponderados mediante dichos pesos de color predefinidos $w^{x,y}$ asignados, y

en donde los colores originales de dichos elementos (610) de imagen múltiples están representados por representaciones de color que se derivan de dichos al menos dos valores de color,

caracterizado por que

dichos pesos de color $w^{x,y}$ están predefinidos para que dichos pesos de color $w^{x,y}$ disminuyan monótonamente a lo largo de cada fila de elementos (610) de imagen en dicho bloque (600) de imagen, o aumenten monótonamente a lo largo de cada fila de elementos (610) de imagen en dicho bloque (600) de imagen; y

dichos pesos de color $w^{x,y}$ están predefinidos para que dichos pesos de color $w^{x,y}$ disminuyan monótonamente a lo largo de cada columna de elementos (610) de imagen en dicho bloque (600) de imagen, o aumenten monótonamente a lo largo de cada columna de elementos (610) de imagen en dicho bloque (600) de imagen.

13. El sistema según la reivindicación 12, que comprende además un selector (460) de modo para seleccionar un modo de descompresión basándose en un índice (710, 720, 780) de modo de dicha representación (700) de bloque comprimida, siendo hechos funcionar dicho generador (410) de color, dicho gestor (420) de pesos y dicho calculador (430) de color en un primer modo de descompresión.

14. Sistema según la reivindicación 13, en donde dicha representación comprimida (700) comprende, si dicho selector (460) de modo selecciona un segundo modo de descompresión, una primera clave (710) de color, una segunda clave (720) de color, una clave (750) de modificador de color y una secuencia (760) de índices de color y dicho sistema (300) comprende, si dicho selector (460) de modo selecciona dicho segundo modo de descompresión:

un primer generador (410) de color para determinar un primer valor (10) de color basándose en dicha primera clave (710) de color;

un segundo generador (410) de color para determinar un segundo valor (20) de color basándose en dicha segunda clave (720) de color, estando ubicados dichos primer (10) y segundo (20) valores de color en una primera línea (40) que tiene una primera dirección (45) en el espacio de color;

un modificador (440) de color para generar múltiples representaciones (30, 32; 10) de color a lo largo de una segunda línea (12) que tiene una segunda dirección (15) en el espacio de color mediante la modificación de dicho

primer valor (10) de color con al menos un modificador de color representado por dicha clave (750) de modificador de color, siendo dicha segunda dirección (15) distinta de dicha primera dirección (45); y

un selector (450) de color para seleccionar, para al menos un elemento (610) de imagen de dicho bloque (600) de imagen y basándose en dicha secuencia (760) de índices de color, una representación en color de entre i) dichas múltiples representaciones (30, 32; 10) de color a lo largo de dicha segunda línea (12) e ii) al menos una representación (31, 33; 20) de color basada en dicho segundo valor (20) de color.

15. El sistema según la reivindicación 13 o 14, en donde dicha representación (700) de bloque comprimida comprende, si dicho selector (460) de modo selecciona un tercer modo de descompresión, una primera clave (710) de color, una segunda clave (720) de color, al menos una clave (750A, 750B) de intensidad y una secuencia (760) de índices de intensidad, y dicho sistema (400) comprende, si dicho selector (460) de modo selecciona dicho tercer modo de descompresión:

un generador (410) de color para determinar un valor de color basándose en dicha primera clave (710) de color o basándose en dichas primera (710) y segunda (720) claves de color;

un gestor (470) de modificadores para proporcionar un conjunto de múltiples modificadores de intensidad basándose en dicha al menos una clave (750A, 750B) de intensidad;

un selector (450) de intensidad para seleccionar, para al menos un elemento (610) de imagen de dicho bloque (600) de imagen, un modificador de intensidad de entre dicho conjunto de modificadores de intensidad basándose en dicha secuencia (760) de índices de intensidad; y

un modificador (440) de intensidad para generar una representación de color mediante la modificación de la intensidad de dicho valor de color basándose en dicho modificador de intensidad seleccionado

16. Un sistema (220) para descodificar una imagen codificada que comprende representaciones (700) de bloque comprimidas de bloques (600) de imagen, comprendiendo cada bloque (600) de imagen múltiples elementos (610) de imagen, comprendiendo dicho sistema (220):

un sistema (400) de descompresión según cualquiera de las reivindicaciones 12 a 15 para descomprimir representaciones (700) de bloque comprimidas de bloques (600) de imagen con el fin de generar múltiples representaciones de color de elementos (610) de imagen; y

un compositor (224) de imágenes para generar una representación descodificada de dicha imagen codificada mediante la composición de dichas múltiples representaciones de color de elementos (610) de imagen.

17. Un terminal (100) de usuario que comprende un sistema (210; 220; 300; 400) según cualquiera de las reivindicaciones 8 a 16.

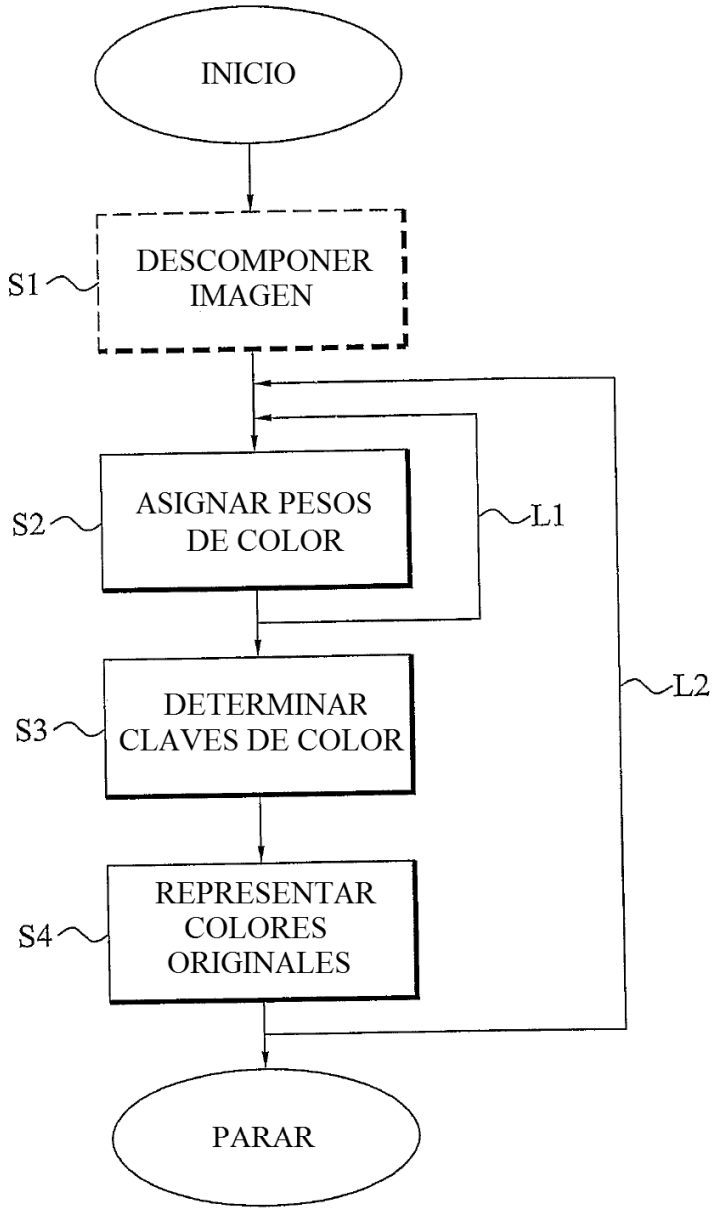
INICIO
S1
DESCOMPONER IMAGEN
S2
ASIGNAR PESOS DE COLOR
L1
S3
DETERMINAR CLAVES DE COLOR
L2
S4
REPRESENTAR COLORES ORIGINALES
PARAR

Fig. 1

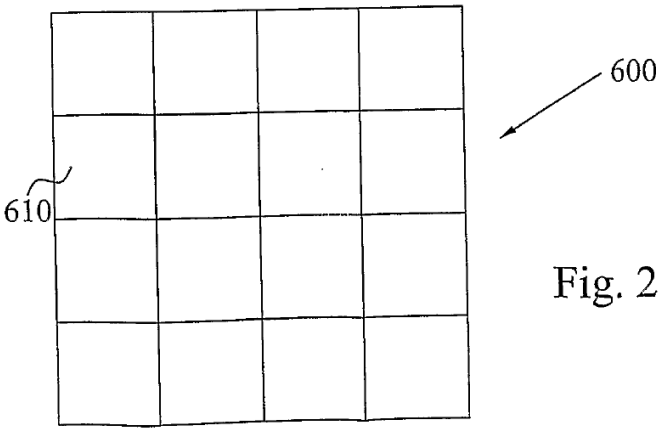
600
610

Fig. 2

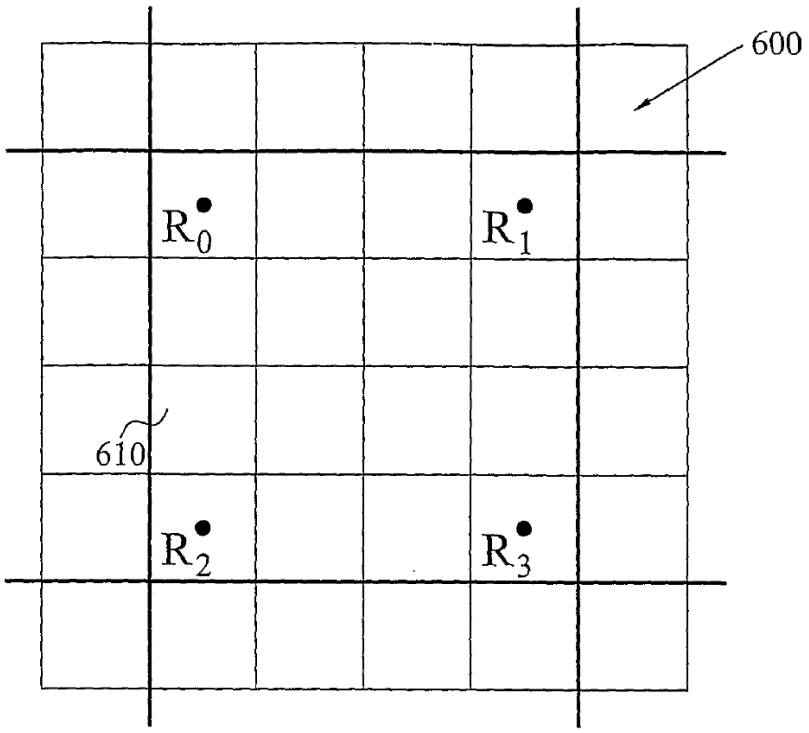

Fig. 3

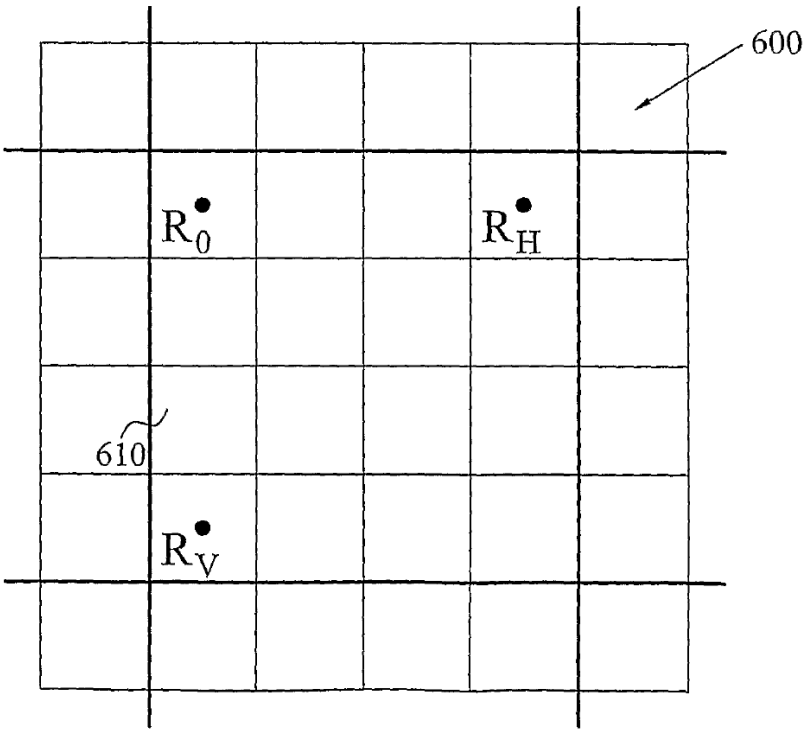

Fig. 5

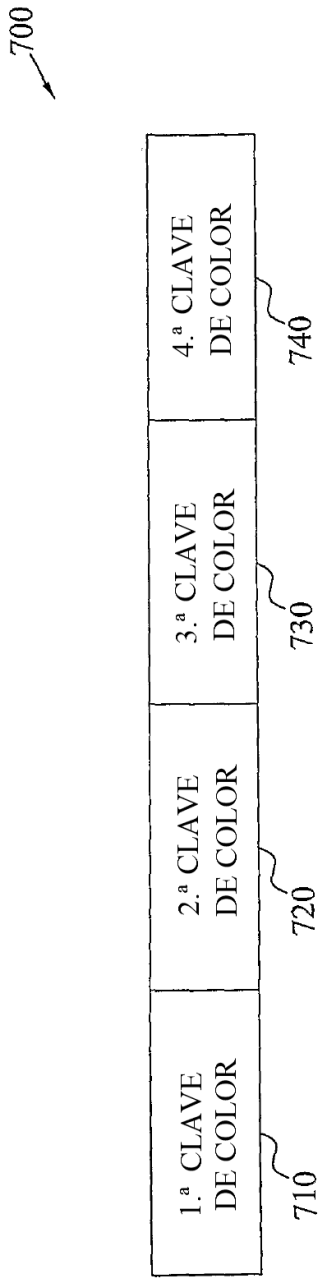
700
1.ª CLAVE DE COLOR
2.ª CLAVE DE COLOR
3.ª CLAVE DE COLOR
4.ª CLAVE DE COLOR
710
720
730
740

Fig. 4

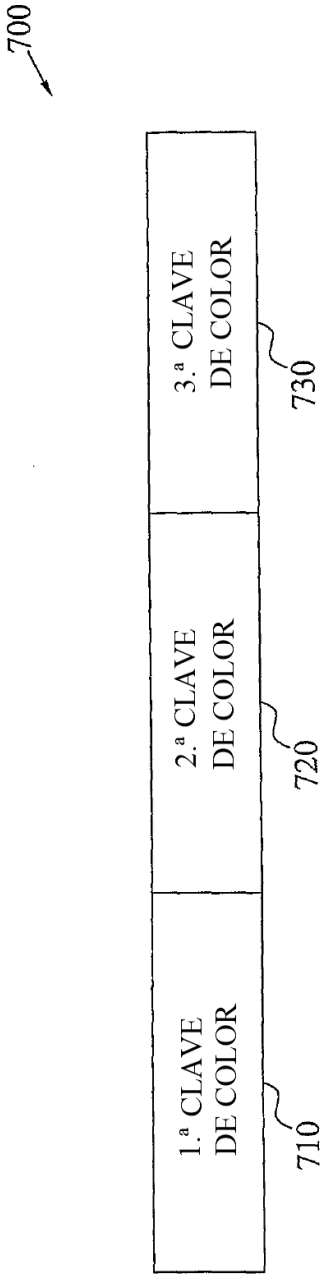
700
1.ª CLAVE DE COLOR
2.ª CLAVE DE COLOR
3.ª CLAVE DE COLOR
710
720
730

Fig. 6

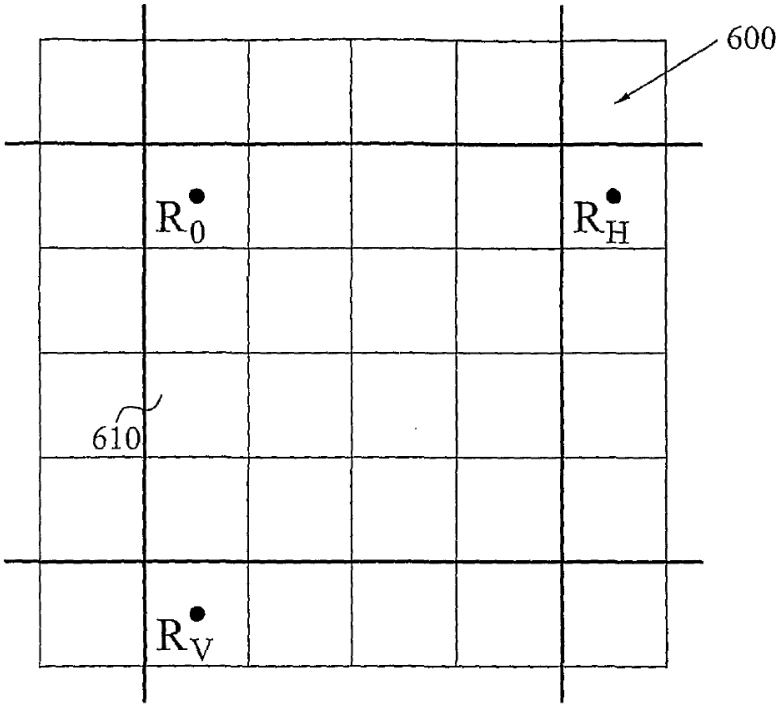

Fig. 7

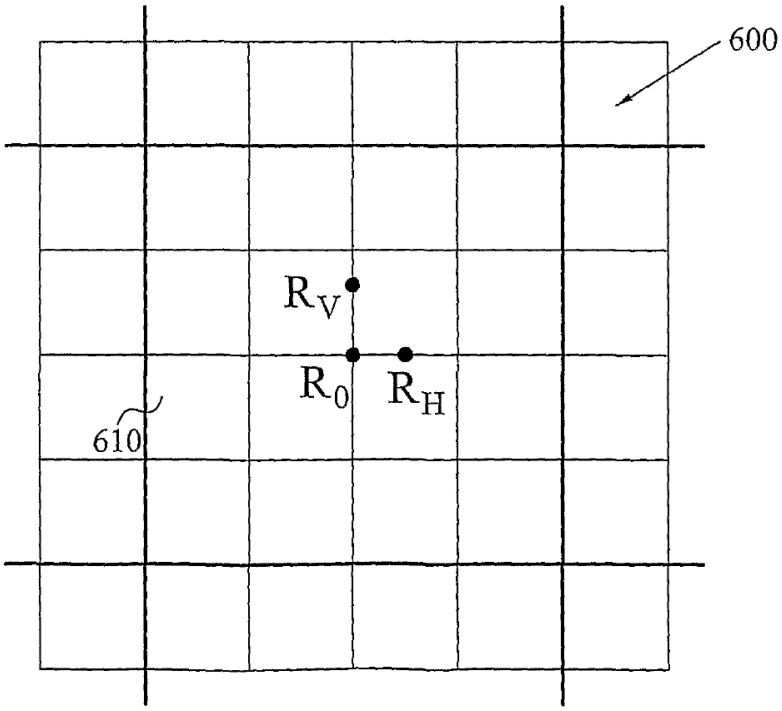

Fig. 8

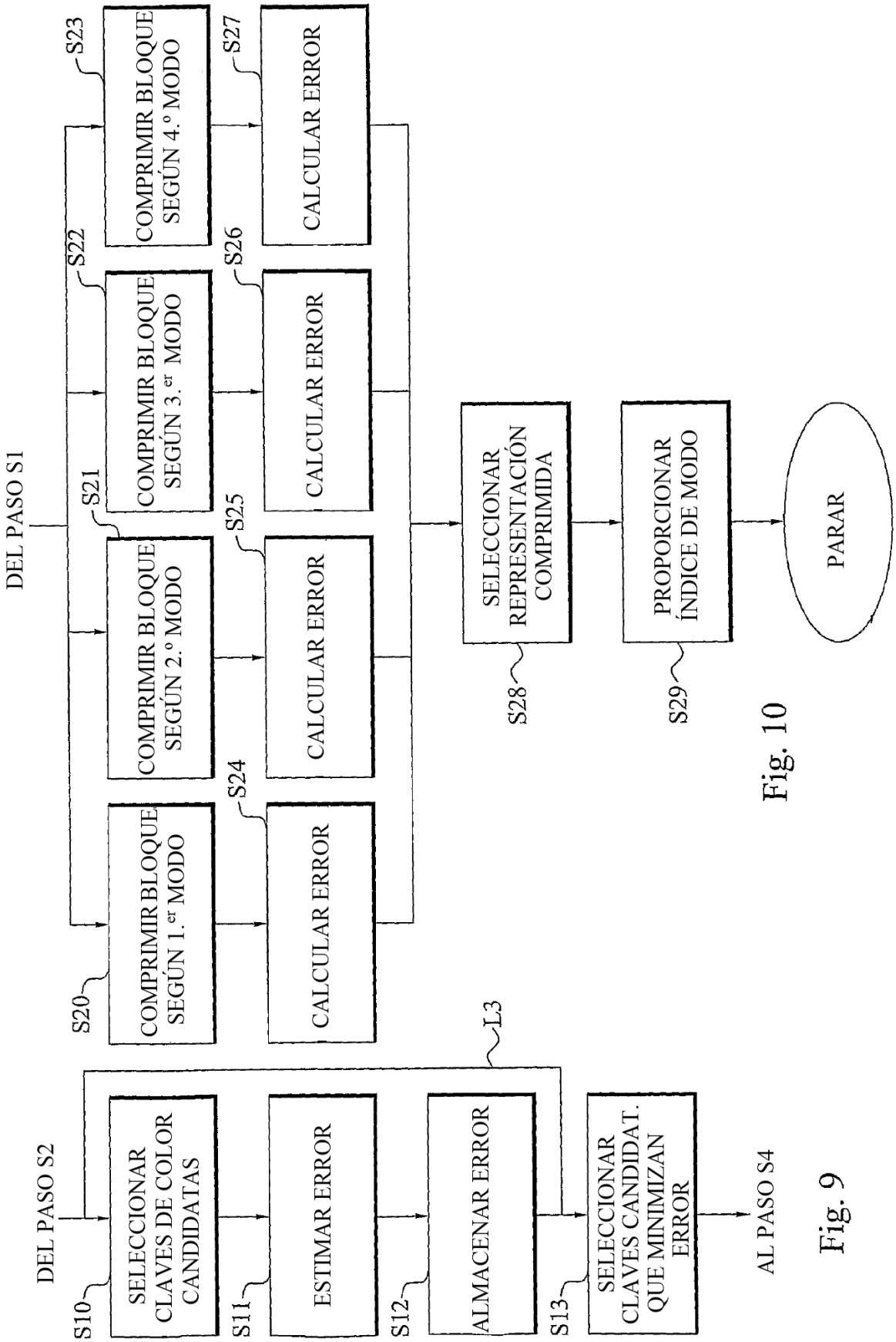
DEL PASO S1
S20
COMPRIMIR BLOQUE SEGÚN 1.er MODO
S21
COMPRIMIR BLOQUE SEGÚN 2.º MODO
S22
COMPRIMIR BLOQUE SEGÚN 3.er MODO
S23
COMPRIMIR BLOQUE SEGÚN 4.º MODO
S24
CALCULAR ERROR
S25
CALCULAR ERROR
S26
CALCULAR ERROR
S27
CALCULAR ERROR
SELECCIONAR REPRESENTACIÓN COMPRIMIDA
S28
PROPORCIONAR ÍNDICE DE MODO
S29
PARAR
L3
DEL PASO S2
S10
SELECCIONAR CLAVES DE COLOR CANDIDATAS
S11
ESTIMAR ERROR
S12
ALMACENAR ERROR
S13
SELECCIONAR CLAVES CANDIDAT. QUE MINIMIZAN ERROR
AL PASO S4

Fig. 10

Fig. 9

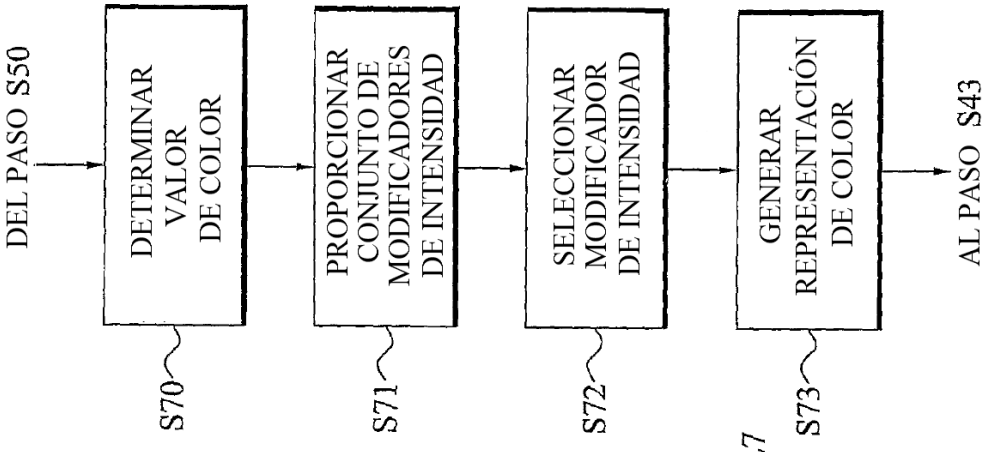

Fig. 21

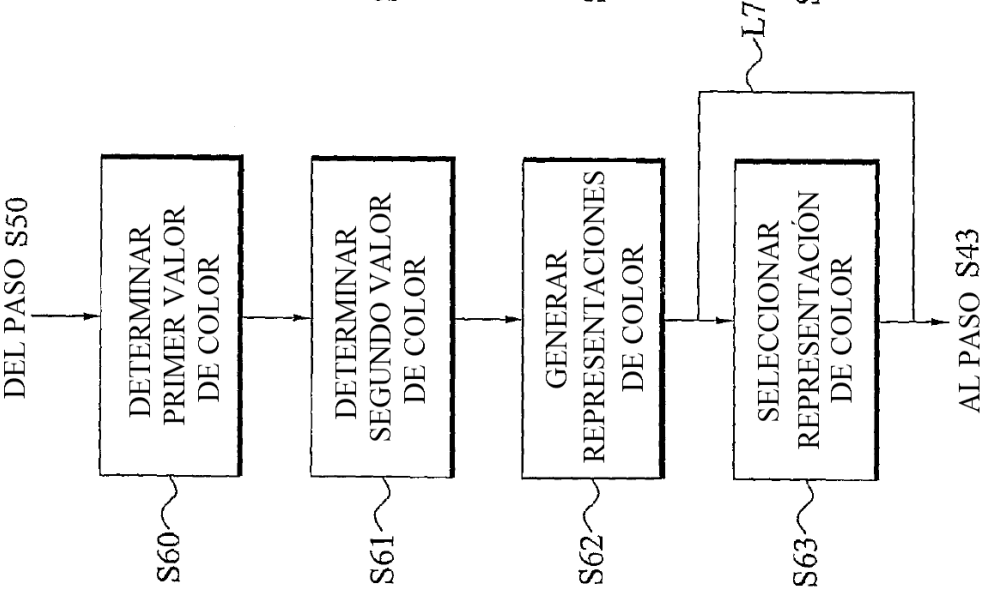

Fig. 20

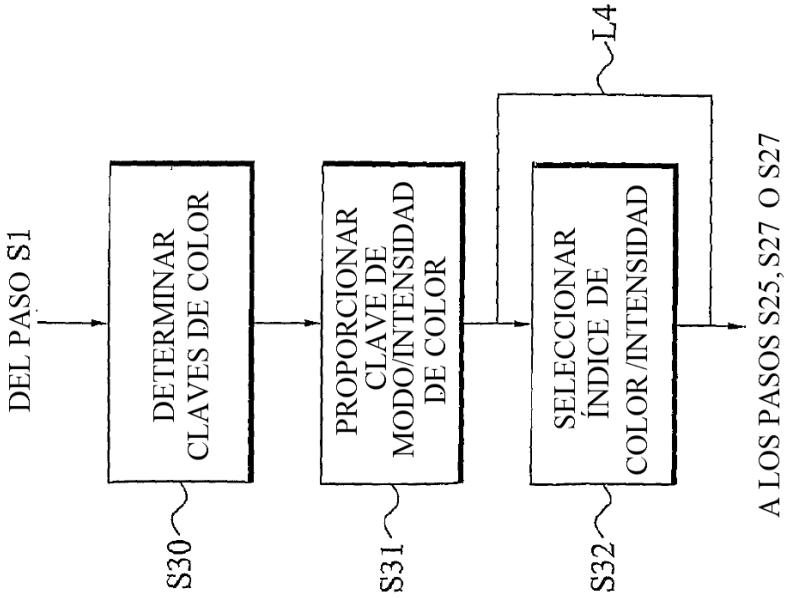

Fig. 11

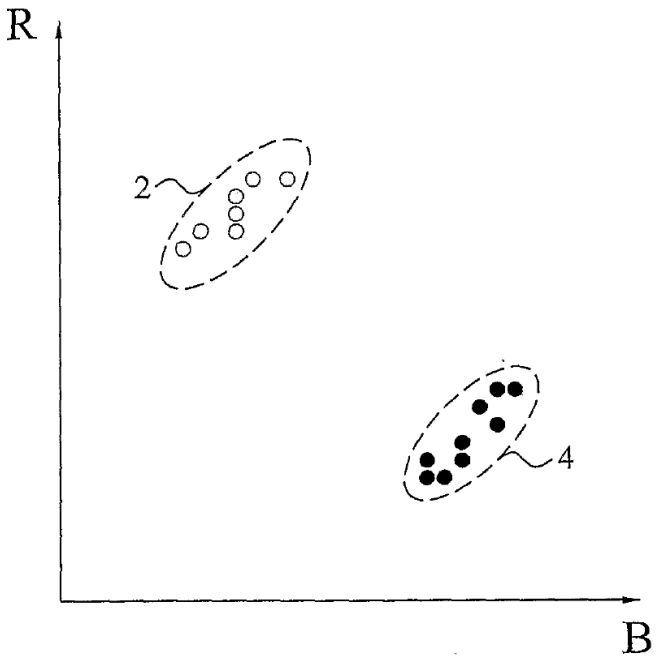

Fig. 12A

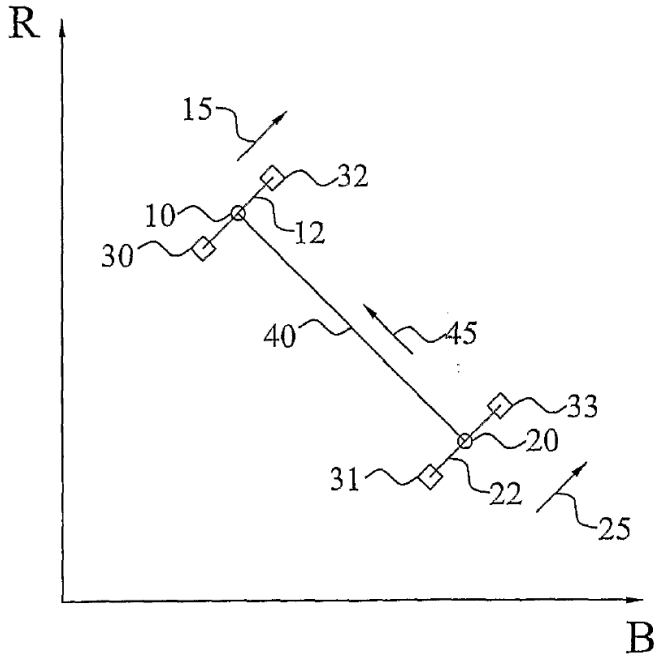

Fig. 12B

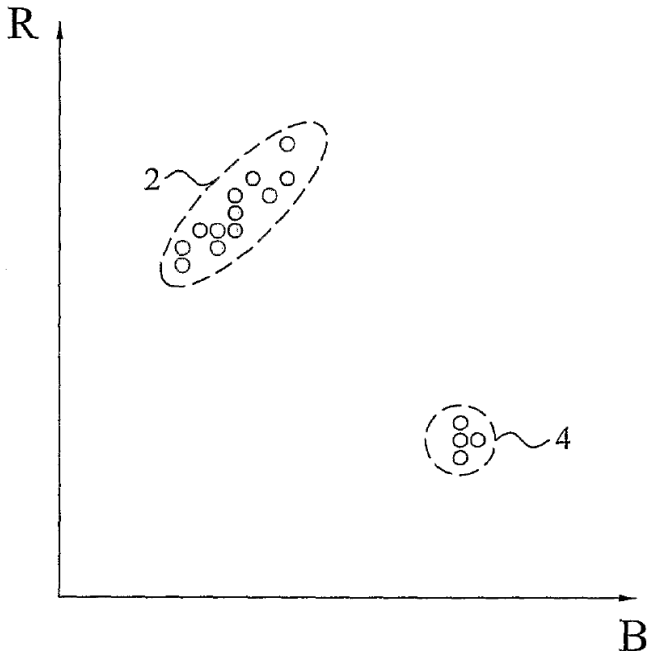

Fig. 13A

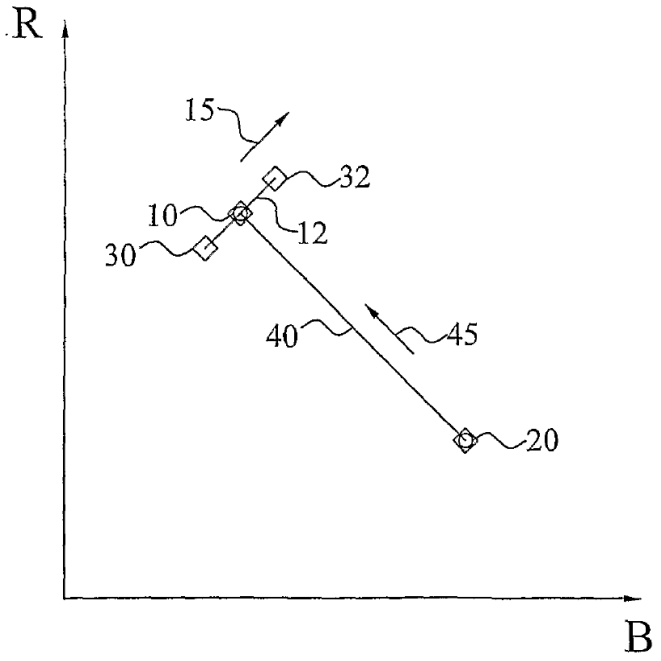

Fig. 13B

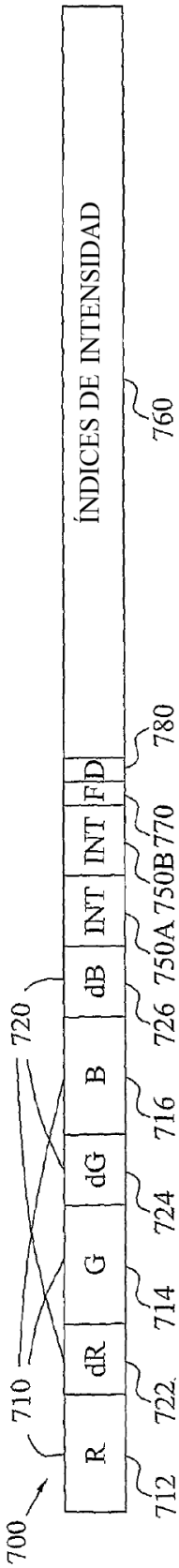

Fig. 14

Fig. 15

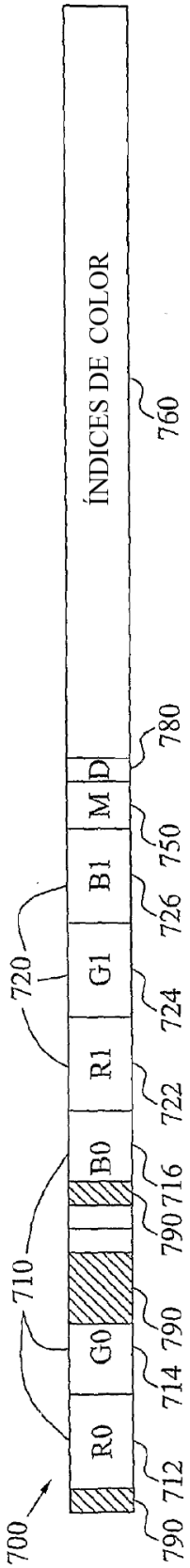

Fig. 16

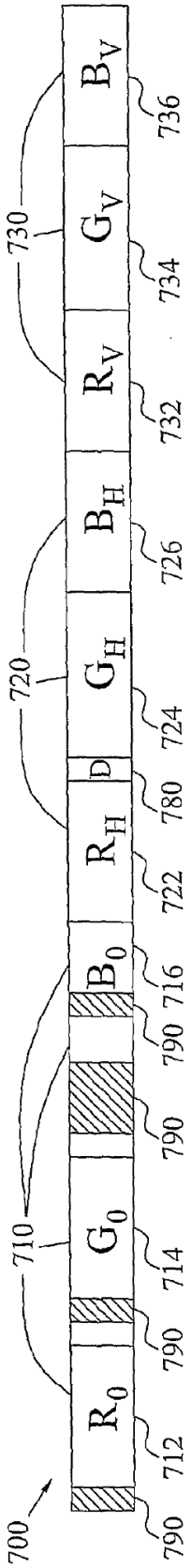

Fig. 17

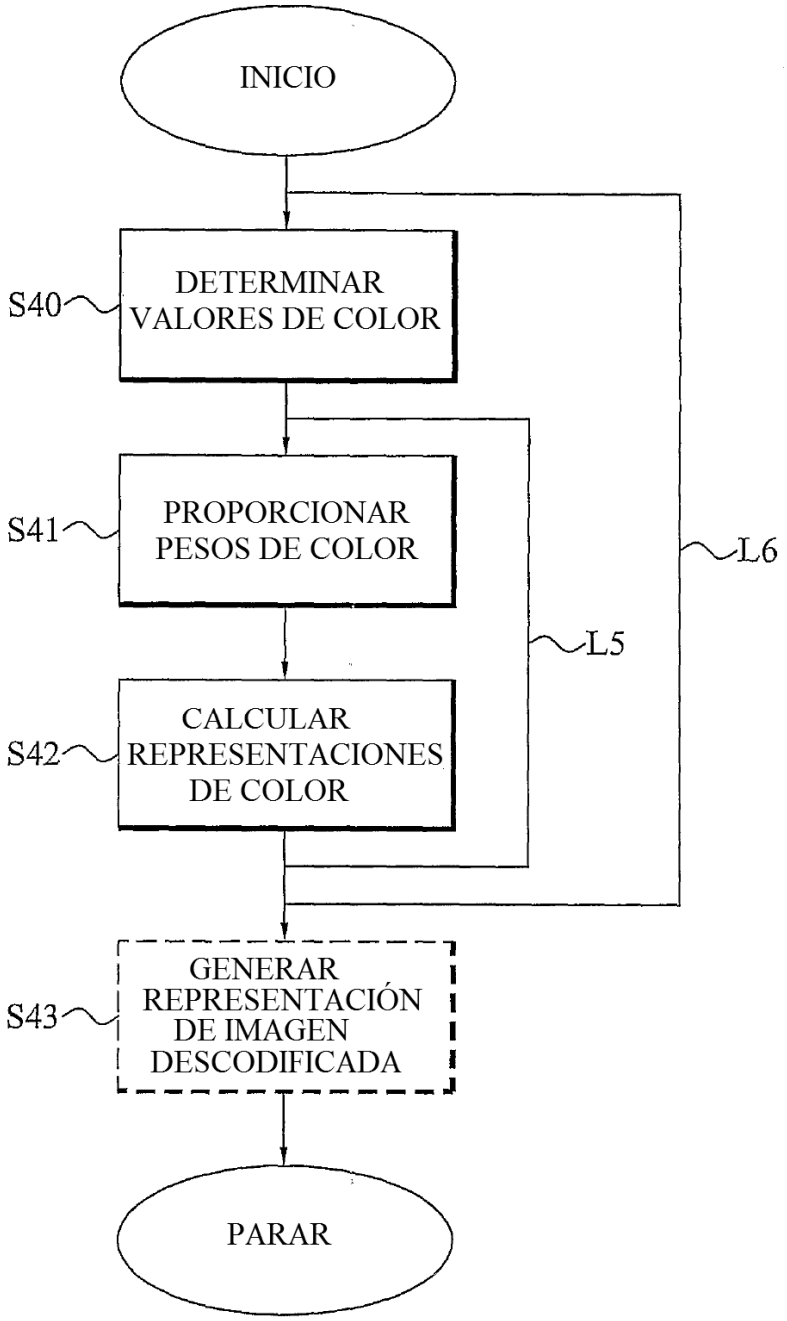
INICIO
S40
DETERMINAR VALORES DE COLOR
S41
PROPORCIONAR PESOS DE COLOR
S42
CALCULAR REPRESENTACIONES DE COLOR
S43
GENERAR REPRESENTACIÓN DE IMAGEN DESCODIFICADA
PARAR
L5
L6

Fig. 18

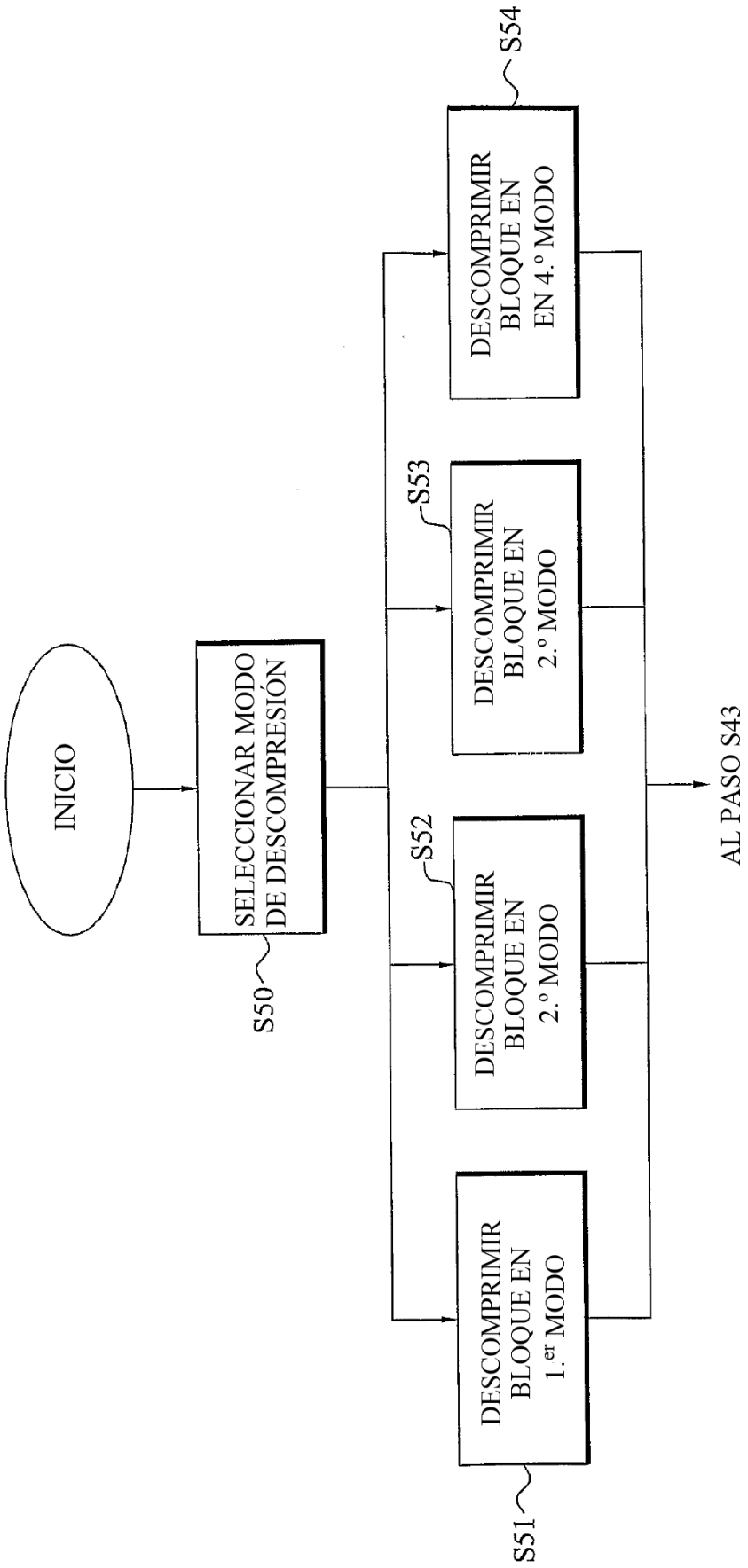
INICIO
SELECCIONAR MODO DE DESCOMPRESIÓN
S50
DESCOMPRIMIR BLOQUE EN 1.er MODO
S51
DESCOMPRIMIR BLOQUE EN 2.º MODO
S52
DESCOMPRIMIR BLOQUE EN 2.º MODO
S53
DESCOMPRIMIR BLOQUE EN EN 4.º MODO
S54
AL PASO S43

Fig. 19

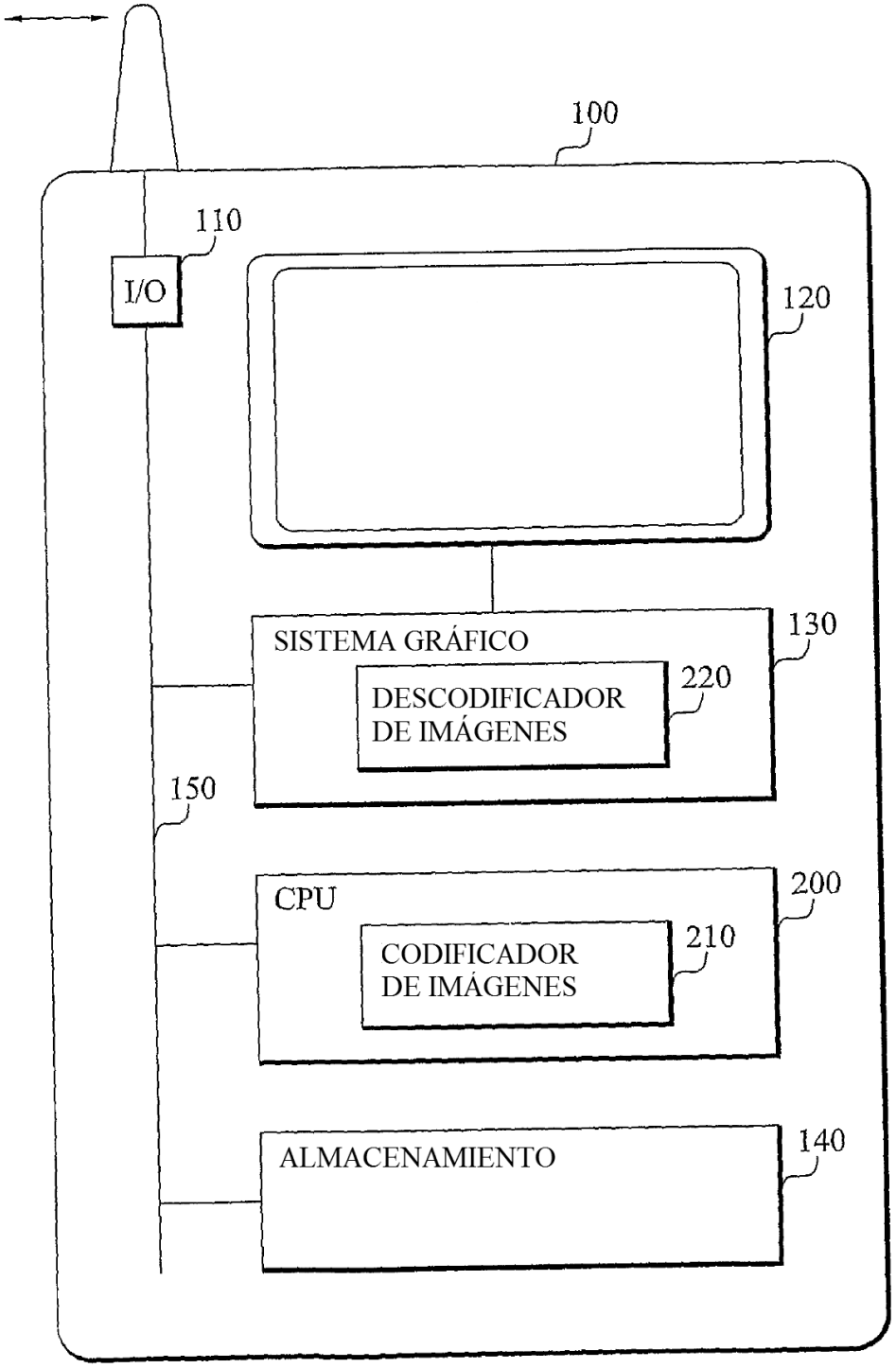
100
110
I/O
120
SISTEMA GRÁFICO
130
DESCODIFICADOR
DE IMÁGENES
220
150
CPU
200
CODIFICADOR
DE IMÁGENES
210
ALMACENAMIENTO
140

Fig. 22

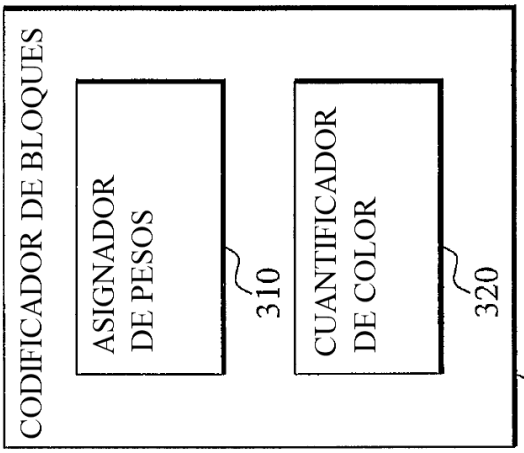
CODIFICADOR DE BLOQUES
ASIGNADOR DE PESOS
310
CUANTIFICADOR DE COLOR
320
300

Fig. 24

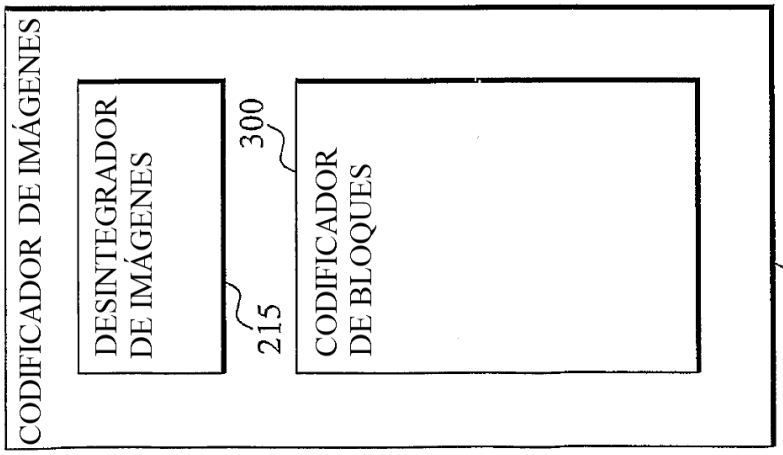
CODIFICADOR DE IMÁGENES
DESINTEGRADOR DE IMÁGENES
215
300
CODIFICADOR DE BLOQUES
210

Fig. 23

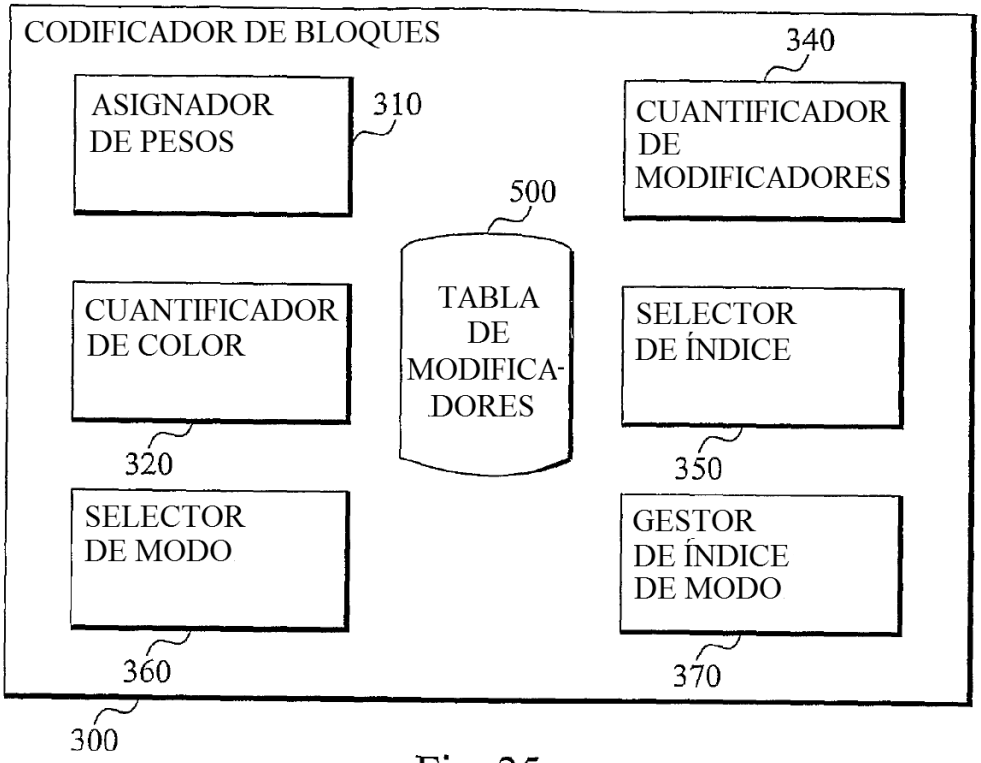
CODIFICADOR DE BLOQUES
ASIGNADOR DE PESOS
310
CUANTIFICADOR DE MODIFICADORES
340
500
TABLA DE MODIFICA-DORES
CUANTIFICADOR DE COLOR
320
SELECTOR DE ÍNDICE
350
SELECTOR DE MODO
360
GESTOR DE ÍNDICE DE MODO
370
300

Fig. 25

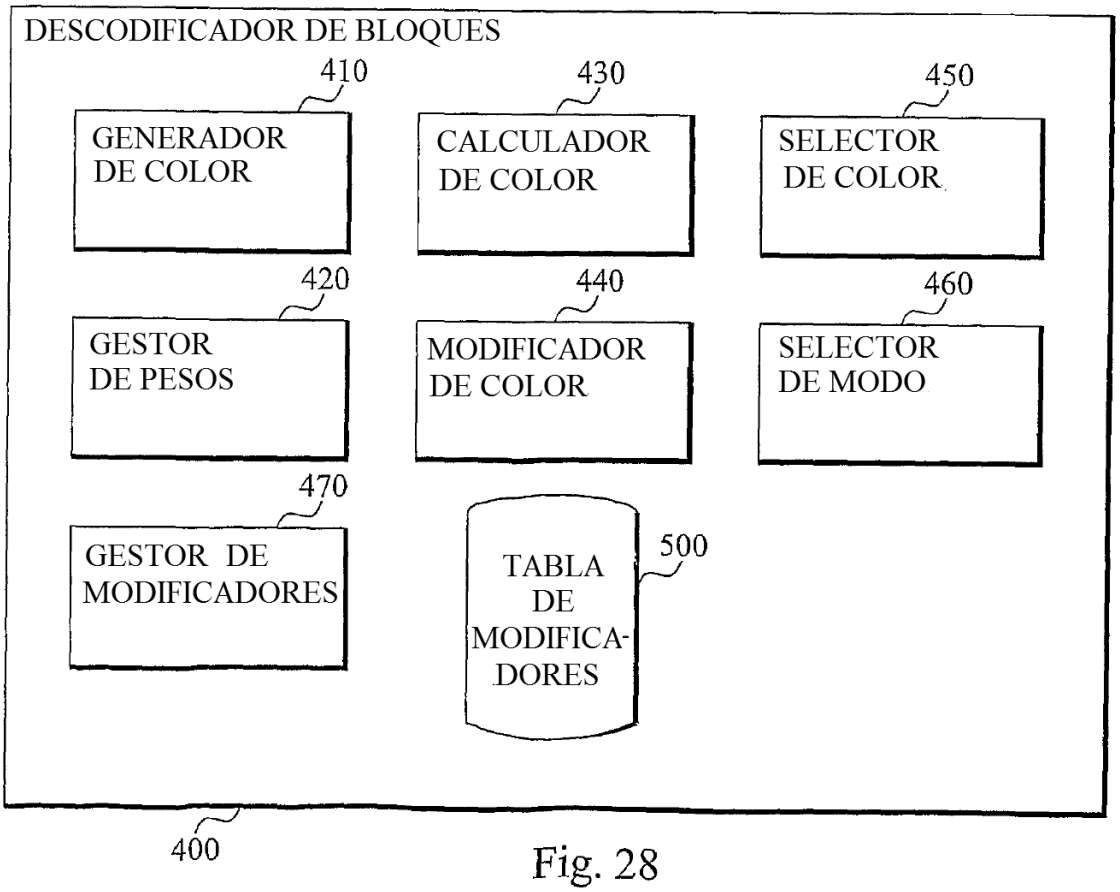
DESCODIFICADOR DE BLOQUES
410
GENERADOR DE COLOR
430
CALCULADOR DE COLOR
450
SELECTOR DE COLOR
420
GESTOR DE PESOS
440
MODIFICADOR DE COLOR
460
SELECTOR DE MODO
470
GESTOR DE MODIFICADORES
500
TABLA DE MODIFICA-DORES
400

Fig. 28

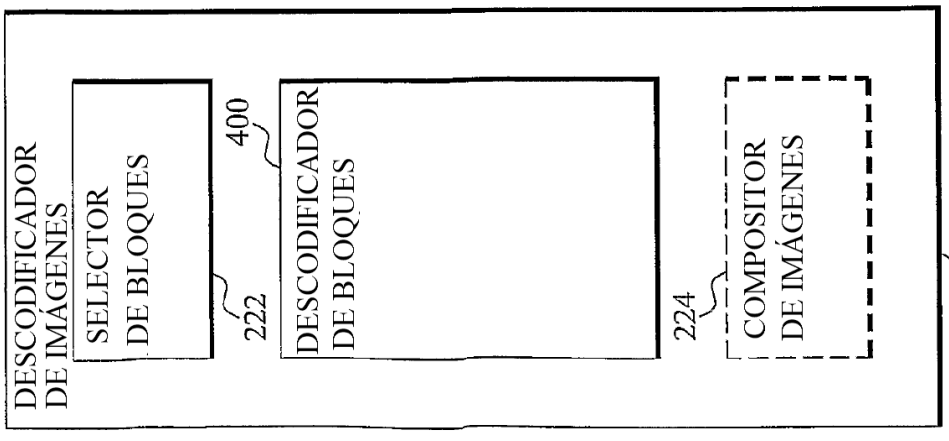
DESCODIFICADOR DE IMÁGENES
SELECTOR DE BLOQUES
222
400
DESCODIFICADOR DE BLOQUES
224
COMPOSITOR DE IMÁGENES
220

Fig. 26

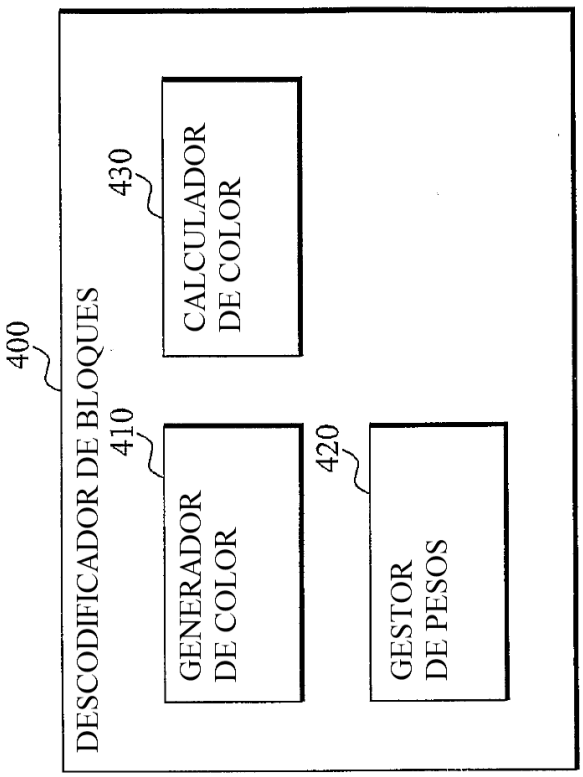
400
DESCODIFICADOR DE BLOQUES
410
GENERADOR DE COLOR
420
GESTOR DE PESOS
430
CALCULADOR DE COLOR

Fig. 27